# Identifying ECUs Using Inimitable Characteristics of Signals in Controller Area Networks

Wonsuk Choi, Hyo Jin Jo, Samuel Woo, Ji Young Chun, Jooyoung Park, and Dong Hoon Lee, *Member, IEEE*

***Abstract*—As the functions of vehicles are more computerized for the safety and convenience of drivers, attack surfaces of vehicle are accordingly increasing. Many attack results have shown that an attacker could intentionally control vehicles. Most of them exploit the vulnerability that controller area network (CAN) protocol, a de-facto standard for the in-vehicle networks, does not support message origin authentication. Although a number of methods to resolve this security vulnerability have been suggested, they have their each limitation to be applied into the current system. They have required either the modification of the CAN standard or dramatical communication load increase, which is infeasible in practice. In this paper, we propose a novel identification method, which works in the physical layer of the in-vehicle CAN network. Our method identifies electronic control units (ECUs) using inimitable characteristics of electrical CAN signals enabling detection of a malicious ECU. Unlike previous attempts to address the security problem in the in-vehicle CAN network, our method works by simply adding a monitoring unit to the existing network, making it deployable in current systems and compliant with required CAN standards. Our experimental results show that our method is able to correctly identify ECUs. In case of misclassfication rate for ECU idnetification, our method yields 0.36% in average which is approximate four times lower than the method proposed by P.-S. Murvay *et al.* This paper is also the first to identify potential attack models that systems should be able to detect.**

***Index Terms*—Controller Area Network, Electronic Control Unit, Automotive ID, Device Fingerprinting.**

## I. Introduction

ELECTRONIC Control Units (ECUs) were originally proposed to ensure optimal engine performance (efficient gasoline and oil consumption) [1]. In recent years, automotive manufactures have installed ECUs not only for engine control but also for various functions related to safety and convenience of drivers. Anti-lock Braking System (ABS) and Parking Assist System are typical examples that enhance drivers' safety and convenience supported by ECUs. For luxury sedans, 50–70 distinct ECUs exist in a vehicle [2]. Among several protocols enabling ECUs to communicate in the in-vehicle network, the CAN (Controller Area Network) protocol is the most widely used [3]. The reliability and simple network structure of the CAN protocol have made it a de facto standard for ECU communication over 30 years.

Despite the advantages of the CAN protocol, it was not designed with security features in mind; hence, the in-vehicle CAN network is open to attack from malicious messages intended to cause malfunctions. The authors of [2] demonstrated attacks on modern vehicles in which adversaries were able to systematically control a wide array of components including the engine, instrument panel, and radio. They had physical access to the in-vehicle CAN network in order to transmit malicious CAN messages into it. The authors of [4] demonstrated long-range or indirect physical access attacks. Compared to physical access to the in-vehicle CAN network, long-range or indirect physical access makes vehicle owners more difficult to recognize the attacks. For example, they suggested that an adversary may deliver malicious CAN messages by encoding it as a media file. When the modified media file is played on a car audio system, the malicious messages causing intentional malfunctions are transmitted into the in-vehicle CAN network. In 2015, for the first time, the authors of [5] demonstrated how to access the in-vehicle CAN network on unaltered vehicle. They exploited vulnerabilities of telematics devices connected to a cellular network in order to remotely deliver malicious CAN messages. The Telematics services have even privacy-related problems because it they provides location-based services [6]–[11]. This Location privacy is one of the important issues that have has been being addressed in other research areas [8], [12].

The fundamental security problem associated with the in-vehicle CAN network is that the CAN protocol does not support message authentication. It is easy for adversaries to cause intentional malfunctions if they succeed in accessing the in-vehicle CAN networks. Unfortunately, the length of data field in a CAN frame is only 1–8 bytes while the MAC (Message Authentication Code) needed for adequate security is more than 20 bytes. To address this vulnerability, some researches have been conducted on alternative methods [13]–[17]. The authors of [16] developed a CAN+ protocol initially proposed in [18], which inserts extra bits between the sampling points of a CAN

Manuscript received July 2, 2016; revised March 17, 2017 and October 30, 2017; accepted February 13, 2018. Date of publication February 27, 2018; date of current version June 18, 2018. This work was supported by Samsung Research Funding and Incubation Center for Future Technology under Project SRFC-TB1403-51. The review of this paper was coordinated by Prof. H. Wang. *(Corresponding author: Dong Hoon Lee.)*

W. Choi, J. Y. Chun, and D. H. Lee are with the Graduate School of Information Security, Korea University, Seoul 02841, South Korea (e-mail: wonsuk85.choi@gmail.com; jychun@korea.ac.kr; donghlee@korea.ac.kr).

H. J. Jo is with the Department of Computer and Information System, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: hyojinjo@upenn.edu).

S. Woo is with the Network Security Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea (e-mail: samuelwoo@etri.re.kr).

J. Park is with the Department of Control and Instrumentation Engineering, Korea University, Sejong City 30019, South Korea (e-mail: parkj@korea.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2810232

bus interface. The extra bits enables MAC to be included in a CAN frame. However, this technique requires a new CAN controller having higher sampling rate, meaning that the current all ECUs would have to be replaced. The authors of [15], [17] have suggested truncated MAC. Their methods require frequent key update due to their limited length of MAC. In addition, key update during vehicles are moving might rather cause malfunctions. The authors of [13], [14] presented the methods to send additional messages for message authentication. Their methods lead to a rapid increase in the bus load (to more than 50%). In general, the bus load has to remain under 50% of the maximum in order to preserve a stable communication environment [17]. As a result, the existing works that attempted to solve the fundamental problem in CAN protocol cannot be a adequate solution in practice.

In this paper, we propose a novel ECU identification method that resolves the fundamental security problem associated with the in-vehicle CAN network. We build on the idea of source identification using inimitable characteristics of CAN electrical signal transmitted by ECUs, as first suggested by [19]. Because our method analyze electrical CAN signal in physical layer of the in-vehicle CAN network, we do not have to consider the problems of the existing works based on MAC. In addition, our method can be applied into the current system without ECU hardware modification. Our method can be used for such security purposes as intrusion detection (discovering ECU impersonation and network tampering), authentication (preventing unauthorized access to the physical network), forensic data collection (tying a physical device to a specific network incident), and quality assurance monitoring (determining whether a device will or is in the process or failing).

