Hindawi
Security and Communication Networks
Volume 2022, Article ID 2554280, 19 pages
https://doi.org/10.1155/2022/2554280

Research Article

Intrusion Detection Method for In-Vehicle CAN Bus Based on Message and Time Transfer Matrix

Zixiang Bi,[1] **Guoai Xu**,[1] **Guosheng Xu**,[1] **Miaoqing Tian**,[2] **Ruobing Jiang**,[2] **and Sutao Zhang**[1]

[1]*School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*Department of Computer Science and Technology, Ocean University of China, Qingdao 266100, China*

Correspondence should be addressed to Guoai Xu; xga@bupt.edu.cn

Received 9 November 2021; Revised 30 December 2021; Accepted 26 January 2022; Published 7 March 2022

Academic Editor: Yunchuan Guo

As the number and computational power of electronic computing units installed in standard automobiles continue to increase, contemporary motor vehicles face more cybersecurity threats than previous designs, while providing greater convenience and various useful features. Although vehicles are attacked at various entry points, eventually, attacks are injected into the in-vehicle controller area network (CAN) to cause vehicle anomalies. Currently, OEMs and research fields have implemented protection for the CAN bus in terms of external interfaces, internal protocols, and intrusion detection. Although the deployment of intrusion detection solutions is the most effective approach, the main challenges currently faced by automobile intrusion detection algorithms in practice involve limited computing resources, insufficient real-time responsiveness, and low recognition accuracy. In this study, we propose a novel intrusion detection method based on the message and time transfer matrix to address these difficulties, which can be applied to the vehicle Electronic Control Unit (ECU) to achieve real-time attack signal identification with high accuracy. Experiments on actual vehicles show that the proposed algorithm identified various attacks with high accuracy while consuming less computational and storage resources than previous methods. Moreover, the efficiency of the proposed algorithm is not affected by the attack injection frequency. Compared with other methods, the proposed method achieved better attack identification performance. Additionally, the message and time transfer matrix used by the algorithm can be used as a message transfer fingerprint of the CAN bus to discover anomalies.

1. Introduction

With the general increase in available computational power and the number of Electronic Control Units (ECUs) [1, 2], automobiles have become more intelligent and networked. Vehicles are no longer only a means of transportation but have become a smart terminal with computing and communication capabilities. Nowadays, intelligent vehicles improve the driving experience and provide drivers with greater convenience, such as through more accurate assisted driving and more diverse media access. Aside from comfort and functionality, security threats to cars have increased [3, 4] and drawn attention from the research community. Researchers have demonstrated the possibility of attacks against intelligent vehicles [3–5]. In one poignant example, in 2015, Miller et al. exploited a CAN bus-connected entertainment system and ECU firmware to attack a Jeep Cherokee driving on the highway, causing a series of severe consequences such as loss of vehicle power, radio problems, and uncontrolled water spray [6]. Moreover, Rogers et al. succeeded in stopping a Tesla in place and shutting down the car by sending commands via an iPhone [7]. In 2019, Keen Labs in China exploited vulnerabilities in assisted driving systems to drive vehicles into the reverse lane and even control vehicles' steering remotely using a gamepad [8]. In 2021, two researchers from Kunnamon used a drone to control the infotainment system of a Tesla via Wi-Fi and conducted a series of malicious operations to open doors, change steering, and acceleration modes [9]. Although the entry points for attacks against vehicles vary,

all these attacks cause anomalies in the vehicle's internal network, leading to anomalous vehicle behavior. The conventional controller area network (CAN) is commonly used for internal vehicle networks. CAN is a reliable serial bus that provides communication between individual ECUs. However, the CAN bus uses broadcasts for communication and has no encryption or authentication mechanisms. Once hackers gain access to the CAN network from other entry points, they can attack the vehicle control unit, severely threatening the vehicle occupants and pedestrians.

Original equipment manufacturers (OEMs) and research organizations continue to strive to find protective measures to address the security of the existing CAN bus. Two protection strategies are currently used by OEMs to prevent the CAN bus from being compromised. The first is to privatize the Database CAN (DBC) file that describes the meaning of CAN messages to prevent attackers from sending control messages to the CAN network. The other is to filter anomalous messages at the OBD-II interface, which is the CAN network's external physical interface. However, works [10–12] have already explored methods to reverse engineer the DBC file used by OEMs for protection. In addition, it is easy for hackers to find other entrances aside from the OBD-II interface to defeat the second security protection to execute an attack. Two main types of CAN protection measures have also been proposed in the relevant literature. The first approach is to add encryption and authentication security mechanisms to the ECU, which will inevitably increase the production cost of the vehicle and the transmission delay of CAN messages and cannot be used in vehicles with time-sensitive control systems. The other is to deploy intrusion detection algorithms in the vehicle to detect and protect against intrusions, which is currently possible but still has several challenges. Practical intrusion detection algorithms mainly include machine learning-based and single-feature lightweight algorithms. Machine learning-based intrusion detection algorithms involve the challenge of limited or deployment in resource-constrained ECUs, and their required computation time does not meet the real-time requirements of automotive systems. Lightweight intrusion detection algorithms do not meet the accuracy requirements of automotive applications, although they solve the latency and resource problems.

To solve the problems of current automotive CAN bus intrusion detection algorithms related to insufficient real-time response, low accuracy, and high resource occupation, in this study, we propose a CAN intrusion detection algorithm with high accuracy, low latency, and low resource consumption. We analyze CAN messages and time transfer characteristics and construct a multidimensional matrix of message ID transfer, data transfer, and time transfer based on the results of feature analysis. This matrix and message association analysis are used in the detection algorithm to achieve fast and accurate identification of intrusion messages in daily automotive driving scenarios. The critical contributions of the proposed method are summarized as follows.

(i) In this study, an innovative attack detection method based on a message and time transfer matrix is proposed to detect anomalous attacks in real time, having a high recognition rate and requiring less resources than existing methods.

(ii) This study uses actual vehicle data to model, validate, and detect. The results show that the method is sufficiently accurate and unaffected by the speed of attack injection in both stationary and moving vehicles.

(iii) Compared to existing methods, the proposed method consumes fewer ECU resources, having a shorter response time and higher accuracy. These characteristics make it more suitable for real-world automotive electronics.

The remainder of this study is organized as follows. Chapter 2 provides a preliminary overview of the CAN bus, describes CAN network-based attack scenarios and models, and analyzes other related works. Chapter 3 analyzes the message transfer and time transfer characteristics of CAN messages. Chapter 4 describes the data collection and injection devices used by the algorithm, the model construction and verification process, and the detection scheme. Chapter 5 describes the accuracy results, time and resource performance, performance of the method in actual vehicle experiments with different attack injection frequencies, advantages over other available schemes, and applications in typical CAN network architectures. Chapter 6 presents a scheme using message and time transfer matrices as fingerprints for CAN data transfer. Finally, Chapter 7 summarizes the work of this paper and provides some directions for future research.

2. Background and Related Work

2.1. CAN Bus Preliminary. As shown in Figure 1, a standard CAN frame begins with the start of frame (SOF), which is explicit and allows all nodes to be hard-synchronized. The SOF is followed by the arbitration field, which consists of an 11-bit identifier (ID) and remote transmission request (RTR). The identifier bit is used for arbitration when multiple nodes send data simultaneously, and the smaller the identifier, the higher the priority. Furthermore, RTR is used to distinguish the type of message. The arbitration field is followed by a 6-bit control field, where IDE and r0 specify the length of the frame, and the data length code (DLC) sets the number of bytes in the data field. Then, the data field of the CAN frame, which contains the car's control commands, status data, and any other data to be transmitted, is included. Subsequently, the circular check code (CRC) and the acknowledgment field (ACK) are used to detect and confirm if an error occurred in the transmission of the message. Finally, EOF identifies the end of the message.

In addition, there are two states in the CAN bus network, 0 is the explicit level, and 1 bit is the implicit level. If both a dominant and a recessive level are present, the status of the CAN bus is set to the dominant level. So when arbitrating

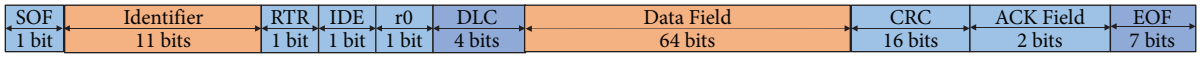


Figure 1: Standard CAN message frame.

between different IDs, the smaller the ID, the higher the priority.

For the attacker, the most meaningful information is the identifier and the data fields. Based on the arbitration mechanism of the CAN bus, an attacker can use the identification bits to implement denial of service (DoS) attacks. In addition, the attacker will focus on the control and status commands in the data field to perform fuzzing and replay attacks.

2.2. CAN Attack Surface and Model. This section analyzes the attack surface and attack models on the automotive bus based on the existing automotive CAN network architecture.

2.2.1. Attack Surface. The current typical CAN network architecture is shown in Figure 2. The in-vehicle CAN bus network is divided into several domains, including the chassis, body, power, and information. Within each domain, communication between individual ECUs is conducted through the CAN bus. Additionally, each domain is connected to a central controller, enabling interdomain communication through the central controller. The attack surface against this architecture can be mainly divided into contact attacks and noncontact attacks.

When implementing physical contact attacks, because the OBD-II interface is typically not easily found under the steering wheel, hackers can access an attack device at the OBD-II interface to inject anomalous messages. Additionally, the attacker may hitch the attack device to an exposed CAN connection to inject the attack. In noncontact attacks, the hacker controls the nodes in the CAN network to send attack messages to the CAN bus. The in-car entertainment system is the most vulnerable to hacking as a node for human-vehicle interaction. Once the node is compromised, the attacker can send any attack message remotely. Furthermore, as vehicle manufacturers are increasingly using Over-the-Air Technology (OTA) to upgrade ECU firmware, hackers can tamper with the upgrade package to take control of the ECU and consequently send arbitrary messages to the CAN bus. Either way, hackers send attack data to the CAN network either directly or indirectly to cause abnormal vehicle behavior.

2.2.2. Attack Model. We surveyed numerous research works on CAN bus attacks and concluded that the current effective attack models included DoS, fuzzy, and replay attacks. In the actual vehicle attack tests conducted by the researchers in this paper, it was found that there is also an ulterior fuzzy attack that has a higher efficiency in executing attacks against the CAN bus. Among the CAN attack models, the easiest to identify is the DoS attack, followed by the fuzzy attack. The ulterior fuzzy attack and the replay attack are more challenging to recognize. The following section describes the four CAN bus attack models.

(1) DoS Attack. All ECU nodes share the same bus resources, and no node is under the administrator's control. In this case, a malicious ECU can increase the bus occupancy without following the bus protocol, causing delays, or even suspending other messages. Therefore, hackers can implement DoS attacks by injecting high-priority messages into the bus in a short time, thus preventing the delivery of other CAN messages. An example of the DoS attack is depicted in Figure 3. The bus messages with higher priority (e.g., ID = 0x000) were randomly injected into the typical message traces at high speed. For example, a DoS attack can be implemented by injecting packets with an ID of 0x000 at a rate that is ten times the regular message interval.

