Electronic Preprint for Journal of Information Processing Vol.26

Regular Paper

Supervised and Unsupervised Intrusion Detection Based on CAN Message Frequencies for In-vehicle Network

Takuya Kuwahara[1,a)] Yukino Baba[1,b)] Hisashi Kashima[1,c)]
Takeshi Kishikawa[2,d)] Junichi Tsurumi[2,e)] Tomoyuki Haga[2,f)]
Yoshihiro Ujiie[2,g)] Takamitsu Sasaki[2,h)] Hideki Matsushima[2,i)]

Received: September 2, 2017, Accepted: December 8, 2017

Abstract: Modern vehicles are equipped with Electronic Control Units (ECUs) and external communication devices. The Controller Area Network (CAN), a widely used communication protocol for ECUs, does not have a security mechanism to detect improper packets; if attackers exploit the vulnerability of an ECU and manage to inject a malicious message, they are able to control other ECUs to cause improper operation of the vehicle. With the increasing popularity of connected cars, it has become an urgent matter to protect in-vehicle networks against security threats. In this paper, we study the applicability of statistical anomaly detection methods for identifying malicious CAN messages in in-vehicle networks. We focus on intrusion attacks of malicious messages. Because the occurrence of an intrusion attack certainly influences the message traffic, we focus on the number of messages observed in a fixed time window to detect intrusion attacks. We formalize features to represent a message sequence that incorporates the number of messages associated with each receiver ID. We collected CAN message data from an actual vehicle and conducted a quantitative analysis of the methods and the features in practical situations. The results of our experiments demonstrated our proposed methods provide fast and accurate detection in various cases.

Keywords: anomaly detection, intrusion detection, in-vehicle network

1. Introduction

Modern vehicles are equipped with multiple Electronic Control Units (ECUs) that control the main operations of vehicular components, such as the engine or the brakes. There are about 100 ECUs in a present-day vehicle, each of which communicates with the others over a bus-topology in-vehicle network. The vehicles are also equipped with external communication devices that allow them to update the firmware and interact with mobile devices such as the drivers' smartphones. These communication devices act as an interface between the in-vehicle network and the out-vehicle network.

Although the external network connectivity offers various benefits, it also increases the security risk within the in-vehicle network. The Controller Area Network (CAN), a widely used communication protocol for ECUs, does not have a security mechanism to detect improper packets. If attackers exploit the vulnerability of an ECU and manage to inject a malicious message, they will be able to control other ECUs to cause incorrect operations within the vehicle. A CAN bus network and an attack on it is illustrated in **Fig. 1**. In 2010, Koscher et al. reported the ability of malicious CAN messages to adversarially control several operations, such as unlocking the doors and stopping the engine. Another experiment demonstrated that it was possible to override the operations of the steering wheel and the brakes [12]. In 2015, it was demonstrated that one could identify the software vulnerability of a Jeep Cherokee and wirelessly hack the operations of the vehicle through a cellular network [*1]. After this revelation, Chrysler announced that it would recall 1.4 million vehicles [*2].

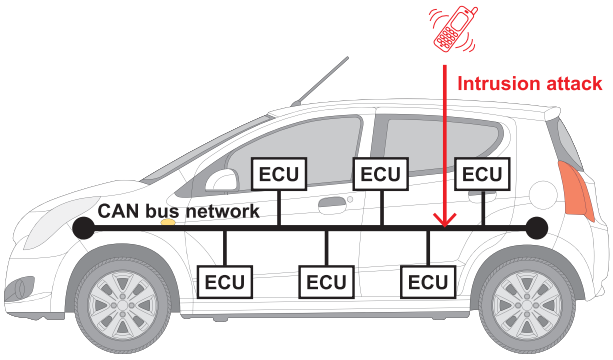


Fig. 1 A CAN bus network and an intrusion attack on it through out-vehicle networks.

[1] Kyoto University, Kyoto 606–8501, Japan
[2] Panasonic Corporation, Kadoma, Osaka 571–8501, Japan
[a)] tkuwahara@ml.ist.i.kyoto-u.ac.jp
[b)] baba@i.kyoto-u.ac.jp
[c)] kashima@i.kyoto-u.ac.jp
[d)] kishikawa.takeshi@jp.panasonic.com
[e)] tsurumi.junichi@jp.panasonic.com
[f)] haga.tomoyuki@jp.panasonic.com
[g)] ujiie.yoshihiro@jp.panasonic.com
[h)] sasaki.takamitsu@jp.panasonic.com
[i)] matsushima.hideki@jp.panasonic.com