### A. Our Contributions

The method of [19] reflected two weaknesses. First, they did not evaluate their method on ECUs that manage critical functions using the high-speed CAN network, but only used the bit rate used in ECUs that manage simpler functions on the low-speed CAN network. Since it is much easier to achieve device identification using the lower bit rate [20], they did not appropriately simulate the CAN environment. Our method overcomes this weakness by using the same bit rate that is used on the high-speed CAN when ECUs perform critical functions (i.e, 500 K b/s). This is a significantly higher bit rate than [19]. Second, they failed to consider how situations involving collisions would impact signal identification using, as he did, the identifier field. A collision occurs when more than two ECUs simultaneously try to transmit CAN messages. In a collision situation, signals are generated from multiple ECUs, priority is assigned via the arbitration decision, and lower priority signals are cut off. The identifier field, which [19] relied upon, is unable to differentiate signals under these collision circumstances, and therefore is insufficient to accurately identify signal characteristics. By contrast, our method measures signals using the extended identifier field in the extended frame format of the CAN protocol, which is able to accurately identify signals from all ECUs free of any collision confusion that may be present in the identifier field. Our main contributions are as follows:

- Our method is able to identify ECUs only by installing an additional device, meaning that our method can be directly applied into current vehicles.
- Our method complies with the current CAN standards and so does not require replacement or alteration of ECUs that have been installed in in-vehicle networks.
- Our method improves upon the work of [19] in that it analyzes more features of in-vehicle CANs, including those that control critical functions; identifying ECUs accurately even under collision conditions when more than two ECUs transmit CAN messages simultaneously; and testing our method at the same bit rate (500 K b/s) as the bit rate of the in-vehicle CAN network..

The rest of the paper is structured as follows: Section II describes background needed to understand our method. In Sections III and IV, we propose our system model and our method for ECU identification, respectively. In Section V, we present our experimental design and results, showing the accuracy of our method. Section VI describes related work revealing research trends. Finally, we describe future work and our conclusion in Sections VII and VIII, respectively.

## II. Backgrounds

In this section, we describe device inconsistencies inherent in ECUs and how Controller Area Networks (CANs) function, in order to provide background necessary to understand our method.

### A. Inconsistency of Device Signals

We have already mentioned that message authentication is actually impossible in the in-vehicle CAN network. Hence, we do not seek to use MAC, but rather the goal of our work is to identify ECUs by examining the distinctive analog characteristics of these devices. By differentiating ECUs based on the inimitable characteristics of electrical CAN signals emitted by individual ECUs, we can identify ECUs. The electrical CAN signals with other characteristics would therefore be considered to be attacks from a malicious adversary emanating from other ECUs that have become hostage to an adversary or from alien ECUs planted by the adversary in the vehicle. Fig. 1 shows an example of inherent variations in signaling behavior of ECUs due to hardware and manufacturing inconsistencies. Even though two ECUs transmit the exact same messages, they generate slightly different analogue signals. If two ECUs might appear to be identical products from the same vendor, they generate even different signals. Those inconsistencies cause minute and unique variations in the signaling behavior of every digital device [20]. In general, the variations are within a defined range and the unique characteristics of signals from individual ECUs remain constant over time; therefore, ECUs can be identified by their unique characteristics.

### B. CANs (Controller Area Networks)

CAN is a communication protocol which was developed in the mid-1980s by Bosch GmbH [3]. The CAN protocol was first designed to provide a cost-effective communication bus for

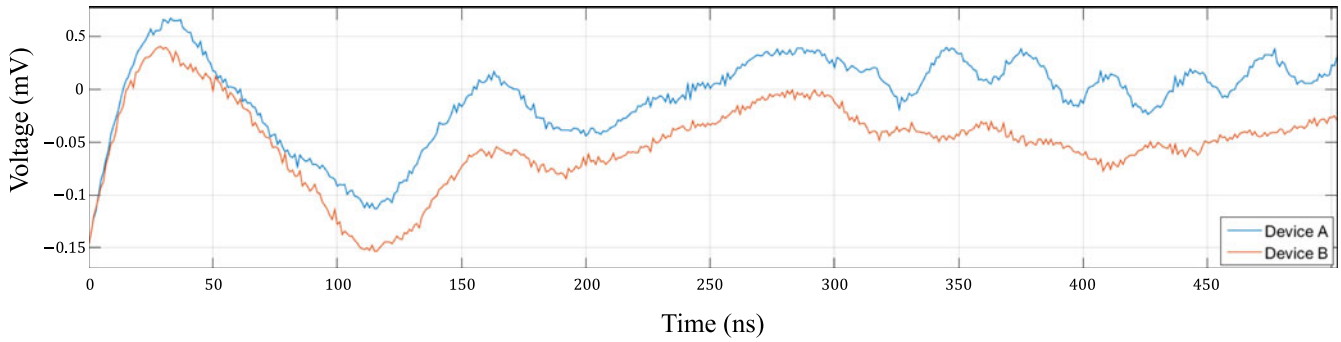



Fig. 1. An example of variations in signaling.

automotive applications, but is today widely used in various industries such as aerospace and railways, elevators, and medical devices [21]. It is necessary to understand the physical layer of the CAN protocol, because our method deals with characteristics in the physical layer. Since today's in-vehicle communications are designed based on the CAN 2.0 standard, we explain the CAN standards in terms of i) encoding/data transmission, ii) data frame formats, and iii) arbitration. In general, the in-vehicle CAN network topology is divided into two groups: the high-speed CAN and low-speed CAN (fault-tolerant CAN). The low-speed CAN consists of ECUs for easy functions such as the door open/lock function. Other more critical functions, such as engine or brake functions, are connected to the high-speed CAN. In this paper, we only consider the high-speed CAN, because this presents the most challenging environment for identifying signal characteristics [20]. Accordingly, once we show that our method is able to identify characteristics at a higher bit rate, it will follow that it will also be able to do so at a lower bit rate.