(2) Fuzzy Attack. A fuzzy attack indicates that an attacker sends random compromised IDs and data to the CAN network at any time, even if the attacker does not have any specific information about the victim. The attacker is only required to send a malicious message in the same form as a normal CAN message all the time, and the vehicle may be successfully attacked. Unlike a DoS attack, the fuzzy attack paralyzes the vehicle's functionality or causes an abnormal ECU reaction instead of preventing regular message delivery by occupying the bus. Therefore, fuzzing attack messages consisting of random IDs and data are randomly inserted into the regular CAN bus traffic. An example of the attack is shown in Figure 4. When the fuzz attack was implemented on the test vehicle, the vehicle randomly performed abnormal reactions, such as abnormal engine noise, flashing lights, abnormal increase in power, and gear switching.

(3) Ulterior Fuzzy Attack. Unlike the basic fuzzy attacks, a savvy attacker may gain prior knowledge of the victim and narrow down the random data by eavesdropping on the bus data. A more advanced fuzzy attacker can perform a more efficient attack by first knowing all the valid IDs in the CAN bus and sending attack messages containing existing IDs and random data. An example of the ulterior fuzz attack is shown in Figure 5. Such attacks are realized efficiently by inserting some messages randomly. The IDs of these messages exist in typical message traces, and the data field is random. Experiments show that ulterior fuzz attacks can cause anomalous vehicle behaviors more frequently than fuzz attacks.

(4) Replay Attack. As shown in Figure 6, the replay attack causes problems by injecting a set of CAN messages extracted and recorded in a particular order into the CAN bus. Replay attack datasets can be generated by randomly inserting a segment of prerecorded CAN messages into the typical traffic trace. Experiments on the test car indicate that the replay attack could cause the car to reproduce some of its

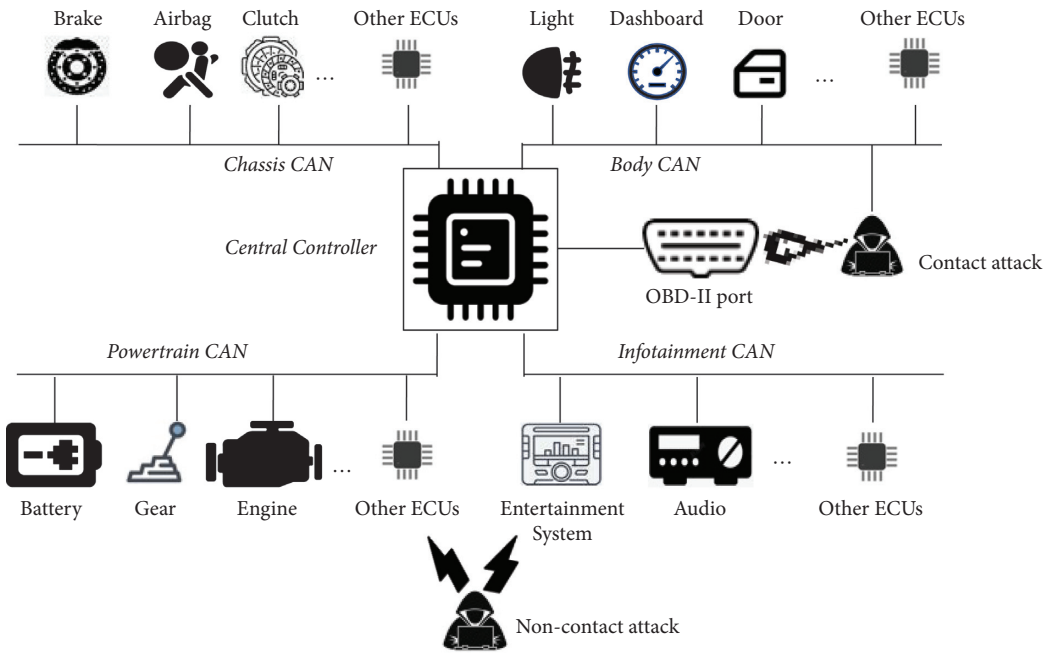


Figure 2: Typical CAN network architecture and attack entry.

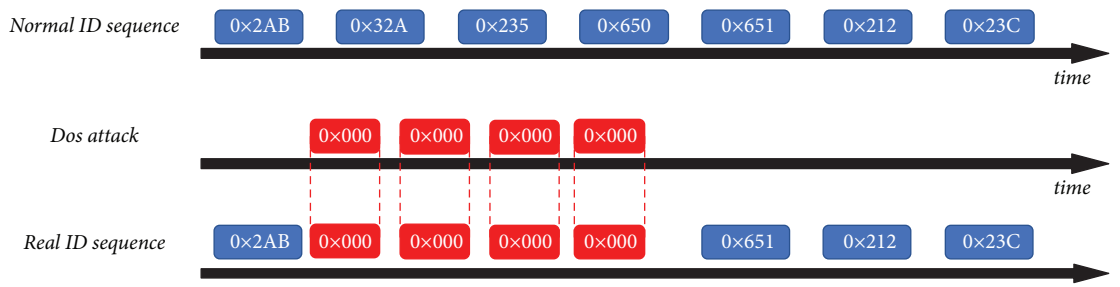


Figure 3: Diagram of a DoS attack on the CAN bus.

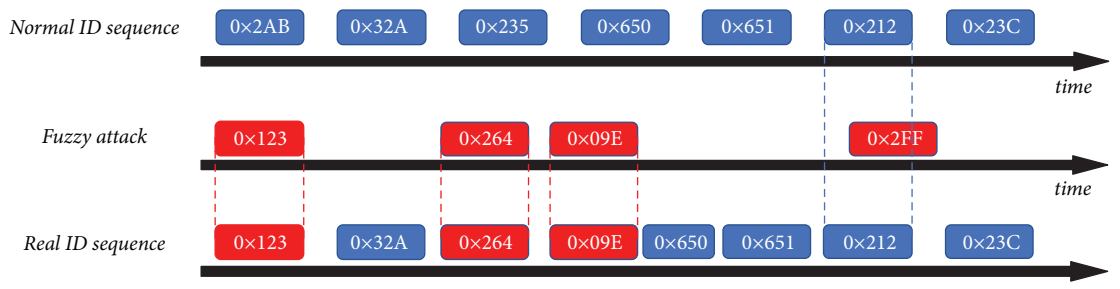


Figure 4: Diagram of a fuzzy attack on the CAN bus.

previous operations, such as increasing air conditioning wind, gear switching, and window lifting.

2.3. Related Work. To combat malicious attacks in the CAN bus, it is crucial to build anomaly detection algorithms by analyzing the characteristics of CAN messages to identify attack messages and provide accurate reference information for system protection. Several methods have emerged for CAN anomaly detection, but they fall short in accuracy, usability, and real-time. The first type of method relies on the physical features of the CAN bus for anomaly detection. For example, Cho and Shin proposed a method called Viden, which extracts the voltage characteristics of ECUs through voltage measurements to identify abnormal voltage values. This method can detect illegal access quickly and accurately [13]. Tian et al. proposed a temperature compilation fingerprinting technique to detect CAN bus intrusions and

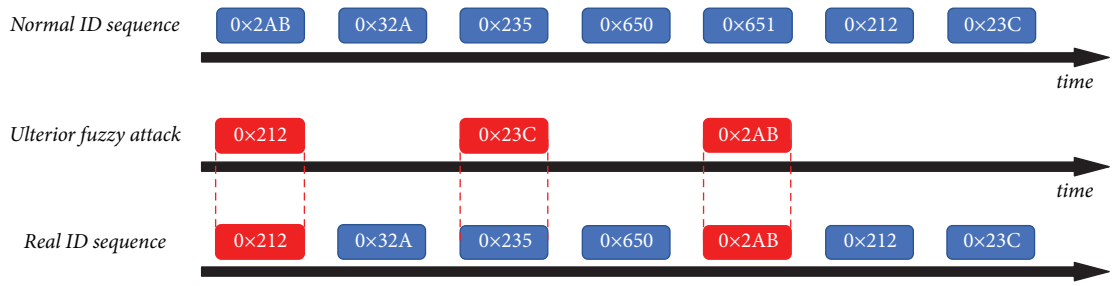


Figure 5: Diagram of an ulterior fuzzy attack on the CAN bus.

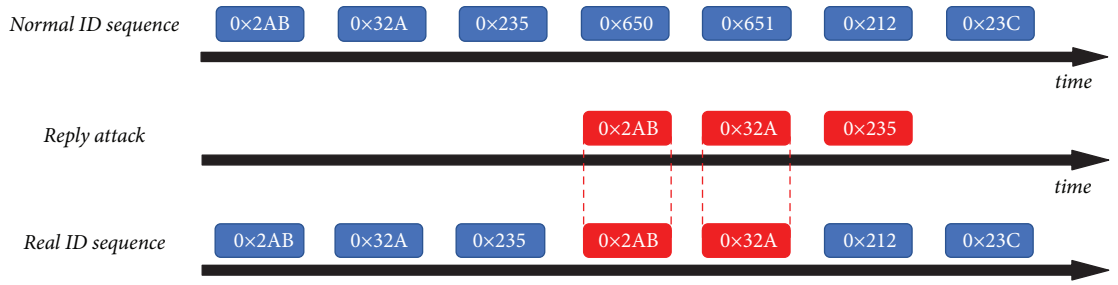


Figure 6: Diagram of a replay attack on the CAN bus.

identify intrusion sources [14]. However, these methods are only suitable for detecting attacks at the physical layer rather than the application layer. Furthermore, they require additional hardware for physical feature acquisition, increasing vehicle manufacturing overheads. The second type of method relies on legitimate CAN data analysis results for anomaly detection. For example, Song et al. analyzed the periodicity of CAN bus messages and found injected attack messages by examining the occurrence periods of the messages with the same ID [15]. Stabili et al. constructed a lightweight CAN intrusion detection algorithm to handle escalating attack messages that did not satisfy the Hamming distance requirement [16]. Additionally, they proposed an anomaly detection algorithm for ID sequences to discover illegitimate ID transfers [17]. Lee et al. identified a fixed pattern of response offset versus time for remote frames, and this pattern will be broken when an attack occurs. They exploited this pattern to detect attacks [18]. These methods used lightweight algorithms for the situation where ECU resources are constrained. However, the algorithms can only discover attacks from a single dimension, obtaining poor accuracy in anomaly identification, particularly in identifying advanced attacks. The third type of method uses information theory to identify anomalous attacks. For example, Yu et al. proposed an intrusion detection system based on the estimation of multiorder Rényi entropy [19]. Wu et al. proposed an information entropy-based approach that uses a fixed amount of information as a sliding window to improve the accuracy of attack identification and reduce false positives [20]. Although this type of method can effectively identify anomalies, it cannot make judgments immediately after an attack occurs and does not meet the requirements of rapid automotive response. Furthermore, there is another type of method based on machine learning algorithms for CAN attack detection. For example, Tariq et al. used a rule-based approach and a recursive neural network algorithm to detect anomalous messages with an accuracy exceeding that of previous algorithms collaboratively [21]. Song et al. proposed an intrusion detection system based on a deep convolutional neural network (DCNN) [22]. The DCNN learned network traffic patterns and detected malicious traffic, reducing the false alarm and missed alarm rates. Seo et al. proposed a GAN-based CAN bus intrusion detection system that uses deep learning models to generate adversarial networks to improve the identification rate of attacks [23]. Wang et al. proposed a distributed anomaly detection system using hierarchical time memory (HTM) [24]. They designed predictors exploiting the HTM algorithm and logarithmic loss function to calculate anomaly scores. These methods identify attacks with high efficiency, but they cannot be fully applied to vehicles owing to their limited computational resources.