[*1] https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/
[*2] https://www.wired.com/2015/07/jeep-hack-chrysler-recalls-1-4m-vehicles-bug-fix/

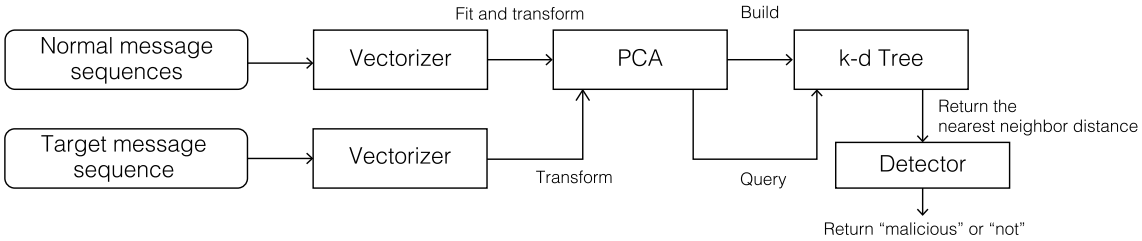


Fig. 3 The proposed pipeline for detecting malicious message sequences by using ID-Counting features.

With the emergence of telematics services and intelligent navigation systems, and the increasing popularity of connected cars, it has become an urgent matter to protect in-vehicle networks against security threats.

In this paper, we study the applicability of statistical anomaly detection methods to identify malicious CAN messages in in-vehicle networks. We focus on intrusion attacks of malicious messages, which is one of several standard attacks wherein an attacker need not investigate the message patterns on the CAN bus network nor conduct a reverse-engineering investigation of vehicle functions. We define a malicious message as a message which is sent by an attacker for causing incorrect operations within the vehicle.

In particular, we aim to detect malicious message sequences, each of which has at least one malicious message in a fixed time window. Owing to the fact that an intrusion attack invariably influences message traffic, we focus on the message frequencies to detect the attack. We assume that a sequence is more likely to contain a malicious message if the number of messages in the sequence is higher. To implement this concept, we design the *Total-Counting feature* to represent the characteristic of a sequence. The Total-Counting feature is a count of the number of messages in a sequence, in other words, the inversed message frequency. We assume that the number of messages in a sequence follows a Gaussian distribution, and we use a statistical testing method to detect whether the sequence is contaminated or not.

Next, we consider that the ID of each message is valuable for discovering a malicious message sequence. A CAN message is associated with the ID field, which determines the priority or the receiver of the message, and we introduce the *ID-Counting feature* by using this ID information. The ID-Counting feature is a vector, each of whose elements is the number of messages associated with each ID. Given a dataset containing only normal messages, we generate the feature for each normal sequence and calculate the distance between the vector corresponding to a target sequence and that corresponding to its nearest neighbor in the normal messages. We assume that the distance follows a normal distribution, and we use a statistical testing method to detect whether the sequence is contaminated or not.

There is a huge demand for real-time intrusion detection in vehicle networks because malicious messages may cause unexpected improper operations of the target vehicle, and therefore, we need to take actions such as stopping the vehicle or sending a warning to the driver as soon as possible. Keeping in perspective such requirements, we propose a pipeline to speed up the detection process by applying a k-d Tree and a dimension reduction method. The proposed two features are illustrated in **Fig. 2**, and the pipeline is shown in **Fig. 3**.

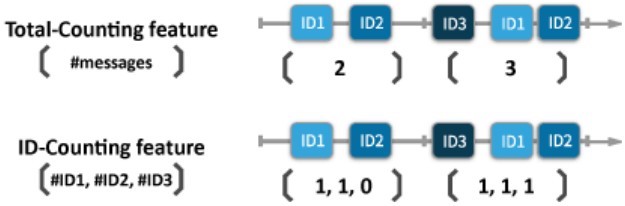


Fig. 2 The proposed message sequence features: Total-Counting feature and ID-Counting feature.

All of the aforementioned approaches are designed for unsupervised cases, where we do not have any examples of malicious messages. However, when we have cases involving malicious messages, we can apply the two features to build a classifier to detect the attack by using supervised learning methods.

We collected CAN message data from an actual vehicle and simulated four attack scenarios by injecting fake CAN messages. We conducted experiments by using two types of malicious messages: one is randomly injected to the network, and the other is mimicking the transmission cycles of the normal messages. By applying our methods to each scenario, we observed that both the Total-Counting and the ID-Counting features accurately detected malicious message sequences and achieved an AUC of over 0.999 in the cases where a large number of malicious messages were injected (i.e., over 50% of the total messages were malicious). The results of the experiments also showed that the ID-Counting feature with supervised learning achieved high detection performance with an AUC of more than 0.95 even when the number of malicious messages was small (i.e., less than 1% of messages were malicious). We further confirmed that our methods take less than 3 ms for detecting whether a message sequence is malicious in almost all the cases.