*1) Encoding/Data Transmission:* The CAN signal is encoded using the Non Return to Zero (NRZ) bit encoding method, in which 1 is represented by one particular voltage and 0 is represented by some other significant voltage. A twisted pair of two wires in a shielded cable is used for CAN communication. Both lines, CAN-H (High) and CAN-L (Low) are biased at 2.5 volts in the case of the recessive state (1). For the dominant state (0), CAN-H goes to around 3.5 volts and CAN-L to 1.5 volts, respectively. In terms of device inconsistencies, those degrees of voltage are all different for different devices. From those differences, we determine the signal characteristics that correspond to particular ECUs.

*2) Data Frame Format:* The CAN protocol defines four types of frames: i) data frame, ii) remote frame, iii) error frame, and iv) overload frame. Of these four types, we only describe the data frame, because the purpose of our method is to identify which ECU is transmitting the data frame. As an interesting features of the CAN protocol, the identifier field of the data frame refers to the identifier of the transmitter ECU not that of the receiver. Accordingly, it is possible to determine which ECU is transmitting CAN messages frame by checking the identifier. Our goal is to detect an impersonator and identify the ECU from which that impersonator is transmitting, using its unique signal characteristics. There are two types of data frames: one is the base frame format whose identifier length is 11 bits, and

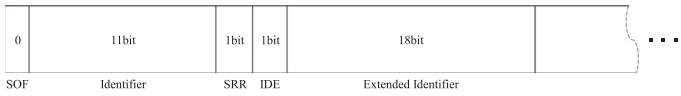



Fig. 2. CAN data frame format (Extended frame).

TABLE I
FIELDS OF EXTENDED FRAME FORMAT

| Field | Length (bits) | Description |
|---|---|---|
| Start-of-frame | 1 | Means the start of frame transmission |
| Identifier | 11 | An identifier of transmitter, which also represents the message priority |
| Substitute remote request (SRR) | 1 | Must be recessive state (1) |
| Identifier extension bit (IDE) | 1 | Indicating whether 11 bit identifier or 29 bit extended identifier is used. Dominant state (0) indicate 11 bit identifier while Recessive state (1) indicate 29 bit extended |
| Extended identifier (EXID) | 18 | If IDE is Recessive state (1), EXID field is available |
| Reserved bits | 2 | – |
| Data length code (DLC) | 4 | Number of bytes of data (0–8 bytes) |
| Data field | 0–64 | Data to be transmitted |
| CRC | 15 | Cyclic redundancy check |
| CRC delimiter | 1 | Must be recessive state (1) |
| ACK slot | 1 | Transmitter sends recessive state (1) and any receiver can assert a dominant sate (0) |
| ACK delimiter | 1 | Must be recessive state (1) |
| End-of-frame | 7 | Must be recessive sate (1) |

the other is the extended frame format, which has 29 identifier bits made up of the 11-bit identifier and an 18-bit identifier extension. We refer to the extended identifier as EXID. To use the extended frame format, the identifier extension bit (IDE) is transmitted in the recessive (1) state. Fig. 2, shows an overview of the CAN data frame and we describe each field in the data frame in Table I. Our method measures signals corresponding to the bit string of the extended identifier field. The reason why we do not use the signals corresponding to the identifier field is because of arbitration. We will describe arbitration below.

*3) Arbitration/Extended Identifier:* There are times when more than two ECUs try to simultaneously transmit their messages. To prioritize such collision signals, the CAN protocol supports the arbitration decision process that is able to lossless

| | Start of Frame (1 bit) | Identifier (11 bits) | | | | | | | | | | | | | | SRR | IDE | Extended Identifier (18 bits) | Rest of Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ECU A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | ... | |
| ECU B | 0 | 0 | 0 | 0 | 0 | 0 | 1 | No Transmission | | | | | | | | | | | | |
| CAN Bus | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | ... | |

Fig. 3. An example of arbitration between two nodes.

bit-wise arbitration method. If one ECU transmits a dominant bit (0) and another ECU transmits a recessive bit (1), then there is a collision and the dominant bit gets higher priority. During transmission, an ECU continuously checks on the CAN bus and compares the signal states, recessive or dominant. Fig. 3 shows an example of how one of two ECU signals gets higher priority through arbitration decision. Because the identifiers of ECU A and B are the same until 5th bit, both ECUs check the same bits as the bits they transmit on the CAN bus. For the first time, ECU B checks a different on the CAN bus. ECU A gets higher priority, and ECU B's signal is cut off and can try again later. As a result, the arbitration decision ends, multiple signals in the identifier field may be generated from multiple ECUs. In our example in Fig. 3, the first five parts of the signals being sent from the two ECUs are the signal generated by both ECUs. Accordingly, it is impossible to identify the unique characteristics of each single ECU from the signals in the identifier field. The extended identifier field was originally created in the CAN protocol to accommodate situations requiring identification of more connection devices than the identifier field could accommodate. The identifier field in the CAN protocol is able to assign identifiers to many more connection devices than the 50–70 ECUs found in vehicles, since the identifier field is capable of identifying thousands of connection devices at a time. The increased capacity of the extended identifier field was needed for other industry applications. However, in the case of in-vehicle ECUs, the deficiency of the identifier field is not in the number of connection devices it can potentially identify, but rather is in its incapacity to differentiate the accurate ECU origin of signals under collision conditions. Hence, our activation and use of the extended identifier field is not to increase the number of potential identifiers, as it was originally created to do, but rather to overcome the problem of signal identification created by signal collisions. Since there are only 50–70 ECUs in a vehicle and the identifier field will have been sufficient to handle all arbitration decisions, we therefore programmed the extended identifier field to be included into the data frame along with the content of the message.