3. CAN Message and Time Transfer Characteristics Analysis

Based on the investigation of existing studies and actual vehicles, in this section, we analyze the data characteristics and time characteristics of data frame transfer in the automotive CAN bus. In addition, mainstream electric cars, sedans, and sport utility vehicles (SUV) in the market were investigated to illustrate the correctness of the results.

3.1. Message Transfer Feature. First, we analyzed the transfer characteristics of the IDs used for arbitration in CAN messages. By analyzing the CAN bus traces of various types of vehicles in daily driving scenarios, this study finds that

each ID follows only a subset of all available IDs. This result implies limited rules for the transfer of two neighboring IDs rather than being completely random. Figures 7 and 8 depict the ID transfer graph for two commercially available mainstream vehicles with the previous ID on the x-axis and the following ID on the y-axis. The dark color in the graph indicates the presence of a legitimate ID transfer from x to y, while the blank indicates no transfer. From this result, it may be observed that there were 39 available IDs for electric cars but only 780 legitimate transfers. In addition, 81 legal IDs exist for the sedan, but only 4542 ID transfers are present. This result is consistent with the finding of Stabili and Marchetti [17]. Based on this conclusion, the ID transfer rule of CAN messages can be used as a feature to detect intrusions, especially for DoS attacks and fuzzy attacks.

In addition, the transfer characteristics of the CAN message data fields are investigated in this paper. By analyzing the variation of adjacent CAN message data fields for different vehicles, this study finds that the distance of data bits fluctuates within an interval rather than being completely random, regardless of the distance model used. Because anomalies are usually caused by colliding control commands randomly when implementing attacks on the CAN bus, we investigated the collision resistance of each legal transfer under different models based on the transfer characteristics of the data fields. Specifically, we used the Hamming distance [25], Levenshtein distance [26], cosine distance [27], Jaro distance [28], and Jaro-Winkler distance [29] for an electric vehicle to calculate the distance interval for each legitimate message transfer. Then, the distance intervals are collided with the fuzzy attack to obtain the best distance model for each legitimate transfer against the fuzzy attack. The best model was applied to achieve accurate anomaly aware messages.

3.2. Time Transfer Feature. This study analyzed the time interval of each legal message transmission in CAN traffic in all test vehicles. We found that the time intervals of message transmission all satisfied the following four distributions. The first type of time interval is a discrete constant, as shown in Figure 9(a), with an integer between 228 and 242. The second type of time interval includes continuous values, as shown in Figure 9(b), with a continuous interval between 240 and 325.The third type approximates time intervals satisfying a one-sided normal distribution, as shown in Figure 9(c). As shown in Figure 9(d), the fourth type contains completely random values for a time interval of a random number between 0 and 6000.

Because injecting anomalous messages into CAN is entirely random, the time interval can be essential for detecting anomalous behavior. For interval distributions that satisfy discrete and continuous values, intrusions that do not meet their distributions can be detected directly and efficiently. For cases that satisfy the normal distribution, the interval of messages can be filtered by using the PauTa criterion to select the best confidence interval. For the utterly random case, the maximum variation range is used as an interval to discover intrusion messages outside the interval. The time interval of the message is unavailable for an attacker, so the attack injection is entirely random. If the time interval distribution of CAN messages is used as a feature to detect anomalies, then they can be identified without distinguishing the type of attack.

4. Methodology of In-Vehicle Network Intrusion Detection

In this study, we analyze the attack surface of CAN networks as described above, demonstrating that the ultimate goal of intruding the CAN network is to send attack messages, irrespective of whether the CAN bus is directly contacted. In addition, this paper analyzes the current measures on CAN network protection in OEM and scientific fields in the related work. The current relatively effective method is to deploy an intrusion detection mechanism to discover intrusions and protect against them. However, existing methods still cannot meet accuracy, real time response, and resource requirements of network-connected vehicles.

To solve the difficulties faced by the current CAN bus intrusion detection, we designed a CAN intrusion detection model with high accuracy, low latency, and low resource consumption based on the message and time transmission characteristics of CAN messages, which achieved fast and accurate identification of four attacks using this model and up-down message correlation analysis. The overview of the method is shown in Figure 10. The algorithm constructs a transfer feature matrix by extracting messages and time transfers from CAN traces generated by regular driving. The attack messages are discovered in real-time using contextual correlation analysis based on the transfer feature matrix. Additionally, with its lightweight feature, the algorithm can be deployed in the central controller to identify cross-domain attacks and in resource-constrained ECUs to identify abnormal data in the domain.

This section first defines a set of variables used in the algorithm, then describes the devices used for data collection and algorithm detection, and finally describes the algorithm feature model construction, model validation, and anomaly detection.

4.1. Definition. The following variables are defined herein to better describe the process of algorithm model construction and algorithm detection.

(i) Ω_T: the message and time transfer matrix used to record CAN message and time transfer characteristics

(ii) m_i: the i-th CAN message

(iii) $data_i$: the data field of the m_i

(iv) id_i: the identifier of the m_i

(v) t_i: the timestamp of the m_i

(vi) D_{ij}: the list that records data distance characteristics from m_i to m_j

(vii) T_{ij}: the list used to record the time interval from the m_i to m_j

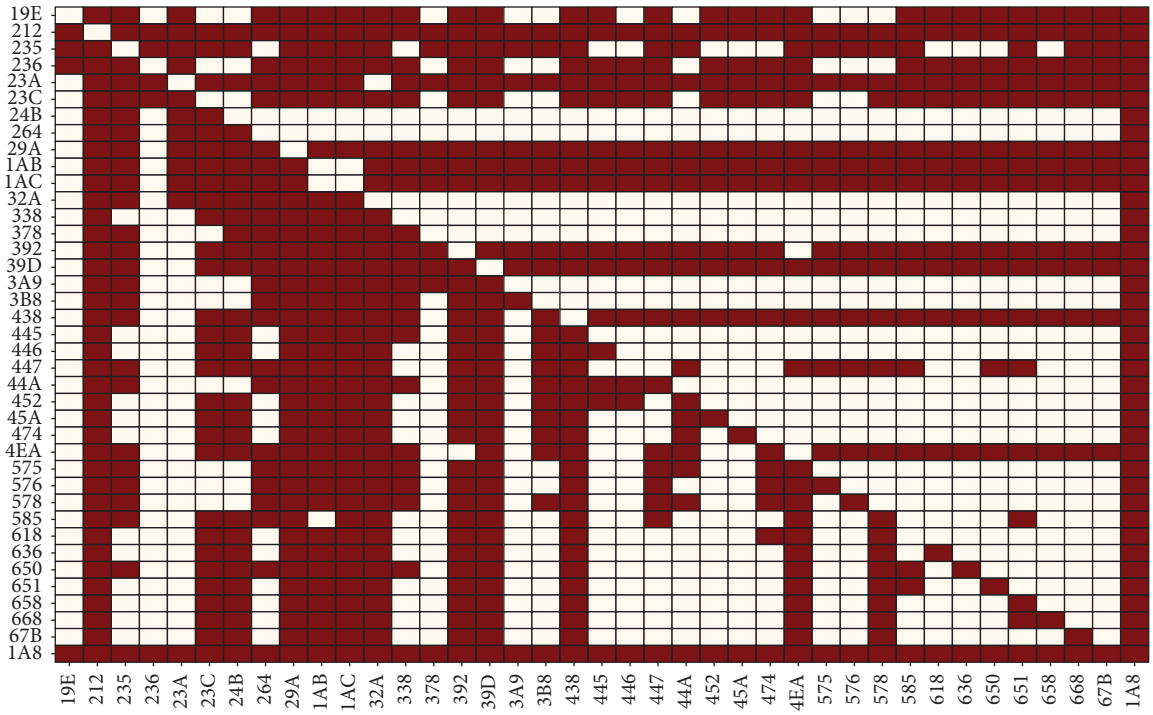


Figure 7: ID transfer of an electric car.

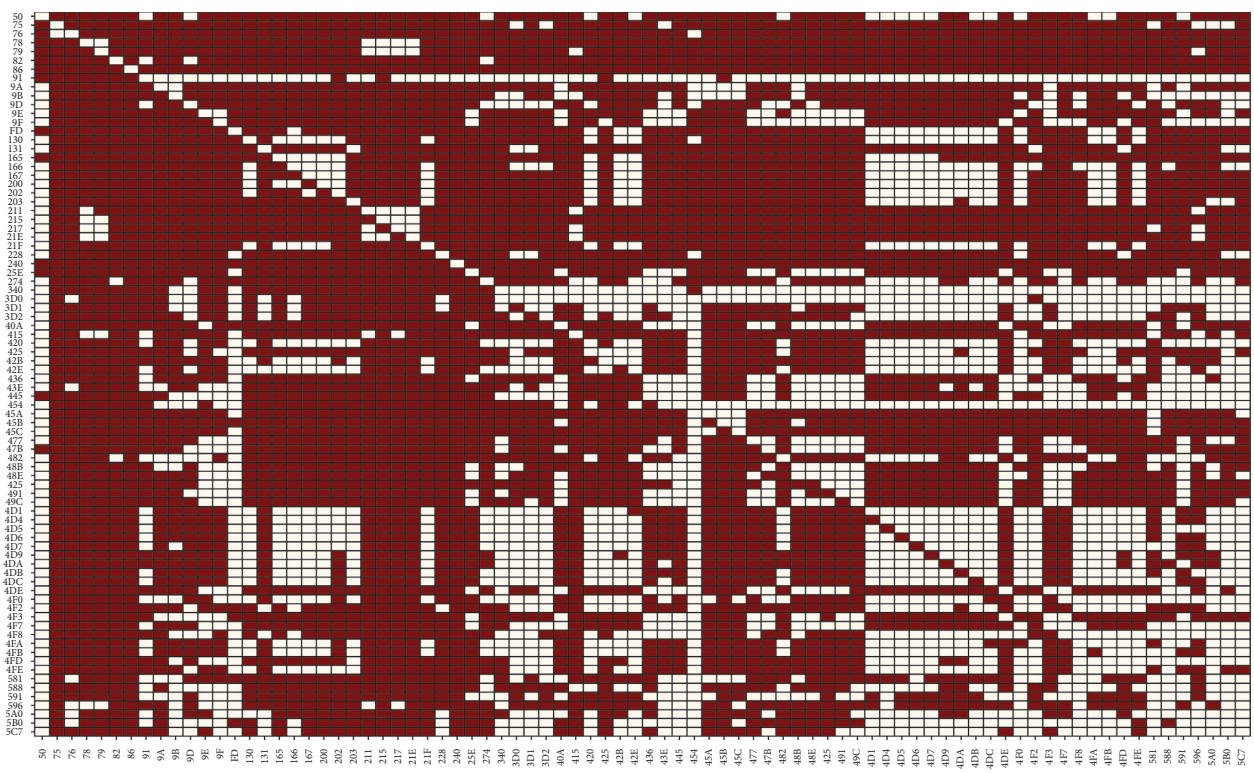


Figure 8: ID transfer of a sedan.