Several statistical approaches have been proposed for detecting malicious CAN messages. Although many methods used payload (i.e., contents) information of each message [5], [10], [11], [13], [16], there has been another line of research aiming to detect malicious messages by only using the timestamp and ID of each message [2], [4], [15]. Our work contributes to this line by proposing methods which focus on the nearest neighbor distance between the target sequence and the normal sequences.

Our key contributions are summarized as follows:

- We formalize two basic features to represent a CAN message sequence. These features can be extracted from the timestamp and the ID information in each message.
- We propose a pipeline for quickly detecting malicious messages in the in-vehicle network.
- We conducted a quantitative analysis of the methods and the

features in practical situations by using actual CAN message datasets and in supervised and unsupervised cases, and demonstrate that our proposed methods provide fast and accurate detection in various cases.

2. Intrusion Detection in In-vehicle Networks

2.1 Controller Area Network (CAN)

The CAN protocol was developed by Robert Bosch GmbH. It is used not only in in-vehicle networks, but also in applications such as the controlling of industrial machines and so on owing to its excellent noise resistance. ECUs are interconnected in the CAN bus network. Each ECU is assigned a specific CAN ID in advance, which is regarded as the receiver, and it determines whether a data frame is received by the specified CAN ID. The decision whether a data frame is received or not is realized through a dedicated hardware called the CAN controller.

2.2 Possible Attacks on CAN Networks

CAN has a broadcasting system whereby all the ECUs can receive any message because CAN uses the bus network. In other words, there are vulnerabilities that we can easily exploit by intercepting and exchanging messages and sending the modified messages to any ECU in the network. The attackers in the abovementioned cases [8], [12] accessed the CAN network through an On-Board Diagnostics II (OBD-II) port, which is available in most standard cars, and sent malicious messages to the vehicle. In the Jeep's case, a cellular network was used to intrude into the in-vehicle network. Following the literature, we focus on intrusion attacks on CAN networks through out-vehicle networks.

2.3 Intrusion Detection Problem

In this study, we aim to detect a *malicious message sequence*, which includes at least one malicious message. When we detect a malicious message sequence, we can take several actions such as stopping the vehicle or sending a warning to the driver. We start with a formulation of the CAN data. Let us assume there are T messages and each message is associated with its ID. By regarding the CAN data as a time series data, we can formulate the CAN data $\boldsymbol{D}$ as $\boldsymbol{D} = (m^{(1)}, \cdots, m^{(T)})$, where the messages are ordered according to the received time. We assume that we are given the ID of the i-th message $m^{(i)}_{\mathrm{id}} \in \mathcal{I}$, where $\mathcal{I} = \{\mathrm{id}1, \mathrm{id}2\ \ldots, \mathrm{id}l\}$ is a set of CAN IDs.

We then assume that the messages are divided into W message sequences by using the width of time window w. The message sequences are represented as $\boldsymbol{S} = (\boldsymbol{s}^{(1)}, \cdots, \boldsymbol{s}^{(W)})$, where $\boldsymbol{s}^{(j)}$ is the j-th sequence of messages. Our goal is to detect whether each message sequence $\boldsymbol{s}^{(j)}$ is a malicious one or not.

3. Proposed Methods

3.1 Analysis of the Normal Behaviors of CAN Messages

We first analyze the typical normal behaviors of CAN messages in terms of their message frequencies. **Figure 4** illustrates the intervals of arrival times for a particular message ID. The figure shows the messages with this particular ID arrive with a quite stable frequency. We found that many other message IDs also had such stable frequencies, and, therefore, we hypothesize that injections of malicious messages cause temporal changes in the message frequencies, which makes it possible to detect them by finding changes in the message frequencies.

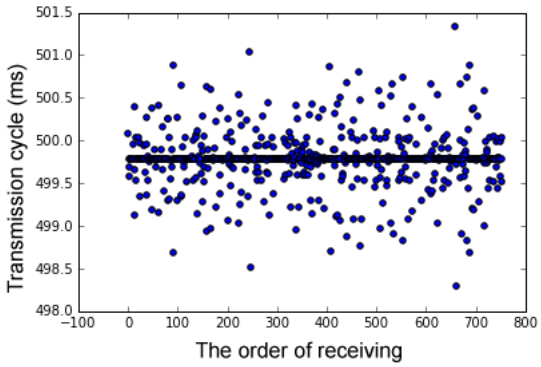


Fig. 4 Analysis of the transmission cycle; the intervals of arrival times for a particular message ID are shown. The vertical axis and the horizontal axis correspond to the time differences between consecutive messages and the order of the arrivals, respectively. The messages with this particular ID arrive with a quite stable frequency.