## III. System Model

In this section, we present a system model for our method. Fig. 4 shows how our method works in order to identify ECUs in the in-vehicle CAN network. As shown in Fig. 4, our system model adds a monitoring unit to the CAN bus network. The monitoring unit is programmed to analyze sent CAN messages in several ways. First, it extracts the known ECU identifier from the identifier field and determines whether the identifier is a known identifier or not. Second, the monitoring unit analyzes

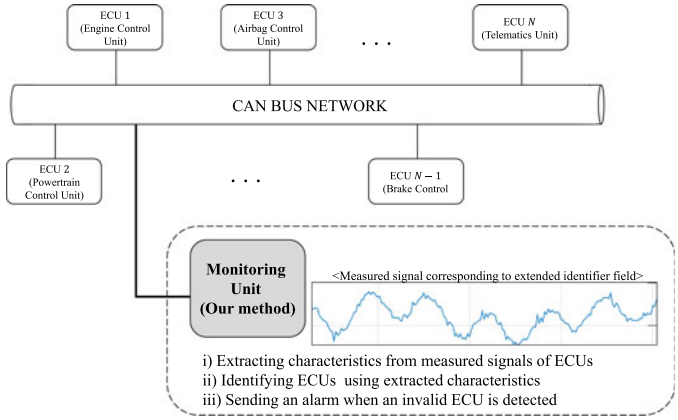



Fig. 4. System model.

the features of the electrical CAN signals in a sent message with the enabled extended identifier field. Basically, the monitoring system utilizes a classification algorithm to which the same message is sent multiple times so that the algorithm can learn them to create a fingerprint template of the unique characteristic from each ECU. Hence, the unique signal characteristics of each ECU are contained in fingerprints as determined by our fingerprinting method. For example, the monitoring unit will know that signals to turn on and off the engine are sent by ECU 1, while signals to control the brakes are sent by ECU 2. The monitoring unit is programmed to know which identifiers should match which fingerprints: it has both ECU identifiers and signal fingerprints in correct pairings.

The monitoring unit receives signals sent by all ECUs, and since it knows which signals should be coming from which ECU, it will know when signals are sent by the wrong ECU. In the case of an adversary attack, the adversary can use the identifier of the ECU that the adversary knows should be sending the signal (telling the engine to shut off, for example). When that happens, the monitoring unit will know that the signal did not originate from the ECU that it is paired with, the one that is supposed to be generating that signal. This will identify the presence of an adversary and the mismatched fingerprint will identify the ECU that was actually used by the adversary to send the signal, if it is an existing ECU that has been taken hostage by the adversary, or it will indicate that the signal originated from some other completely unknown device. In other words, when a particular signal is not being generated from the properly assigned ECU identifier, the monitoring system knows it and it then sends an alarm to the vehicle owner.

Because ECUs are all connected in the same shared bus line, we can apply our method to current vehicle systems by adding the monitoring unit to the in-vehicle CAN network and adjusting ECU programming to include the extended identifier field. To clarify our method, we will describe some possible adversary models and our underlying assumptions about the CAN.

### A. Adversary Models

The goal of adversaries is to transmit malicious commands to the in-vehicle CAN network so that the adversary can take control of the vehicle. Because the CAN protocol does not support message authentication, an adversary could can apply a

simply simple attack by connecting with the in-vehicle CAN network and launching a replay attack. In the future, telematics devices would will be connected with mobile social networks, that which could be used as an attack channel for the spreading of malicious software [22].

We consider two types of adversary threats based on how the adversary would be able to access the in-vehicle CAN network.

*Adversary Model Type I:* The first adversary method is one where the adversary gains access through an additional external device that is physically planted by the adversary. The new device could be inserted into the On-Board Diagnostics (OBD)-II port, which is an interface between in-vehicle networks, including the in-vehicle CAN network. OBD systems give the vehicle owner or repair technician access to the status of various vehicle subsystems. Various tools are available that plug into the OBD connector to access in-vehicle sub-networks, including in the CAN network. Since the OBD-II port is located under the dashboard, an adversary could gain access to the in-vehicle CAN network by attaching an additional device to the OBD-II port. The adversary could also gain access through an external device (e.g., laptop or smartphone). A mobile device-based or PC-based scan tool is one that an adversary can easily buy on the Internet [23], [24]. Using such additional devices, an adversary is able to transmit malicious commands.

*Adversary Model Type II:* The second adversary model is for the adversary to compromise an existing ECU in the in-vehicle CAN network. In this attack, an adversary basically takes an existing ECU hostage to imitate existing signal features and send messages with the correct signal but from a different ECU that they have been able to take hostage. To root out attackers who are operating from a hostage ECU, we needed to identify signal fingerprints that are not paired with the correct ECU identifier. One example would be compromise of a telematics ECU. Telematics ECUs are being installed in more and more vehicles to enable additional functions (e.g., remote engine start through a smartphone). Due to the fact that telematics ECUs provide several access points of access through external networks such as Bluetooth, Wi-Fi, and 3G communication, telematics ECUs are vulnerable to attack by the adversary. In fact, Chekoway *et al.* performed such an attack on a telematics ECU using these known vulnerabilities [4]. After compromising a telematics ECU, the adversary is then able to transmit messages to intentionally cause malfunctions.

We separate adversary models according to the way an adversary is able to connect to the in-vehicle CAN network. We consider Adversary Model Type I to constitute a lesser threat than Adversary Model Type II. For one thing, it is difficult for a Type I adversary to succeed since it requires additional devices to be plugged into the OBD-II port that can easily be seen by the driver and removed. Since Type I is so much more difficult to localize and eradicate, we designed our method to be effective with a Type II adversary. Our method can be used to address both types of adversaries