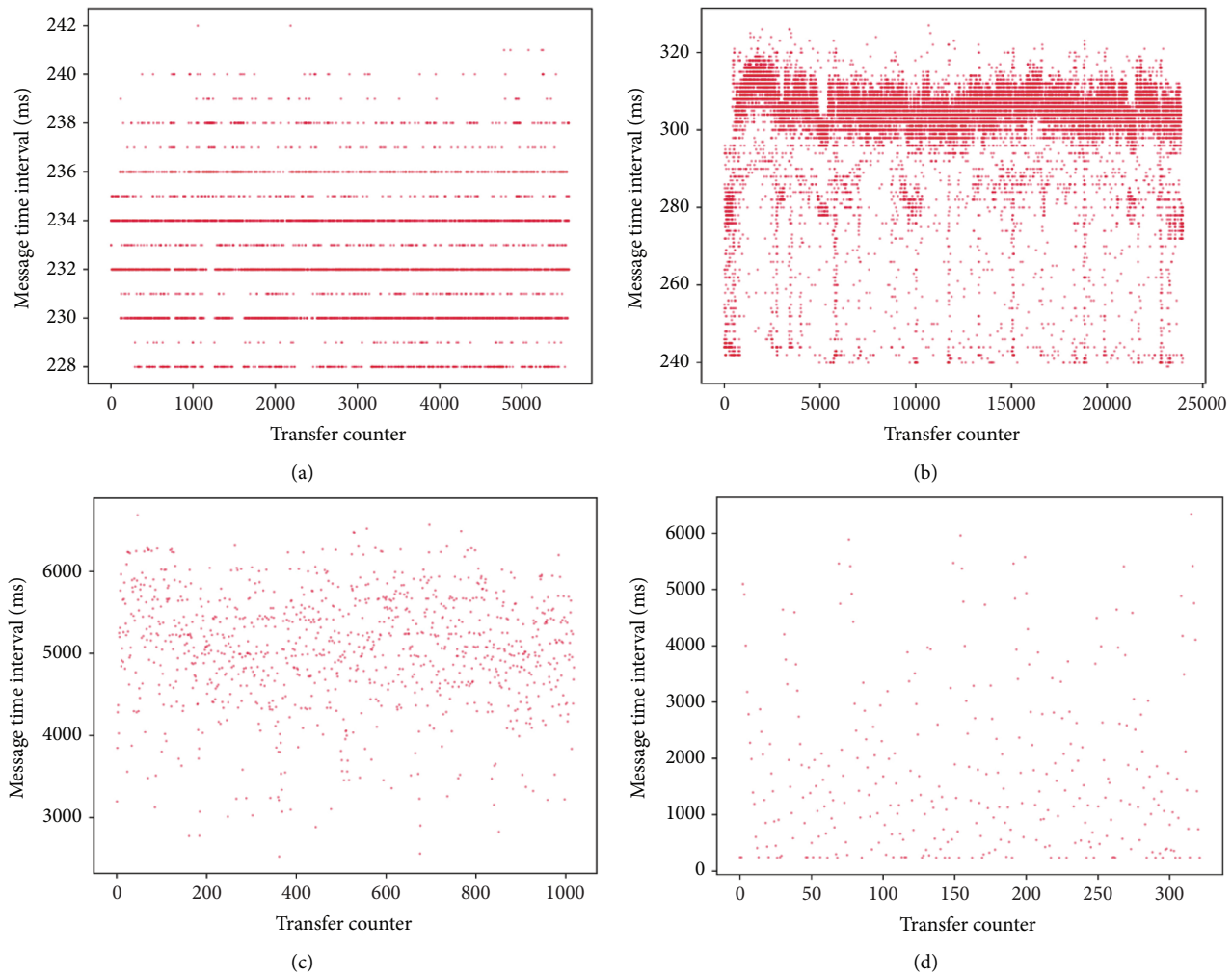


Figure 9: Time intervals distributions of the test vehicle. (a) Discrete time interval. (b) Continuous time interval. (c) Normal distribution time interval. (d) Random time interval.

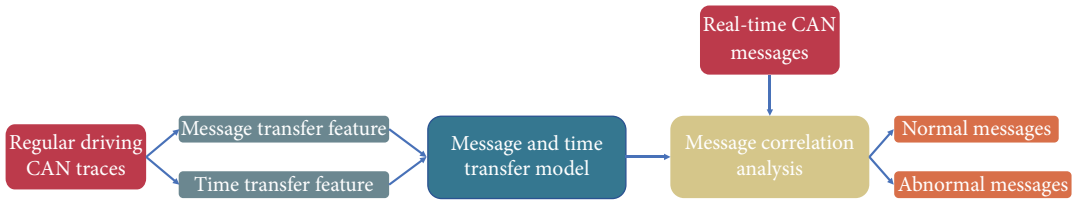
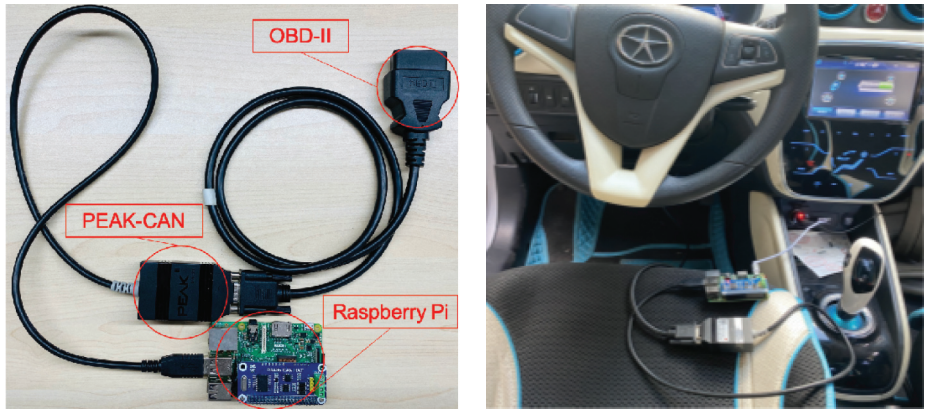


Figure 10: Algorithm overview.

(viii) $(ID_0, ID_1, ..., ID_n)$: the set of unique IDs containing all occurrence identifiers in the CAN traffic

4.2. Data Acquisition and Injection Equipment. The test vehicle used in this study was a Chinese-made electric vehicle with an internal network implementing the standard CAN bus protocol. Furthermore, the vehicle has an open OBD-II interface for message collection and injection (OBD-II is an interface for monitoring CAN bus data and obtaining diagnostic trouble codes). Additionally, the proposed algorithm is applied to the test vehicle to demonstrate its accuracy, real time responsiveness, and feasibility, because the vehicle has typical interfaces and bus protocols representative of most current commercially available vehicles.

The data acquisition device used is shown in Figure 11(a). We used a Raspberry Pi 3B+ and RS 485 expansion board for real-time communication with the vehicle CAN bus. The main relevant technical parameters of the Raspberry Pi are as follows. A 64-bit 1.2 GHz quad-core ARM Cortex-A53 CPU is installed, along with 1 GB of RAM and an 802.11n Wi-Fi wireless chipset. The RS 485 CAN expansion board is CAN-capable, and it uses the MCP2515



(a) (b)

Figure 11: Data acquisition equipment. (a) Data acquisition device. (b) Data acquisition setup.

CAN controller with an SPI interface and SN65HVD230 transceiver to receive CAN information. The setup for in-vehicle CAN message collection is shown in Figure 11(b). The acquisition device collects CAN traffic from the test vehicle at a baud rate of 500 kbps.

The message injection device was a combination cable comprising an OBD-II to DB9 diagnostic cable and a PCANUSB FD adapter. The cable was connected from the vehicle's OBD-II port to the USB port on the computer side to enable real-time attack message injection. It is noteworthy that we utilized a splitter to divide the OBD-II interface into two parallel hardware interfaces when detecting anomalies, as shown in Figure 12(a). One interface was connected to a laptop computer configured with the proposed algorithms for identifying intrusions. The other interface was connected to an attack device capable of sending four attack messages based on the attack model, as shown in Figure 12(b).

4.3. Model Construction and Validation. In Chapter 3, we analyzed the features of CAN messages and time transfer and illustrated their advantages in detecting various types of attack messages. Based on these facts, the model construction phase extracts the ID transfer, data transfer, and time transfer features from the CAN traces generated during everyday driving of the test vehicle and deposits them into the feature matrix Ω_T. Ω_T is a square matrix of order n, where n is the number of elements in the set of unique IDs $(ID_0, ID_1, ..., ID_n)$ in the CAN bus traces. The model construction flow is depicted in Figure 13.

First, the row and column index of Ω_T are initialized to the set of unique IDs $(ID_0, ID_1, ..., ID_n)$ in the initialization phase. The initialized values of the elements in the matrix are two empty lists D_{ij} and T_{ij}, which hold the data distance and time interval distribution of message transfers in CAN traces, respectively. Additionally, the optimal distance algorithm table for each type of message transfer is set considering the input dataset using the transfer characteristics of the CAN message data fields.

Next, when Ω_T is created, all legally transferred adjacent CAN messages must be examined. When two consecutive messages are input, the best distance model is selected based on the results of the characteristic analysis. The selected model is exploited to calculate the distance value, and the calculation result is stored in D_{ij}. Similarly, the time interval of adjacent messages is calculated, and the calculation result is appended to T_{ij}. As an example, if there are two legal adjacent CAN messages (m_i and m_j, where m_i with id_i, payload $da\ ta_i$, and timestamp t_i, and m_j with id_j, payload $da\ ta_j$, and timestamp t_j), then the distances between $da\ ta_i$ and $da\ ta_j$ and $t_j - t_i$ are, respectively, appended to D_{ij} and T_{ij} in $\Omega_T[id_i][id_j]$. For all legal CAN traces, Ω_T is traversed and filled in this manner.

Finally, after all CAN trajectories have been traversed, all D_{ij} in Ω_T keep only the maximum and minimum values of the list. Furthermore, for T_{ij} in Ω_T, the values in each list are classified according to the message time shift characteristics. Each discrete value is kept if it satisfies the discrete distribution. The maximum interval is kept if it satisfies the continuous or the completely random. The interval corresponding to the 3-sigma principle is kept if it satisfies the normal distribution. In particular, for nonexistent transfers, the corresponding elements of the matrix are two empty lists.