3.2 Unsupervised Detection Using the Total-Counting Feature

The observation in Section 3.1 suggests that the number of messages in a sequence can be used as a clue to detect a malicious message sequence. Thus, we design the *Total-Counting feature* to represent a message sequence. The Total-Counting feature of the j-th message sequence, $x^{(j)}_{\mathrm{tc}}$, is given as $x^{(j)}_{\mathrm{tc}} = n_j$, where n_j indicates the number of messages in the j-th sequence.

We then introduce a detection algorithm based on a statistical test. We assume that each $x^{(j)}_{\mathrm{tc}}$ follows the normal distribution $\mathcal{N}(\mu_{\mathrm{tc}}, \sigma^2_{\mathrm{tc}})$. The mean μ_{tc} and the variance σ^2_{tc} of normal distribution are estimated by the maximum likelihood estimation. Note that we build the model by using a dataset that possibly contains malicious messages because the effect of these messages on estimating the parameters can be limited.

The obtained normal distribution $\mathcal{N}(\hat{\mu}_{\mathrm{tc}}, \hat{\sigma}^2_{\mathrm{tc}})$ is a model representing the probability of the number of messages in a message sequence. To detect whether a message sequence is malicious, we perform a statistical test and obtain the p-value by using the model. If the value is less than a predefined threshold, we determine that the message sequence is a malicious one.

3.3 Unsupervised Detection Using the ID-Counting Feature

The Total-Counting feature just counts the number of messages in each sequence and the information of the CAN ID is not taken into account. We next present another feature, called the *ID-Counting feature*, that is constructed from the number of messages associated with each ID. The ID-Counting feature of the j-th message sequence, $\boldsymbol{x}^{(j)}_{\mathrm{ic}}$, is defined as $\boldsymbol{x}^{(j)}_{\mathrm{ic}} = \left(n_{j1}, n_{j2}, \ldots, n_{jl}\right)$, where n_{jk} is the number of messages associated with id$k$ in the j-th sequence.

Whereas the Total-Counting feature is represented by a scalar value, the ID-Counting feature is a multi-dimensional vector. Therefore, we apply a different detection approach for the ID-Counting feature. Instead of considering the feature value itself, we assume that the nearest neighbor distance between a message

Table 1 Statistics about the datasets used in the experiments.

Data	No. of messages	No. of malicious messages	Time length
SMALL-PARKED	62,015	144	60.9 s
LARGE-PARKED	64,518	2,905	60.6 s
SMALL-MOVING	49,625	819	46.3 s
LARGE-MOVING	218,024	118,152	95.7 s

sequence and *normal* message sequences follow a normal distribution. Let us assume that we have a set of ID-Counting features of the normal message sequences, $\mathcal{X}_N$. Our algorithm first calculates the nearest neighbor distance between the ID-Counting feature of each message sequence $x_{\mathrm{ic}}^{(j)}$ and $\mathcal{X}_N$, which is denoted by d_j, and obtains the maximum likelihood estimate of the mean and the variance of the normal distribution, namely, $\hat{\mu}_{\mathrm{ic}}$ and $\hat{\sigma}_{\mathrm{ic}}^2$. To detect whether a message sequence is malicious, we calculate the nearest neighbor distance between the sequence and $\mathcal{X}_N$, and perform a statistical test and compute the p-value by using the model. If the value is below a predefined threshold, the message sequence is detected as malicious.

Since we focus on an intrusion detection in in-vehicle networks, there is a demand for detecting a malicious message sequence as fast as we can so that we can stop the vehicle or send a warning to the driver before an accident occurs. For implementing a real-time detection, we propose a pipeline combining a k-d Tree data structure and a dimension reduction method for efficiently calculate the nearest neighbor distances to detect whether a message sequence is malicious. k-d Tree is a data structure having an average computational complexity $O(\log|\mathcal{S}|)$ for finding the nearest neighbor in a vector set $\mathcal{S}$ [1], [6]. This data structure allows us to calculate the nearest neighbor distance efficiently even when we have a large number of normal messages. The ID-Counting feature of a message sequence is an l-dimensional vector, and l (i.e., the number of unique CAN IDs) is typically around 100; however, a k-d Tree is not suitable for such high-dimensional vectors. We thus apply principal component analysis (PCA) to reduce the dimension of the feature vectors for speeding up the nearest neighbor calculation.