### B. Our Assumptions

Our method uses the extended frame format instead of the base frame format, since the base frame format with only its identifier field does not properly identify signal characteristics in collision situations, described in Section II. The extended frame format contains both the 11-bit identifier field and also an 18-bit extended identifier field. We assume that all arbitration decisions are made within the identifier field since we know that it has the capacity to accommodate all signals sent by the 50–70 some vehicle ECUs. Based on the assumption that all collision signals will have been subject to arbitration decisions in the identifier field, we further assume that signals in the extended identifier field are not affected by arbitration decisions so that the unique characteristics of signals from different ECUs will be distinguishable. In other words, we assume that the extended identifier field will allow measurement of physical signals of all sent messages without interference from the effects of arbitration that render the identifier field by itself to be unreliable in accurately distinguishing those characteristics. The second assumption is that it is impossible to imitate an ECU's unique signal characteristics. Due to the effects of hardware inconsistencies, there are subtle differences between devices in terms of signal variation. Thus, we assume that those variations are unique characteristics of devices and inimitable. Although an adversary could learn ECU identifier, the adversary could not hide the fact that a message is being sent from an alien or from a hostage ECU. Finally, we assume that a single malicious command will not be effective to cause vehicle malfunction. For example, if an adversary wants to incapacitate the brake system, he must transmit malicious messages over and over until the intended malfunction occurs. This means that a single malicious message is assumed to be ineffective. We therefore assume that an adversary will transmit malicious messages repeatedly in order to perpetrate an attack on a vehicle.

## IV. Our Method

We describe our method that is able to identify ECUs in the in-vehicle CAN network. We utilized the characteristics of electrical CAN signals that are uniquely generated from ECUs. Our main idea is to extract suitable statistical features of electrical CAN signals, which include unique characteristics, and then perform a supervised learning for classification.

### A. Assigning Bit String/Data Collection

Our method measures electrical CAN signals that will be analyzed for ECU identification. Fig. 5 shows an example of an electrical CAN signal corresponding to a CAN data frame with an extended identifier (EXID). From a whole signal, only the part of the signal corresponding to EXID will be analyzed. First, an 18-bit string is assigned to the EXID field, and thus every CAN data frame has the same EXID in our method. To avoid bit stuffing at an unexpected position, the most significant bit (MSB) of EXID must be 0. In the CAN standard, six consecutive bits of the same type are considered to be an error; thus, the stuff bit that corresponds to the opposite polarity is automatically inserted after five consecutive bits of the same polarity. Because the identifier extension bit (IDE) is located before the EXID, and its value must be 1 for an extended frame format, our method is able to obtain the signal in which bit stuffing always occurs at the same position within EXID by assigning an MSB of EXID to 0. The other 17 bits can be any value in our method, whereas

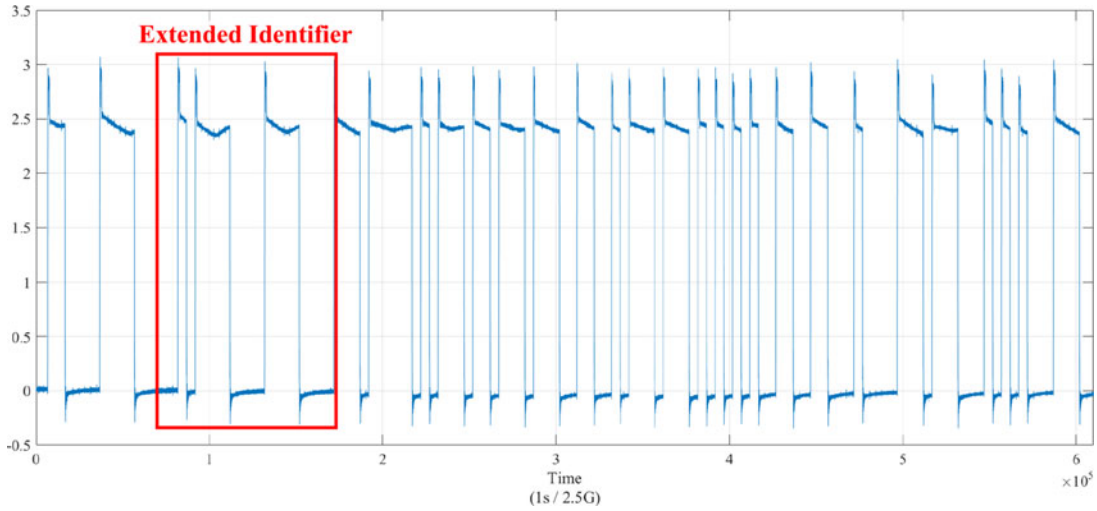



Fig. 5. An example of electrical CAN signal.

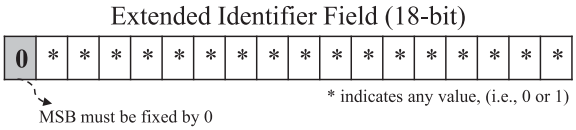



Fig. 6. The condition of the extended identifier field used in our method.

MSB must be 0. Fig 6 illustrates the condition of an extended identifier field with our method. As a result, our method can measure the part of the electrical CAN signal corresponding to the EXID field without considering the bit stuffing positions. The measured signal is denoted by $s$, which is a set of sampled values.

### B. Feature Extraction

We took the method proposed by the authors of [25] in order to extract suitable statistical features from the measured signal $s$. For the measured signal $s$, we extracted 40 scalar features in both time and frequency domains using LibXtract, a popular feature extraction library [26]. Among the 40 possible features, we selected the only relevant features by ranking the features using the FEAST toolbox and utilize the joint mutual information criterion for ranking [27]. From the results, we selected the top 8 time-domain features and 9 frequency-domain features. Tables II and III show the selected 17 features.

A set of selected features of $s$ is denoted as $F(s)$. As a result, our method extracts a set of features $F(s)$ corresponding to a measured signal $s$. Accordingly, $F(s)$ represents a fingerprint of an ECU.

### C. Training Phase (Fingerprint Template Generation)

Our method uses a supervised learning to identify the source of signals (i.e., ECU). The supervised learning has two main phases: training phase and testing phase. In the training phase, a classifier is created which can be used as fingerprint templates. The concept of using a supervised learning is to train a classifier with a set of electrical CAN signals that valid ECUs generate and then, the signal that a malicious ECU generates can be identified by the trained classifier. It is noted that the supervised learning is broadly utilized as a useful tool for security methods. For example, IDSs (Intrusion Detection Systems) can be designed with a classification algorithm to detect an anomaly behavior in a network [28].