At this point, the matrix based on the CAN message and time transfer characteristics has been established, and it can describe the legal ID transfer, data transfer, and time transfer characteristics in automotive CAN networks.

Ω_T must be validated to improve the algorithm's effectiveness and minimize the false positive rate during detection. The model validation phase used Ω_T and the validation dataset as input, and the validation steps were as follows:

(1) For consecutive CAN messages in the validation dataset, check whether their ID transfer exists in the index of Ω_T. If it exists, proceed to the next step. If it does not exist, add the ID transfer as the index in Ω_T and go to the next step.

(2) Check the element indexed by the ID transfer in Ω_T. When the element is not empty, check whether the distance and time interval of consecutive CAN

Figure 12: Experimental environment for detection. (a) OBD-II splitter. (b) Detection setup.

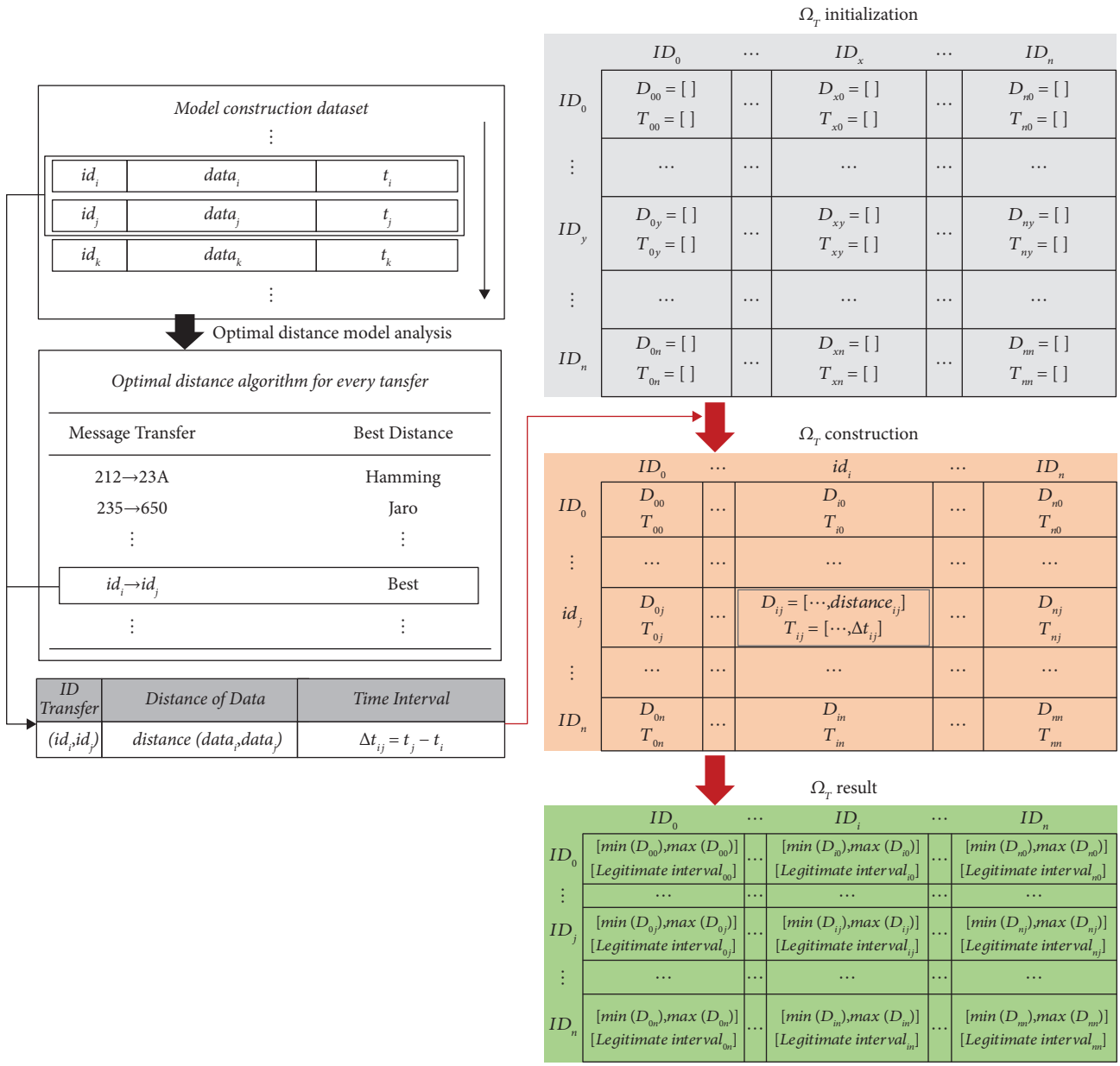


Figure 13: Model construction flowchart.

messages satisfy D_{ij} and T_{ij} and if they do, go to the next step; if they do not, update D_{ij} and T_{ij} with the current value. When the element is empty, fill the element with the distance and time interval distribution.

(3) Extract the next message transfer in the verification dataset and repeat the above steps.

4.4. Detection Scheme. The detection scheme used by the proposed algorithm utilizes CAN messages correlation analysis rather than simply comparing the transfer characteristics of adjacent messages. The algorithm uses the validated Ω_T to identify anomalies in the raw CAN bus data in the detection phase. Three consecutive messages (m_i, m_j, and m_k) are used as inputs to determine if m_j is an anomaly, where m_i is the reference message, m_j is the current message, and m_k is the following message. According to Figure 14, the steps for detection are as follows.

(1) First, the message transfer characteristics from m_i to m_j are checked. If the transfer satisfies the transfer characteristics in Ω_T, i.e., $\Omega_T[i][j]$ is not empty, and the distance and time interval of message transfer are legal values, the transfer from m_i to m_j exists and proceeds to the next step. If it fails to satisfy the transfer characteristics, m_j is identified as an abnormal message.

(2) Check whether the transfer from m_j to m_k satisfies the transfer characteristics of Ω_T. If it does, m_j is a standard message; otherwise, go to the next step.

(3) Determine if the transfer from m_i to m_k is legal. If the transfer exists, m_j is identified as an exception message; otherwise, m_j is legal.

Specifically, if m_j is identified as a legitimate message, the algorithm uses m_j as a reference to determine the legitimacy of m_k. When m_j is identified as an abnormal message, the algorithm still uses m_i as a reference and m_{k+1} as a subsequent message to check the legitimacy of m_k.

5. Performance Evaluation

This section details the application and performance evaluation of the proposed CAN intrusion detection scheme in test vehicles. Firstly, this section describes the dataset used for model construction, validation, and detection, together with the data collection and attack injection scenarios. Secondly, the performance of the proposed algorithm is evaluated in terms of recognition efficiency, time and resource consumption, and the algorithm's performance under different fault injection rates. Afterward, the algorithm's advantages in terms of accuracy are compared with other existing algorithms. Finally, specific applications of the algorithm and protection strategies combined with the central controller are described.

5.1. Datasets and Scenarios. To fully and effectively evaluate the performance of the proposed algorithm, we used datasets from different driving scenarios in the model construction, validation, and detection phases.

The dataset used for model construction contained CAN traffic collected while the test vehicle was driving on its daily commute route. The driving route of the test vehicle is shown in Figure 15(a), which included three driving scenarios: country roads, highways, and congested city roads. The dataset for model construction is shown in Table 1; it has 29213281 messages containing seven days of CAN traffic for commuter driving. Road conditions include slippery roads, congested roads, and rainy and foggy conditions. The CAN message and time transfer matrix Ω_T constructed using this dataset is a 39×39 square matrix that contains 780 legal transfers. The best distance model for the CAN message data fields in the test vehicle is publicly available, and the results indicate the best distance model for resisting fuzzy collisions.

The dataset used to validate the model is shown in Table 1, which is twice the size of the modeling dataset. This dataset was collected using the vehicle driving route, as shown in Figure 15(b). After the model was validated, it had zero false positives for the CAN bus traffic of the validation dataset, and the final validated matrix was the same as the original feature transfer matrix. This result confirms the existence of a fixed bus message transfer pattern under different driving habits and vehicle driving conditions.

To evaluate the detection efficiency of the proposed algorithm, the anomalous data were injected into the CAN bus of the test vehicle using data injection equipment based on four attack models in the vehicle stationary state and vehicle driving state, respectively. The attack dataset is shown in Table 2, in which 1003756 attack messages were injected when the vehicle was stationary and 1316894 attack messages were injected during driving. They all contained DoS attacks, fuzzy attacks, ulterior fuzzy attacks, and replay attacks. In addition, the driving route was set in a relatively safe country road to prevent the occupants of the vehicle from being harmed by any anomalies, as shown in Figure 15(c).

5.2. Identification Efficiency Evaluation. The performance of the proposed intrusion detection algorithm to identify anomalies was evaluated in terms of the accuracy of online detection of injected attack messages in actual vehicles. This section describes the accuracy of the algorithm in detecting attack messages in actual stationary and moving vehicles.

The experimental results show that the algorithm identifies all four types of CAN bus attacks well, as shown in Figure 16. The algorithm performs best in identifying DoS and fuzz attacks, maintaining almost 100% accuracy regardless of the vehicle state. The accuracy of identifying ulterior fuzz attacks was 95.7% when the vehicle was stationary, compared to 95.5% when the vehicle was running. The accuracy of identifying replay attacks was 90.6% when the vehicle was running and 93.6% when the vehicle was stationary. Overall, the anomaly detection accuracy when the vehicle was stationary was slightly higher than that of the vehicle in the running state. This result occurred because fewer types of messages appear