4. Experiments

4.1 Setting

To evaluate the accuracies of our methods, we prepared actual CAN message datasets and compared the detection accuracies of the two features. The accuracies were evaluated using the area under the ROC curve (AUC). Although we introduced unsupervised detection methods in Section 3, when we are given the ground truth of malicious and normal message sequences, we can build a detector in a supervised manner. We applied L2 regularized logistic regression to learn the detectors and investigated their performance as well as the performance of the unsupervised detectors.

4.2 Datasets

We collected CAN messages sent into the in-vehicle network of an actual automobile. We generated datasets with malicious messages by simulating attack scenarios. We prepared four datasets called SMALL-PARKED, LARGE-PARKED, SMALL-MOVING, and LARGE-MOVING. The datasets differed in the amount of malicious messages (SMALL or LARGE) and in the status of the vehicle (PARKED or MOVING). A summary of the datasets is described in **Table 1**. These scenarios were simulated in the following ways:

- LARGE-PARKED and LARGE-MOVING: A large number of malicious messages were randomly injected into the network. These datasets simulate DoS attacks, which attempt to flood the target network and to prevent other ECUs from sending normal messages. Specifically, we set several attack sequences and inserted a large number of malicious messages to each sequence. In both datasets, the numbers of the inserted messages per unit time are almost the same; however, the number of the attach sequences is smaller in the LARGE-PARKED dataset then in the LARGE-MOVING dataset, and thus the total number of malicious messages is smaller in the LARGE-PARKED dataset.
- SMALL-PARKED and SMALL-MOVING: A small number of malicious messages were injected. Each malicious message was injected right after a normal message having the selected IDs. We assumed that attackers knew the cycle of normal messages [*3] and that they managed to inject malicious messages in accordance with this cycle.

We had 97 IDs in our dataset and three of them were selected for sending malicious messages. The three IDs were related to the operations of the steering wheel, and, therefore, we simulated the attacks wherein the vehicle could be steered to an unexpected direction. We prepared another dataset for each attack dataset for applying the supervised learning method. We prepared a normal dataset as well, which contained about 380,000 messages recorded while the vehicle was parked for 6 min. This normal dataset was used for our unsupervised method with the ID-Counting feature.

4.3 Parameters

We varied the width of the time window among values of 10, 20, 50, 100, and 200 ms in the experiments. It would be more beneficial in a practical situation if our methods could detect a malicious sequence of a shorter time width to deal with the attacks immediately. When we apply the ID-Counting features with PCA, we need to establish the dimension size. Our initial experiment with varying dimension size from 1 to 18 showed that the performance improvement tended to saturate when the dimension size was about 10 or more. As a large dimension size causes more computational costs, we found that setting the dimension size to 10 was a reasonable choice and we fixed the dimension size to 10 in the experiments.

4.4 Results

Table 2 shows the AUCs of the unsupervised learning method

[*3] This can be done by plugging an OBD-II scanner tool to the target vehicle, and analyzing CAN message logs.

Table 2 AUCs of the unsupervised learning method with the Total-Counting feature and the ID-Counting feature for the four datasets. The ID-Counting feature showed better performance than the Total-Counting feature when the time width was small.

Time width	Large-Parked		Large-Moving		Small-Parked		Small-Moving	
	Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting
10 ms	0.996	0.987	0.999	1.000	0.602	0.991	0.634	0.707
20 ms	0.999	0.989	0.999	1.000	0.626	0.964	0.658	0.679
50 ms	0.999	0.986	0.999	1.000	0.885	0.987	0.900	0.630
100 ms	0.999	0.970	0.999	1.000	0.944	0.967	0.978	0.567
200 ms	0.999	0.954	0.999	1.000	0.960	0.952	0.987	0.566

Table 3 AUCs of the supervised learning method with the Total-Counting feature and the ID-Counting feature for the four datasets. The ID-Counting feature outperformed the Total-Counting feature in almost all the cases.

Time width	Large-Parked		Large-Moving		Small-Parked		Small-Moving	
	Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting
10 ms	0.996	0.998	0.999	1.000	0.706	0.839	0.751	0.860
20 ms	0.999	0.998	0.999	1.000	0.772	0.980	0.798	0.950
50 ms	0.999	0.999	0.999	1.000	0.928	0.997	0.937	0.997
100 ms	0.999	0.999	0.999	1.000	0.965	0.998	0.987	0.995
200 ms	0.999	0.999	0.999	1.000	0.977	0.994	0.987	0.991

Table 4 Average computation time of the unsupervised method for detecting whether each message sequence is malicious or not. Although the ID-Counting features have more dimensions than the Total-Counting features, the computation time was comparable in all the cases except the Large-Moving dataset.