We assume that no attack could occur during the training phase. The classifier is trained with n sets of features per ECU and its label. It would need to be performed again whenever a new ECU is added to or an ECU is removed from the in-vehicle network; however, we note that adding, removing, or replacing an ECU after initial manufacture is an infrequent occurrence in an in-vehicle network, so that this learning phase is not frequently performed after leaving the factory.

TABLE II
A LIST OF TIME DOMAIN FEATURES

| Feature | Description |
|---|---|
| Mean | $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x(i)$ |
| Std-Dev | $\sigma = \sqrt{\frac{1}{N-1}\sum_{i-1}^{N}(x(i)-\overline{x})}$ |
| Average Deviation | $D_{\overline{x}} = \frac{1}{N}\sum_{i=1}^{N}\lvert x(i)-\overline{x}\rvert$ |
| Skewness | $\gamma = \frac{1}{N}\sum_{i-1}^{N}\left(\frac{x(i)-\overline{x}}{\sigma}\right)^3$ |
| Kurtosis | $\beta = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x(i)-\overline{x}}{\sigma}\right)^4 - 3$ |
| RMS Amplitude | $A = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x(i))^2}$ |
| Lowest Value | $L = (\mathrm{Min}\,(\mathrm{x(i)}))\lvert_{\mathrm{i=1..N}})$ |
| Highest Value | $H = (\mathrm{Max}\,(\mathrm{x(i)}))\lvert_{\mathrm{i=1..N}})$ |

Vector $x$ is the time domain representation of electrical CAN signal $s$. $N$ is the number elements in $x$

### D. Testing Phase (Fingerprint Matching)

After completing the above training phase, *Fingerprint template generation*, which produced fingerprint templates for valid ECUs in the in-vehicle CAN network, we then proceeded to the testing phase where new signals are classified. The trained

TABLE III
A LIST OF FREQUENCY DOMAIN FEATURES. VECTOR $y$ IS THE FREQUENCY DOMAIN REPRESENTATION OF ELECTRICAL CAN SIGNAL $s$

| Feature Name | Description |
|---|---|
| Spec. Centroid | $\mu = \frac{\sum_{i=1}^{N} y_f(i) y_m(i)}{\sum_{i=1}^{N} y_m(i)}$ |
| Spec. Entropy | $H = \sum_{i=1}^{N} w_i \cdot \log_2 w_i$, where $w_i = \frac{y_m(i)}{\sum_{i=1}^{N} y_m(i)}$ |
| Spec. Spread | $\sigma = \sqrt{\sum_{i=1}^{N}((f_i - \mu)^2 \cdot w_i)}$ |
| Spec. Skewness | $\text{Skewness} = \frac{\sqrt{\sum_{i=1}^{N}(f_i-\mu)^3 \cdot w_i}}{\sigma^3}$ |
| Spec. Kurtosis | $\text{Kurtosis} = \frac{\sqrt{\sum_{i=1}^{N}(f_i-\mu)^4 \cdot w_i}}{\sigma_s^4}$ |
| Spec. Brightness | $\text{Brightness} = \sum_{i=f_c}^{N} y_m(i)$, where $f_c$ is the cut-off frequency |
| Spec. Roll off | $\underset{f \in 1,\ldots,N}{\operatorname{argmin}} \sum_{i=1}^{f} y_m(i) \geq 0.85 \cdot \sum_{i=1}^{N} y_m(i)$ |
| Spec. Irregularity | $\text{Irregularity} = \frac{\sum_{i=1}^{N-1}(y_m^2(i) - y_m^2(i+1))^2}{\sum_{i=1}^{N-1}(y_m^2(i))^2}$ |
| Spec. flux | $\text{Flux} = \sum_{i=1}^{N-1}(N(i) - N(i+1))^2$, where $N(i)$ is the normalized magnitude as $N(i) = \frac{y_m(i)}{\sum_{i=1} N y_m(i)}$ |

Vectors $y_m$ and $y_f$ hold the magnitude coefficients and bin frequencies respectively. $N$ is the number of elements in $y_m$ and $y_f$

classifier predicts the most probable ECU for the unlabeled set of features. In our method, testing phase can be divided into two steps. First, our method tests a unlabeled set of features for Type I adversary. Since Type I adversaries utilize additional devices that are not trained and therefore the classifier needs to determine whether there is a sufficient match with any of the known fingerprints or not. If the classifier cannot match new signals with any known signals fingerprint templates, the inquiry is over. It means there is an adversary with a completely alien signal from a completely alien device. The alien signal identification belongs to novelty detection which is the identification of new or unknown data that a machine learning system has not been trained with [29]. We presented a simple threshold-based approach for alien detection problem, which is a convenient extension of our classification-based basic model. Comparing a new signal with a given feature set F(S) to all known fingerprint templates in the classifier yields scores showing the new feature set F(s) belongs to the known one of the fingerprint templates. Thus, if all scores are below the threshold, we classify the feature set F(S) to the class of unknown sets (i.e., an invalid ECU or device is the source of the signal). In short, we set a threshold, explained in SubSection V-D, below which the classifier determines that a newly identified fingerprint is an unknown adversary. An alarm will be sent. Once our method has determined that there are no alien ECUs, our method classifies the new signal into one of classes (i.e., ECUs) Our method checks if the classification result is matched into CAN ID that can be seen in the corresponding CAN frame. A mismatch resulted in an alarm to the owner.