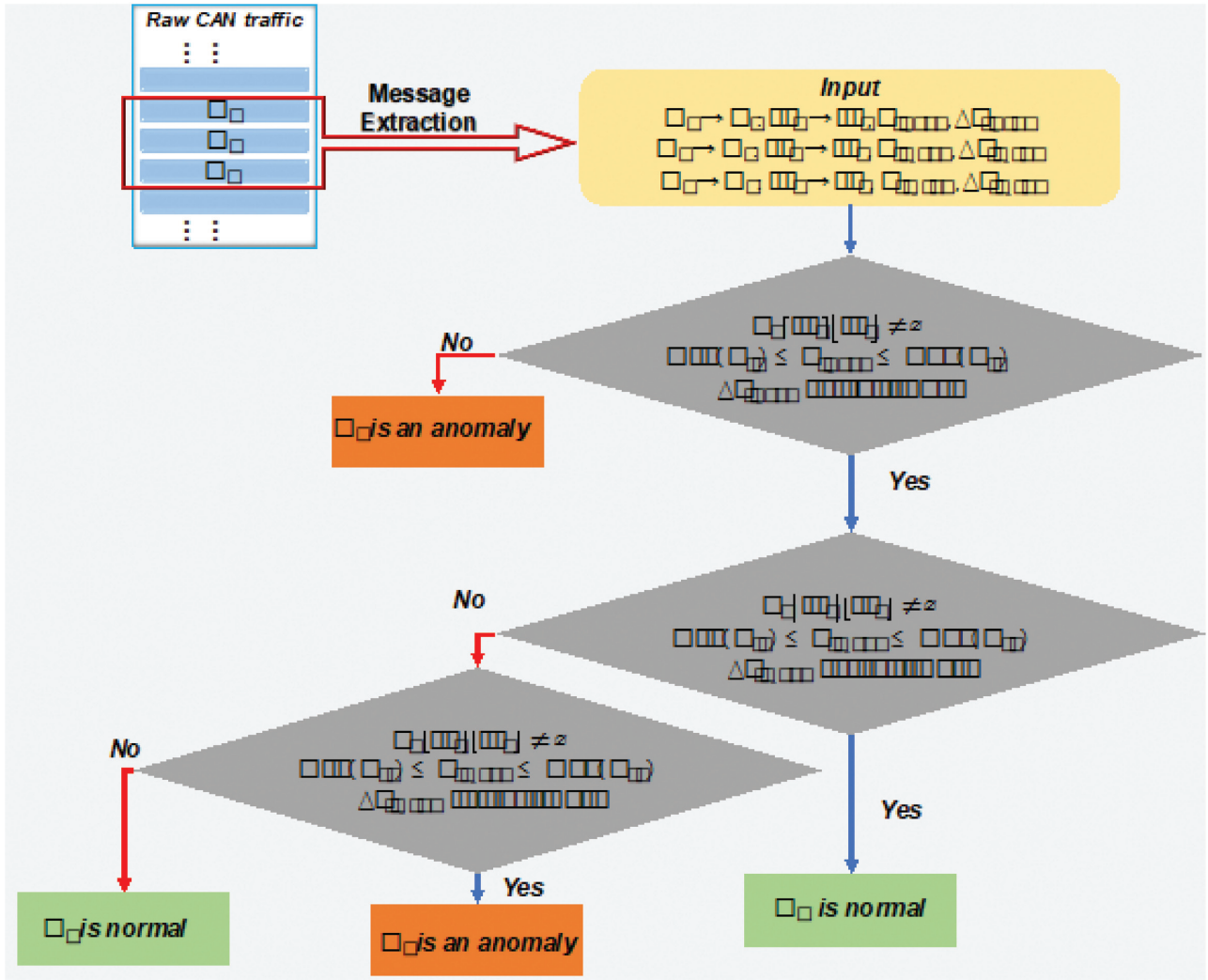


Figure 14: The diagram of detection stage.

(a) (b) (c)

Figure 15: Test vehicle travel routes. (a) Modeling dataset driving route. (b) Validation data set driving route. (c) Detection driving route.

when the vehicle is stationary, there are relatively few message transfers, and attack messages are not easily ignored. Furthermore, more messages appear in the CAN network when the vehicle is moving, and the probability of fuzzy attacks and replay attacks randomly colliding as legitimate transfers increases.

The performance analysis of the proposed algorithm is conducted as follows. When implementing DoS attacks, the

Table 1: Model construction and validation dataset.

Model construction dataset				Validation dataset	
No.	Message Quantity	Travel Time	Road Condit	Message Quantity	Travel Time
1	5379428	72 min	Jammed	10161141	136 min
2	5155282	69 min	Rainy	10086421	135 min
3	3362140	45 min	Smooth	9264563	124 min
4	3138009	42 min	Smooth	7770308	104 min
5	4781710	64 min	Jammed	6724279	90 min
6	3063285	41 min	Smooth	7247284	97 min
7	4333427	58 min	Foggy	7546140	101 min

Table 2: Attack dataset in detection.

Vehicle Status	Attack type	Attack messages
Stationary	DoS	595704
	Fuzzy	153947
	Ulterior fuzzy	153307
	Replay	100798
Driving	DoS	597031
	Fuzzy	239962
	Ulterior fuzzy	239950
	Replay	239951

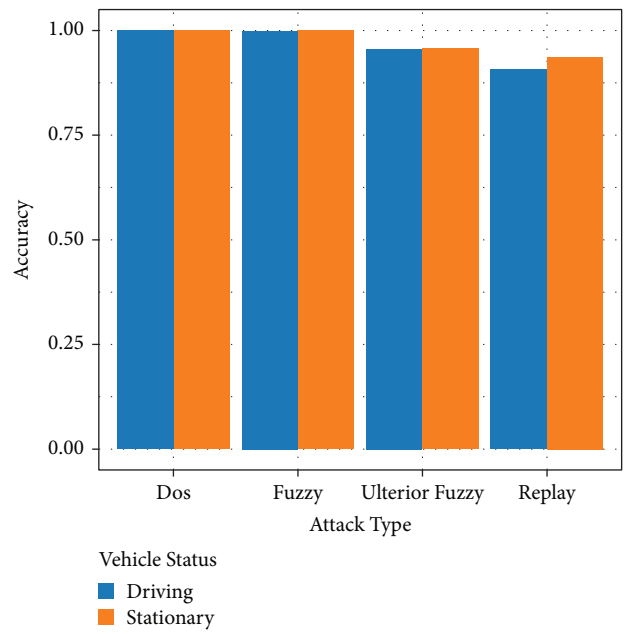


Figure 16: Accuracy of the proposed algorithm against different attacks.

attacker interferes with the CAN bus communication by sending messages with higher priority (i.e., message data with smaller IDs), thus occupying the total resources of the CAN bus. The proposed algorithm can quickly identify malicious messages using the presence or absence of ID transmissions alone, so the algorithm has a very high identification accuracy for DoS attacks. Fuzzy attacks inject messages randomly, implying that the message ID, data, and timestamp are random. When the algorithm detects an ambiguous attack, it checks whether the ID transfer, data distance, and time interval of three neighboring messages satisfy the legal conditions of the transfer matrix. Most fuzzy messages fail to satisfy these legal conditions, and very few fuzzy attacks that happen to satisfy the conditions can be misclassified as correct messages. Because the distance model, which works best for random data, is utilized in the modeling phase, the probability of such misclassification is minimal, causing the algorithm to have an accuracy rate close to 100% for fuzzy attacks. However, the ulterior fuzz attack is subtler than the fuzz attack, as it sends random messages with legitimate IDs. Therefore, ulterior fuzz attacks are more likely to be misclassified as legitimate by the algorithm. Although the ID transfer of messages is correct in detecting such attacks, the data distance and time interval between adjacent messages are random, so the proposed algorithm can still effectively identify ulterior fuzz attack messages. However, the identification accuracy is relatively lower than that for the first two attack messages. The replay attack is the most difficult to identify because it exactly replicates the previous correct message sequence and its ID and data fields are legitimate. The proposed algorithm identifies replay attacks by examining the characteristics of three consecutive messages. Although the ID transfer and data distance may be judged as correct, the replay timestamp is entirely random and judged as abnormal by the algorithm.

5.3. Time and Resource Effectiveness Assessment. Table 3 lists the time consumed by the algorithm for identifying the four types of attacks. The time required to identify standard information was the shortest, averaging 0.261 ms. In detecting attack messages, the time required to identify DoS attack information was the shortest, averaging 0.265 ms, and that required to identify replay attacks was the longest, averaging 0.284 ms. This result also validates the algorithm's model identification process, which uses only the ID transfer to identify DoS attacks and therefore takes the shortest time. Identifying the replay attack takes the longest time, as the three main features of ID transfer, data field distance, and time interval distribution are used. The total time to identify anomalies is the sum of the time interval of adjacent messages and the time required for algorithm identification. The time from the appearance of an anomaly message to its recognition is the sum of the time from that message to the appearance of the following message and the algorithm's recognition time. During normal CAN message acquisition,

Table 3: Identification times of different messages.

Message Type	Average Time (ms)
DOS message	0.265
Fuzz message	0.274
Ulterior fuzz message	0.271
Replay message	0.284
Normal message	0.261

the average time interval between adjacent messages is 0.991 ms. The average time required for the algorithm to identify anomalous messages is then between 1.256 milliseconds and 1.275 milliseconds. Overall, the algorithm achieved a high level of real-time performance in CAN bus anomaly detection, showing the ability to respond to anomalies as soon as they occur.

The proposed algorithm can meet the hardware resource constraints of automotive ECUs in terms of computational and storage resources. The execution of the algorithm in the detection phase is based on a simple lookup and comparison operation. Finding legitimate transfer features in the transfer matrix is a high-speed operation with a computational cost of O (1) that is not dependent on the number of IDs. In contrast, the memory requirements of the algorithm are related to the number of different IDs transferred in the vehicle CAN bus. To facilitate a fast search for information transmission characteristics, the algorithm uses a two-dimensional matrix with indexes for storage. In the experiments, there were 39 legitimate IDs and 780 legitimate transmissions in the test vehicle's CAN traces, and the transmission matrix's storage requirement was 20,064 bytes. When the algorithm detects the original CAN bus information, it must identify three adjacent messages with a storage requirement of 327 bytes. The total storage requirement of the algorithm is 20,391 bytes, which is much smaller than the hardware limitation of current automotive ECUs [30]. Therefore, the proposed algorithm can be applied to resource-constrained vehicles.

5.4. Performance under Different Injection Frequencies. To better evaluate the method's performance in identifying attack information, the accuracy and recall of the algorithm were tested at different attack injection frequencies. The performance of the proposed algorithm under different injection frequencies is shown in Figure 17, where four types of CAN bus attacks are injected at a frequency ranging from 10% to 100%. The percentage represents the ratio between the frequency of attack injections and actual messages. The detection of DoS attacks was independent of the injection frequency, and the accuracy and recall were always maintained at 1. Similarly, the identification of fuzz attacks was almost independent of the attack injection frequency, and the accuracy and recall remained close to 1. As the injection frequency increased, the recall of the proposed algorithm for detecting ulterior fuzz attacks approached 1, and the accuracy decreased but remained at 0.85 or more. As for detecting replay attacks, the recall of the proposed algorithm increased with the injection frequency, while the accuracy decreased.

DoS and fuzz attacks respectively rely on higher-priority IDs and completely random messages to cause CAN bus anomalies. The proposed algorithm can almost identify these anomalies using ID shifting and data distance, so the detection results are independent of the injection frequency. For the detection of ulterior fuzz attacks, the judgment using the ID transfer is no longer valid. Whether the messages are legal must be checked based on the dimensions of data distance and time interval, so the detection result is affected by the injection frequency to a certain extent. The increase in injection frequency causes an increase in attack messages, resulting in a certain number of missed and false positives and decreasing accuracy and precision. However, with the increase in injection speed, anomalies are more easily detected with time, and the recall is subsequently increased. The detection of replay attacks by the proposed algorithm almost depends on the time interval between messages, so it is affected by the injection frequency. The faster the replay attacks are injected, the easier the algorithm is to identify. In this case, the number of missed anomalies is reduced, and recall and accuracy are subsequently improved.