Large-Parked		Large-Moving		Small-Parked		Small-Moving	
Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting	Total Counting	ID Counting
1.043 ms	2.542 ms	1.327 ms	8.423 ms	0.789 ms	0.578 ms	0.886 ms	0.797 ms

with the Total-Counting feature and the ID-Counting feature. We observed that both the Total-Counting feature and the ID-Counting feature accurately detected malicious sequences in cases consisting of a large number of malicious messages (i.e., Large-Parked and Large-Moving) even when the time width was small. The performance achieved an AUC = 0.999 in most of the cases.

When the time width was over 100 ms, the Total-Counting feature achieved an AUC of over 0.90 in cases consisting of a small number of malicious messages (i.e., Small-Parked and Small-Moving). However, its performance was relatively inferior when the time width was small. This is because the number of malicious messages in a short sequence is small, and, therefore, the number of messages in a malicious sequence does not greatly differ from that in a normal sequence. The ID-Counting feature showed higher AUCs than the Total-Counting feature in these two datasets when the time width was small.

Table 3 shows the AUCs of the supervised learning method with the Total-Counting feature and the ID-Counting feature. As one can easily expect, the supervised learning method demonstrated better performance than the unsupervised learning method in most cases. Exceptions were the cases where the time width was small such as in the Small-Parked dataset; this might have been caused by the mismatch between the malicious message patterns in the training samples and those in the test samples. It is notable that the unsupervised learning method with the ID-Counting feature achieved an AUC of over 0.95 in this dataset. The ID-Counting feature generally outperformed the Total-Counting feature even when the time width was short. This highlights the effectiveness of the ID information in detecting malicious messages in the in-vehicle network.

We further investigated the computation time for detection. The experiments were run on Intel Core i5 dual-core processors at 2.9 GHz with 16 GB RAM. The time width of each message sequence was fixed to 10 ms. **Table 4** shows the average computation time of the unsupervised methods for detecting whether each message sequence is malicious or not. Although the ID-Counting features have more dimensions than the Total-Counting features, the computation time with the ID-Counting feature was comparative to that with the Total-Counting feature in almost all the cases. This result supported the efficiency of our proposed pipeline including the nearest neighbor calculation and the dimension reduction to speed up the detection. Because of the variety of messages in the Large-Moving dataset, our method with the ID-Counting feature required more time for detection; however, it is still less than 10 ms. We conclude that our proposed methods are not only accurate but also fast for detecting malicious message sequences.

5. Related Work

Several studies have reported the possibility of injecting malicious packets into in-vehicle networks and adversarially controlling several operations. For example, Koscher et al. presented several attack approaches including understanding the CAN messages and reverse-engineering the functionality, and demonstrated the capability of these attacks [8]. Their results showed that various operations such as stopping the engine, preventing the brakes, displaying a false speedometer, and disabling the headlights and the brakelights can be controlled by the at-

tacks. Another experiment demonstrated that one could override the operation of the steering, the brakes, the accelerators, and the displays [12].

These reports have motivated research for improving the CAN message security. Nilsson and Larson categorized the security approaches into four layers: (1) prevention of the modification, reading, and injecting messages in the in-vehicle network; (2) detection of malicious messages; (3) diverting attackers to trick them into believing that they are interacting with a real in-vehicle network; and (4) collecting physical and digital evidence of the attacks to apply forensic approaches [14]. Our study belongs to the second group.

Malicious message detection approaches can be divided into two main categories: one requires a modification of the CAN protocols and the other does not. An example of the first approaches is CANAuth [17]. CANAuth is a message authentication protocol for the CAN bus network, which is lightweight and does not require any reconfiguration of existing ECUs; however, all the ECUs have to share a key to use this authentication protocol. The second category is further grouped into rule-based approaches and other approaches. A rule-based approach has been proposed by Ref. [9]. This method generates protocol-level security specifications from the values appearing in each message field and the field dependency, and creates ECU-behavior security specifications by inspecting the communication patterns of each ECU. The method identifies a message to be malicious if its structure is different from the specifications.