**Algorithm 1:** Our method.

```
1:  function TRAINING(S: a set of electrical CAN signals)
2:      for i = 1 to |S| do
3:          CAN_data_i ← DECODING_CAN_SIGNAL(s_i ∈ S)
4:          Label_i ← GET_CAN_ID(CAN_data_i)
5:          s'_i ← GET_SIGNAL_OF_EXTENDED_IDENTIFIER(s_i)
6:          [f1_i, f2_i, ... f18_i] ← FEATURE_EXTRACTION(s'_i)
7:          TrainingSet(i) ← [f1_i, f2_i, ..., f18_i : Label_i]
8:      end for
9:      Classifier ← GET_CLASSIFIER(TrainingSet)
10:         /* e.g., SVM, NN, or BDT algorithm */
11:     return Classifier
12: end function
13:
14: function ECU_IDENTIFICATION(s: a new electrical CAN signal)
15:         CAN_data ← DECODING_CAN_SIGNAL(s)
16:         s' ← GET_SIGNAL_OF_EXTENDED_IDENTIFIER(s)
17:         [f1, f2, ... f18] ← FEATURE_EXTRACTION(s')
18:         [Class, Score] ← CLASSIFICATION ([f1, f2, ..., f18], Classifier)
19:             /* Classifier must be learned in advance at TRAINING function */
20:         if Score< threshold then
21:             return Type 1 Adversary      /* Alert: anomaly detected */
22:         else if Class ≠ GET_CAN_ID(CAN_data) then
23:             return Type 2 Adversary      /* Alert: anomaly detected */
24:         else
25:             return No Adversary
26:         end if
27: end function
```

A flow chart of our method is shown in Fig. 7, and Algorithm 1 describes how the classifier is created and ECU is identified based on their electrical CAN signals. Although we did not describe the sub-functions used in Algorithm 1, it should not be difficult to see how they are applied.

## V. EXPERIMENTAL RESULT

We evaluated our method on our CAN bus prototype. By a series of experiments, it is justified that electrical CAN signal can be used as fingerprint of ECUs so that the in-vehicle network intrusions can be detected by our method.

### A. Experimental Setup

We describe our CAN bus prototype and classification algorithms that are used for evaluation. Table IV describes the equipments we used and their specification.

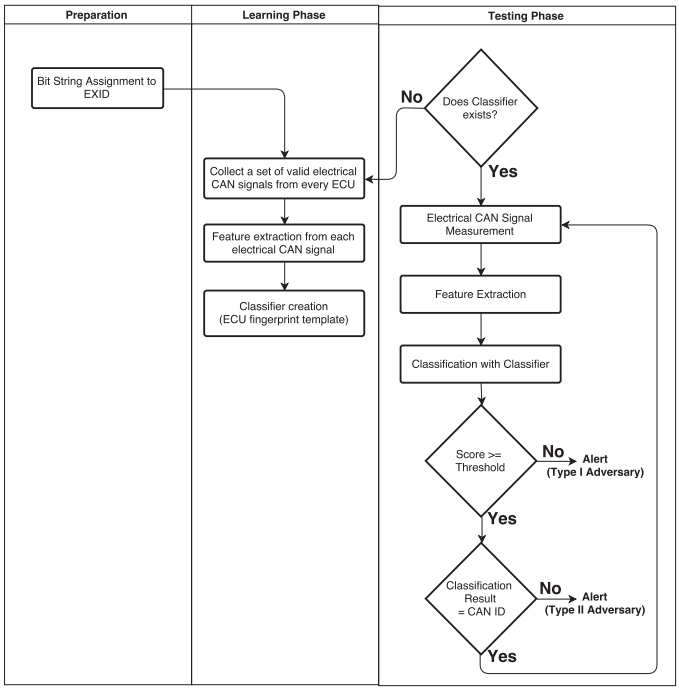



Fig. 7. Flow chart of our method.

TABLE IV
COMPONENTS OF EXPERIMENTAL SETUP

| Components | Specification | Explanation |
|---|---|---|
| Oscilloscope | HDO6104 (Lecroy)<br>Bandwidth: 1 GHz<br>Sampling rate: 2.5 GS/s<br>Vertical resolution: 12-bit | For a CAN frame, it measures electrical CAN signal |
| CAN bus network | CAN breakout box (National Instrument) | It provides CAN bus network |
| Monitoring tool for CAN Communication | Off-the-shelf laptop CAN Communication monitoring tool: PCAN-Explorer | It monitors CAN packets on the CAN bus network |
| CAN development board (Type A) | Order-made board<br>Microcontroller: TMS320F28335 (TEXAS INSTRUMENTS)<br>Transceiver: MCP2551 (MICROCHIP) | It transmits CAN frames and the corresponding electrical CAN signals |
| CAN development board (Type B) | TR28335 Training Kit (SyncWorks)<br>Microcontroller: TMS320F28335 (TEXAS INSTRUMENTS)<br>Transceiver: SN65HVD235 (TEXAS INSTRUMENTS) | It transmits CAN frames and the corresponding electrical CAN signals |

*CAN Bus prototype:* We set up a CAN bus prototype to simulate the CAN bus network with 12 CAN development boards. Even though the total number of ECUs is approximately 70 in case of a luxury sedan, in-vehicle networks are physically divided into several subnetworks. The Volvo XC90, for example, has 7-12 ECUs for an in-vehicle subnetwork [30]. In the CAN protocol, two types of network topologies are defined: low-speed CAN (fault tolerant) and high-speed CAN. In general, high-speed CAN is used for the in-vehicle CAN network where critical-function ECUs are connected at 500 Kbps. Our CAN bus prototype also forms the high-speed CAN network topology and at 500 Kbps. It is more difficult to identify devices using signal characteristics sampled at high bit rate than at low bit rate; hence and we assumed that if our method worked successfully at the high-speed CAN, it would also function well at the low-speed CAN bit rate [20]. Fig. 8 shows our experimental setup which includes two types of CAN development boards, an oscilloscope to measure electrical CAN signal, and a monitoring tool for CAN communication.

*Classification algorithm:* We performed the default classification algorithms provided by MatLab R2016a [31]. The classification algorithms we used were Support Vector Machine (SVM), Neural Network (NN), and Bagged Decision Tree (BDT) with default options MatLab provides. We performed the 10-fold cross validation test as a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. In 10-fold cross validation, a given data set was partitioned into ten folds. A classifier was trained with nine folds (training data) and the classification model was tested with the remaining one fold (testing data). This process was repeated ten times so that each fold could be used exactly once as the validation data. We used the averaged classification accuracy as 10-fold cross validation accuracy (i.e., every result which we show in this section is the result of 10-fold cross validation test). We measured 900 electrical CAN signal per ECU.