5.5. Results Compared to Other Available Methods. This section compares the proposed intrusion detection methods with schemes applied to onboard ECUs. The currently available intrusion detection schemes are the ID sequence-based approach proposed by M. Marchetti and D. Stabili, the Hamming distance-based scheme, and the message-cycle-based method proposed by H. M. Song. These schemes can be applied to ECUs due to their lightweight design, but their accuracy in identifying anomalous messages needs improvement.

Table 4 indicates that both the sequence-based method and the proposed method can identify DoS attack messages very well. In contrast, the other two methods cannot identify DoS attacks because they do not use ID features. These four methods maintain an accuracy above 0.99 for identifying fuzz attacks. For identifying ulterior fuzz messages, the algorithm proposed in this paper achieved the best performance with an accuracy of 0.957, followed by the Hamming distance-based method, message-cycle-based method, and sequence-based method. For identifying replay attacks, the proposed method achieved an accuracy of 0.936. Except for the proposed method, the Hamming distance-based method achieved the best performance with an accuracy of 0.878, but its recall indicated that the method had missed several messages.

5.6. Applications and Discussion. The experimental results verify that the proposed method achieved excellent performance in detecting CAN bus attack messages with high accuracy for DoS attacks, fuzz attacks, advanced ulterior fuzz attacks, and replay attacks. Moreover, the proposed algorithm achieves outstanding performance under different attack injection frequencies. DoS attacks were injected at the highest possible frequency to suspend CAN bus

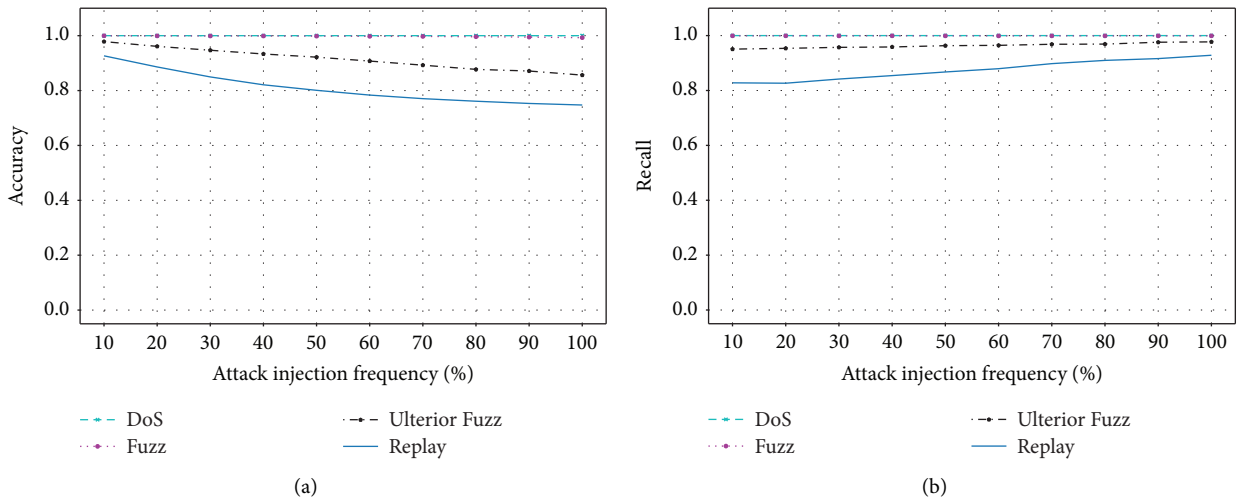


Figure 17: Performance at different injection frequency. (a) Accuracy under different injection frequency. (b) Recall under different injection frequency.

Table 4: Comparison with other applicable algorithms in terms of accuracy and recall.

Scheme	Indicator	DoS	Fuzz (%)	Ulterior fuzz (%)	Replay (%)
Message and time transfer matrix	Accuracy	100%	99.87	95.74	93.63
	Recall	100%	99.89	94.34	84.67
ID sequence-based	Accuracy	100%	99.64	85.09	79.65
	Recall	100%	98.8	44.64	17.08
Hamming distance-based	Accuracy	—	99.6	88.66	87.81
	Recall	—	98.25	98.13	6.8
Message-cycle-based	Accuracy	—	99.21	87.56	87.05
	Recall	—	31.09	68.1	47.49

communications in an actual attack. The proposed algorithm's excellent performance in detecting DoS attacks allows it to effectively identify the attack messages and implement protection along with the corresponding ECU. Fuzz attacks are usually sent randomly with entirely random content, and the proposed algorithm can maintain an accuracy exceeding 95% for anomaly identifications regardless of the injection frequency. In practical tests, it was found that the faster the replay attacks were injected, the better the algorithm recognized them. The proposed method outperformed all other methods in identifying anomalous messages. Additionally, it effectively detected four common CAN bus attacks and exhibited a convincing ability to handle all the attacks under different injection frequencies, making it suitable for practical application scenarios. If the algorithm is applied to an ECU and combined with the CAN bus protection function, it will be able to identify CAN attacks quickly to protect the vehicle effectively.

In addition, the vehicle CAN bus intrusion detection scheme proposed herein can be applied in the typical CAN network architecture mentioned in Chapter 2. For intradomain communication in CAN networks, the scheme can be deployed in any intradomain ECU to achieve intradomain anomaly detection and instruct the attacked ECU to respond. For cross-domain communication in the CAN network, the solution can be deployed in the central controller to detect abnormal CAN messages in cross-domain communication traffic in time to block them and notify the driver to check the vehicle. Primarily, when numerous replay messages and high-frequency messages appear, timely blocking is performed to prevent the car from being maliciously manipulated.

6. Message and Time Transfer Matrix as CAN Bus Data Transmission Fingerprinting

In this section, the features in the Ω_T and the features of the four attack models are analyzed. From the results, significant differences between the features of Ω_T and those of the present attacks are revealed. Therefore, the message and time transfer matrix are exploited as a fingerprint to quickly distinguish the legitimate transfer matrix from the anomalous transfer matrix.

6.1. Comparison of Legal Transfer Characteristics and Abnormal Characteristics. As shown in Figure 18, there was a clear difference between the legitimate Ω_T and the transfer matrix when a DoS attack occurred. Based on this difference, it is possible to quickly identify whether the CAN bus is under a DoS attack. The difference between the message distance of the legitimate Ω_T and that of the message transfer

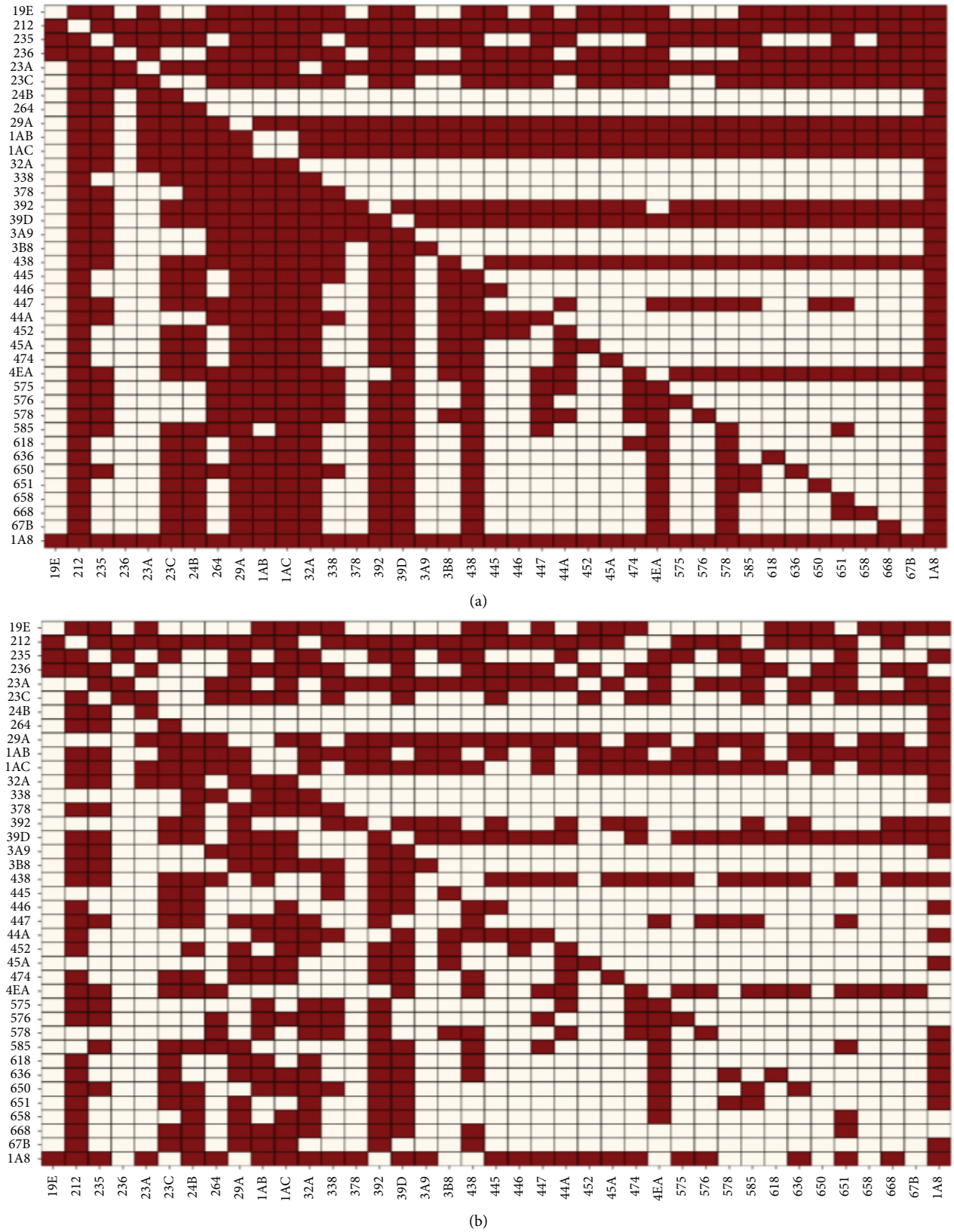


Figure 18: Legitimate ID transfer vs. ID transfer in the presence of DoS attack. (a) Legitimate ID transfer matrix. (b) ID transfer matrix for DoS attack occurred.