Several statistical approaches have been proposed for detecting malicious messages without a modification of the CAN protocols. The methods are categorized into two groups: payload-based and timestamp-based. Muüter and Asaj presented an entropy-based anomaly detection method for detecting malicious messages by using payload information [13]. They demonstrated that the entropy-based method would have a capability to detect attacks by measuring how a message conforms to the expected normal behavior of CAN messages. Marchetti et al. presented the effectiveness of the entropy-based method by applying the method to real CAN traffic gathered from an unmodified licensed vehicle in real driving conditions [10]. Markovitz and Wool proposed a method for detecting malicious messages by considering the types of payload information [11]. The authors demonstrated that there were five types of payload fields: constant, multi-value, counter, sensor, and one that did not match any of the types. They built a classifier to identify the field types and the boundaries, and showed its applicability in malicious message detection. Taylor et al. applied Long Short-Term Memory (LSTM) for malicious message detection [16]. The neural network was trained to predict the payload of the next message, and its prediction errors are used as a signal for detecting malicious messages. Kang and Kang considered the detection problem as classification, and employed a deep neural network [5].

There have been a few methods aim to detect malicious messages by only using timestamp information of each message. Based on the fact that most in-vehicle messages are sent periodically, Cho and Shin proposed to estimate a model on normal clock behavior, and detected a message is malicious if its behavior deviates from the normal one [2]. Hamada et al. proposed a similar approach; they represented the interval between messages by a Gaussian mixture model, which was used for malicious message detection [4]. These two methods aim to detect *malicious messages*, whereas our method focuses on detecting *malicious message sequences*. The closest work to ours is that by Ref. [15]. The authors studied statistical methods for detecting malicious message sequences. Their methods assume that a message sequence is malicious if the time intervals of the messages in the target sequence and those in the historical sequences are drawn from independent normal distributions. In contrast, we focus on the distribution of the nearest neighbor distance between the target sequence and the normal sequences; that is, we only focus the normal sequence that is the most similar to the target sequence instead of considering all the normal sequences. Our approach would be more robust in a case where the normal sequences are diverse. In addition, we evaluate our method in both supervised and unsupervised learning cases while they only considered unsupervised cases.

6. Conclusion

In this paper, we proposed statistical anomaly-detection methods for the CAN bus network. Our methods focus on injection attacks, which can control several operations of a vehicle while one can easily implement the attacks. We proposed two types of features: the Total-Counting feature, which represents the number of elements within each message sequence, and the ID-Counting feature, which represents the frequencies of each ID to recognize a slight change in message frequency. We further introduced the unsupervised learning method and the supervised learning method for detecting malicious messages. The presented unsupervised learning method finds the nearest neighbor of a target sequence from the normal data and uses the distance between them to detect malicious messages. Because the ID-Counting feature is likely to be highly dimensional, we proposed using a dimension reduction method and a k-d tree data structure to speed up the nearest neighbor discovery.

We conducted experiments to investigate whether these features are effective for anomaly detection of the CAN data. We had four types of datasets that included an attack message and were collected from an actual car. The datasets were broadly classified into data that showed that a large number of malicious messages were injected and data that showed a small number of malicious messages were injected, and there were two situations that indicated whether the vehicle was moving or parked. By applying the proposed methods to our datasets, we indeed demonstrated that these methods can detect anomalous messages. We also found that the ID-Counting feature was more effective than the Total-Counting feature in most cases.

The result of our experiments showed that the supervised method, which requires training data, outperformed the unsupervised learning method in many cases. However, the supervised learning method requires data that cover the actual variety of attacks. When we apply our methods to an actual system, the detection performance would rely on the preparation of the training datasets. We consider the unsupervised method to be more prac-

tical, which showed comparable performance to the supervised method in several cases in our experiments.

References

[1] Bentley, J.L.: Multidimensional binary search trees used for associative searching, *Comm. ACM*, Vol.18, No.9, pp.509–517 (1975).

[2] Cho, K.-T. and Shin, K.G.: Fingerprinting electronic control units for vehicle intrusion detection, *Proc. 25th USENIX Security Symposium* (*USENIX*), pp.911–927 (2016).

[3] Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G. and Vázquez, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges, *Computers & Security*, Vol.28, No.1, pp.18–28 (2009).

[4] Hamada, Y., Inoue, M., Horihata, S. and Kamemura, A.: Intrusion detection by density estimation of reception cycle periods for in-vehicle networks: A proposal, *Proc. Embedded Security in Cars Conference* (*ESCAR*) (2016).

[5] Kang, M.-J. and Kang, J.-W.: Intrusion detection system using deep neural network for in-vehicle network security, *PLOS ONE*, Vol.11, No.6, pp.1–17 (2016).

[6] Kim, S., Cho, N.W., Kang, B. and Kang, S.-H.: Fast outlier detection for very large log data, *Expert Systems with Applications*, Vol.38, No.8, pp.9587–9596 (2011).