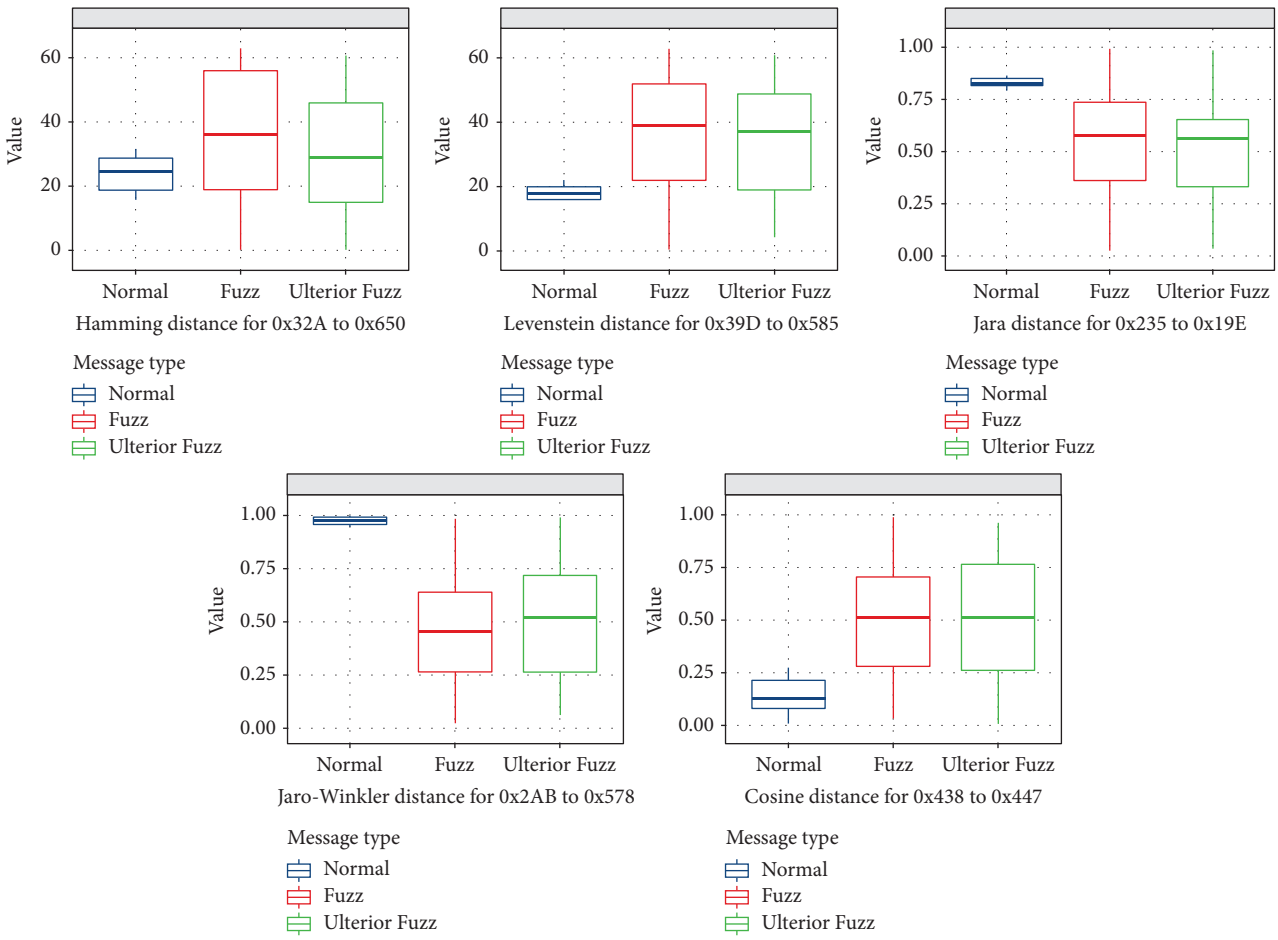


Figure 19: Distance distribution of normal and abnormal packet transfers.

under fuzzy attack and ulterior fuzz attack is shown in Figure 19. When an attack occurs, the message distance fluctuates in a much larger range than that of the legitimate transfer and takes different values. The time intervals of the legitimate message transfer can be classified as discrete and continuous intervals, respectively exhibiting a normal distribution and random value distribution, as shown in Figure 9. However, the time intervals under an attack are entirely random. Thus, the anomalies can be distinguished by the characteristics of the transfer matrix.

6.2. Legal Message and Time Transfer Matrix as CAN Message Transfer Fingerprint. Based on the results of this study, the legal Ω_T was used as a fingerprint for CAN message transfers to quickly identify abnormal messages in the automotive CAN bus and assist the ECU in ensuring vehicle security and further safeguarding the safety of drivers and pedestrians. Moreover, this fingerprint can help the government and enterprises monitor the illegal modification of vehicles and other unlawful behaviors.

7. Conclusion

In this study, we have proposed a CAN bus anomaly detection algorithm based on CAN message transmission characteristics applied to real vehicle applications. We found that a message and time transmission matrix for attack detection can be constructed using the characteristics of ID transmission, data distance, and time interval. Moreover, this message and time transmission matrix combined with message correlation analysis can effectively identify DoS attacks, fuzzy attacks, and the more challenging collapse attacks and replay attacks. Additionally, the efficiency of the proposed algorithm in identifying these four types of attacks is not affected by the frequency of attack injection. Furthermore, the proposed algorithm has low computational and storage resource consumption, facilitating its application to vehicles without significant hardware resources. In addition, compared with three other lightweight CAN bus intrusion detection methods, the proposed method can effectively identify four types of attacks and achieve a high identification accuracy and recall rate. In practical vehicle applications, the proposed method can be deployed in any ECU within a functional area to identify intradomain attack messages, thanks to its low resource consumption. Alternatively, it can be deployed in the central controller to detect cross-domain attacks.

The excellent attack discovery capability of the proposed algorithm can be combined with the security protection function of the ECU to filter and locate the attack information, which helps to rank the threat nodes in the vehicle

CAN bus network. Future research will focus on more lightweight algorithms and finer-grained feature processing. Meanwhile, a CAN intrusion detection system will be built by combining bus traffic and physical bus features.

Data Availability

The best distance model for every transfer can be found at http://49.232.218.41:8000/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was supported by the China Postdoctoral Science Foundation under grant of no. 2021T140074, National Key Research and Development Program of China (2020YFB1707700), China Postdoctoral Science Foundation (2019M652475), the National Natural Science Foundation of China under grant no. 62102042, and the Data Security Risk Monitoring Traceability and Integrated Management Platform project from the 2020 China Industrial Internet Innovation and Development Project.

References

[1] J. E. Siegel, D. C. Erb, and S. E. Sarma, "A survey of the connected vehicle landscape-architectures, enabling technologies, applications, and development areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2391–2406, 2018.

[2] K. Matheus and T. Königseder, *Automotive Ethernet*, Cambridge University Press, 2021.

[3] S. Checkoway, D. McCoy, B. Kantor et al., "Comprehensive experimental analyses of automotive attack surfaces," in *Proceedings of the USENIX Security Symposium*, vol. 4, pp. 447–462, San Francisco, August 2011.

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

[5] Keen Security Lab, "Keen security lab blog," 2016, https://keenlab.tencent.com/.

[6] A. Greenberg, "Hackers remotely kill a Jeep on the highway—with me in it," 2015, https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[7] A. Shahani, "Tesla model S can Be hacked, and fixed," 2015, https://www.npr.org/sections/alltechconsidered/2015/08/06/429907506/tesla-model-s-can-be-hacked-and-fixed-which-is-the-real-news.

[8] Tencent Keen Security Lab, "Tencent keen security lab: experimental security research of Tesla autopilotd," 2019, https://keenlab.tencent.com/en/2019/03/29/Tencent-Keen-Security-Lab-Experimental-Security-Research-of-\Tesla-Autopilot/.

[9] R.-P. Weinmann and B. Schmotzle, "Tbone – a zero-click exploit for Tesla MCUs," 2020, https://kunnamon.io/tbone/tbone-v1.0-redacted.pdf.

[10] C. Young, J. Svoboda, and J. Zambreno, "Towards reverse engineering controller area network messages using machine learning," in *Proceedings of the 2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, pp. 1–6, IEEE, New Orleans, LA, USA, June 2020.

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

[12] B. Blaauwendraad and V. Kieberl, "Automated reverse-engineering of can messages using obd-ii and correlation coefficients," https://www.os3.nl/_media/2019-2020/courses/rp2/p103_report.pdf.

[13] K. T. Cho and G. S. Kang, "Viden: attacker identification on in-vehicle networks," in *Proceedings of the 2017 ACM SIGSAC Conference*, Texas, Dallas, USA, October 2017.

[14] M. Tian, R. Jiang, H. Qu, Q. Lu, and X. Zhou, "Advanced temperature-varied ecu fingerprints for source identification and intrusion detection in controller area networks," *Security and Communication Networks*, vol. 2020, no. 4, pp. 1–17, 2020.

[15] 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 *Proceedings of the International Conference on Information Networking*, Kota Kinabalu, Malaysia, January 2016.

[16] D. Stabili, M. Marchetti, and M. Colajanni, "Detecting attacks to internal vehicle networks through hamming distance," in *Proceedings of the Aeit International Conference*, pp. 1–6, Cagliari, Italy, September 2017.

[17] M. Marchetti and D. Stabili, "Anomaly detection of can bus messages through analysis of id sequences," in *Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, CA, USA, June 2017.

[18] H. Lee, S. H. Jeong, and H. K. Kim, "Otids: a novel intrusion detection system for in-vehicle network by using remote frame," in *Proceedings of the 2017 15th Annual Conference on Privacy, Security and Trust (PST)*, Calgary, AB, Canada, August 2018.

[19] K.-S. Yu, S. H. Kim, D. W. Lim, and Y. S. Kim, "A multiple Rényi entropy based intrusion detection system for connected vehicles," *Entropy*, vol. 22, no. 2, p. 186, 2020.

[20] W. Wu, Y. Huang, R. Kurachi et al., "Sliding window optimized information entropy analysis method for intrusion detection on in-vehicle networks," *IEEE Access*, vol. 6, pp. 45233–45245, 2018.

[21] S. Tariq, S. Lee, H. K. Kim, and S. S. Woo, "Can-adf: the controller area network attack detection framework," *Computers & Security*, vol. 94, Article ID 101857, 2020.

[22] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, no. Jan., pp. 100198.1–100198.13, 2020.

[23] E. Seo, H. M. Song, and H. K. Kim, "Gids: Gan based intrusion detection system for in-vehicle network," in *Proceedings of the 2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pp. 1–6, Belfast, Ireland, August 2018.

[24] C. Wang, Z. Zhao, L. Gong, L. Zhu, Z. Liu, and X. Cheng, "A distributed anomaly detection system for in-vehicle network using htm," *IEEE Access*, vol. 6, pp. 9091–9098, 2018.

[25] B. Waggener, W. N. Waggener, and W. M. Waggener, *Pulse code modulation techniques*, Springer Science & Business Media, United States, 1995.

[26] G. Navarro, "A guided tour to approximate string matching," *ACM Computing Surveys*, vol. 33, no. 1, pp. 31–88, 2001.

[27] S. Amit, "Modern information retrieval: a brief overview," *IEEE Data Eng. Bull.*vol. 24, no. 4, pp. 35–43, 2001.

[28] W. E. Winkler, "Overview of record linkage and current research directions," in *Bureau of the Census*Citeseer, 2006.

[29] W. E. Winkler, "String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage," 1990, https://files.eric.ed.gov/fulltext/ED325505.pdf.

[30] NXP, "S12XS automotive and industrial microcontrollers," 2021, https://www.nxp.com/products/processors-and-microcontrollers/additional-mpu-mcus-architectures/16-bit-s12-and-s12x-mcus/s12xs-automotive-and-industrial-microcontrollers-mcus:S12XS.