[7] Kolter, J.Z. and Maloof, M.A.: Learning to detect malicious executables in the wild, *Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD*), pp.470–478 (2004).

[8] Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H. and Savage, S.: Experimental security analysis of a modern automobile, *Proc. IEEE Symposium on Security and Privacy*, pp.447–462 (2010).

[9] Larson, U.E., Nilsson, D.K. and Jonsson, E.: An approach to specification-based attack detection for in-vehicle networks, *Proc. IEEE Intelligent Vehicles Symposium* (*IV*), pp.220–225, IEEE (2008).

[10] Marchetti, M., Stabili, D., Guido, A. and Colajanni, M.: Evaluation of anomaly detection for in-vehicle networks through information-theoretic algorithms, *Proc. 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow* (*RTSI*), pp.1–6 (2016).

[11] Markovitz, M. and Wool, A.: Field Classification, Modeling and Anomaly Detection in Unknown CAN Bus Networks, *Proc. Embedded Security in Cars Conference* (*ESCAR*) (2015).

[12] Miller, C. and Valasek, C.: Adventures in automotive networks and control units, available from ⟨http://www.ioactive.com/pdfs/IOActive_Adventures_in_Automotive_Networks_and_Control_Units.pdf⟩ (2013).

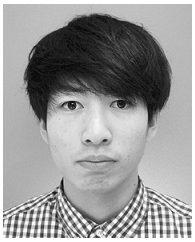
[13] Müter, M. and Asaj, N.: Entropy-based anomaly detection for in-vehicle networks, *Proc. IEEE Intelligent Vehicles Symposium*, pp.1110–1115 (2011).

[14] Nilsson, D.K. and Larson, U.E.: A defense-in-depth approach to securing the wireless vehicle infrastructure, *Journal of Networks*, Vol.4, No.7, pp.552–564 (2009).

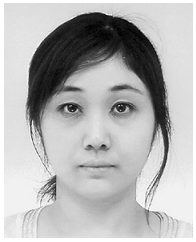
[15] Taylor, A., Japkowicz, N. and Leblanc, S.: Frequency-based anomaly detection for the automotive CAN bus, *2015 World Congress on Industrial Control Systems Security* (*WCICSS*) (2015).

[16] Taylor, A., Leblanc, S. and Japkowicz, N.: Anomaly detection in automobile control network data with long short-term memory networks, *2016 IEEE International Conference on Data Science and Advanced Analytics* (*DSAA*), pp.130–139 (2016).

[17] Van Herrewege, A., Singelee, D. and Verbauwhede, I.: CANAuth—A simple, backward compatible broadcast authentication protocol for CAN bus, *Proc. ECRYPT Workshop on Lightweight Cryptography* (2011).



Takuya Kuwahara is a master course student at Kyoto University. He received his B.S. in Engineering from Kyoto University in 2016. His research interest includes anomaly detection and data mining.



Yukino Baba is an assistant professor at Department of Intelligence Science and Technology, Kyoto University. She received her Ph.D. degree from The University of Tokyo in 2012. Her research focuses on machine learning, data mining, and human computation.

Hisashi Kashima is a professor at Department of Intelligence Science and Technology, Kyoto University. He received his Ph.D. degree from Kyoto University in 2007. He was a researcher at IBM Tokyo Research Laboratory during 1999–2009, and was an associate professor at The University of Tokyo during 2009–2014. His research focuses on machine learning, data mining, and human computation.

Takeshi Kishikawa received his M.E. degrees from Yokohama National University in 2013. In 2013, he joined the Corporate R&D Division, Panasonic Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.

Junichi Tsurumi received his M.E. degrees from Kobe University in 2014. In 2014, he joined the Corporate R&D Division, Panasonic Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.

Tomoyuki Haga received his M.E. degrees from Chuo University in 2001. In 2001, he joined the Corporate R&D Division, Matsushita Electric Industrial (now Panasonic) Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.

Yoshihiro Ujiie received his M.E. degrees from Osaka University in 2007. In 2007, he joined the Corporate R&D Division, Matsushita Electric Industrial (now Panasonic) Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.

Takamitsu Sasaki received his M.E. degrees from Kyushu University in 2004. In 2004, he joined the Corporate R&D Division, Matsushita Electric Industrial (now Panasonic) Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.

Hideki Matsushima received his M.E. degrees from Kobe University in 1998. In 1998, he joined the Corporate R&D Division, Matsushita Electric Industrial (now Panasonic) Co., Ltd., Osaka Japan. His research interest is cyber security for IoT, including the automotive area.