### B. Basic Identification

Our method examines electrical CAN signals corresponding to the part of extended identifier field which are generated by ECUs. At first, we assign the 18-bit extended identifier to $01010101010101010101_{(2)}$. Using signals identified with the extended identifier field, we extracted statistical features and then performed classification algorithms on the features. Table V is the confusion matrix which allows visualization of the performance of classification algorithms. Each column of the matrix represents the instances in a predicted ECU while each row represents the instances in an actual ECU and each diagonal cell indicates the success rate for that signal or observation to be classified in the valid ECU, with three values representing the rates obtained from three different classification algorithms: SVM with linear kernel, NN with 100 hidden nodes, and BDT with 100 classifiers, in that order. We can see the success rate is enough to correctly identify ECUs as well as the misclassification rate is enough less. Misclassification means that ECUs are incorrectly identified by our method. Table VII summarizes the average misclassification rates of our method and [19], respectively. Our method has approximate 4 times as less misclassification rate compared to [19].

It is noted that because the authors of [19] did not consider Type I adversary model, we compare our method with [19] when it comes to Type II adversary model.

### C. Identifying Known ECUs (for Type II Adversary)

In this subsection, we calculate the average success rates, changing the extended identifier to other bit strings and parameters of classification algorithms. We want to know whether bit strings affect our success rate or not. We performed

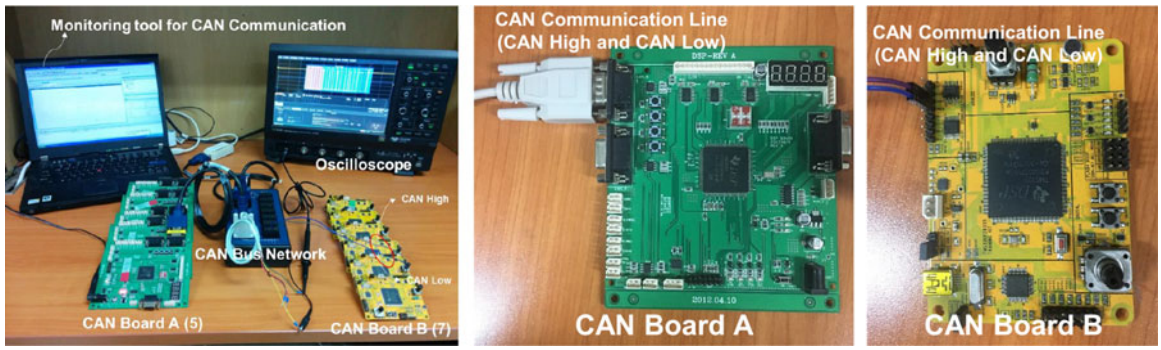



Fig. 8. Experimental setup.

TABLE V
CONFUSION MATRIX FOR IDENTIFYING ECUS WHEN SVM, NN, AND BDT ARE USED, RESPECTIVELY

| | ECU1 | ECU2 | ECU3 | ECU4 | ECU5 | ECU6 | ECU7 | ECU8 | ECU9 | ECU10 | ECU11 | ECU12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ECU1 | **98.22/ 98.33/ 97.78** | 0/ 0/ 0.11 | 0/ 0/ 0 | 0.67/ 0.33/ 0.22 | 1.11/ 1.22/ 1.89 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0.11/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU2 | 0/ 0/ 0 | **95.89/ 95.11/ 95.89** | 0/ 0/ 0 | 4.11/ 4.89/ 4.11 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU3 | 0/ 0/ 0 | 0/ 0/ 0 | **99.33/ 98.78/ 98.67** | 0.67/ 1.00/ 1.33 | 0/ 0.22/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU4 | 0.78/ 0.44/ 0.44 | 6.78/ 4.78/ 6.33 | 0.56/ 1.00/ 0.89 | **91.67/ 93.78/ 91.67** | 0.22/ 0/ 0.67 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU5 | 1.44/ 1.33/ 2.11 | 0/ 0/ 0 | 0.11/ 0.11/ 0.11 | 0/ 0/ 0.78 | **98.33/ 98.44/ 97.00** | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0.11/ 0 | 0.11/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU6 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ ‘0 | 0/ 0/ 0 | 0/ 0/ 0 | **86.33/ 85.00/ 86.33** | 0/ 0/ 0 | 0.44/ 0.56/ 0.44 | 0/ 0/ 0 | 0/ 0/ 0 | 13.22/ 14.44/ 13.22 | 0/ 0/ 0 |
| ECU7 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | **99.78/ 99.78/ 99.56** | 0/ 0/ 0 | 0/ 0/ 0 | 0.22/ 0.22/ 0.44 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU8 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 1.89/ 1.22/ 1.56 | 0/ 0/ 0 | **97.00/ 97.56/ 95.22** | 0/ 0/ 0 | 0/ 0/ 0 | 1.11/ 1.22/ 3.22 | 0/ 0/ 0 |
| ECU9 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | **100/ 99.89/ 99.44** | 0/ 0.11/ 0.56 | 0/ 0/ 0 | 0/ 0/ 0 |
| ECU10 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0.22 | 0/ 0/ 0 | 0/ 0/ 0.11 | **100/ 100/ 99.00** | 0/ 0/ 0 | 0/ 0/ 0.67 |
| ECU11 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0.11/ 0 | 13.22/ 12.44/ 14.22 | 0/ 0/ 0 | 1.89/ 1.67/ 2.78 | 0/ 0/ 0 | 0/ 0/ 0 | **84.89/ 85.78/ 83.00** | 0/ 0/ 0 |
| ECU12 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0 | 0/ 0/ 0.22 | 0/ 0/ 0 | **100/ 100/ 99.78** |

TABLE VI
AVERAGE SUCCESS RATE CORRESPONDING TO DIFFERENT BIT STRING OF EXTENDED IDENTIFER USING THREE DIFFERENT CLASSIFICATION ALGORITHMS: SVM, NN, AND BDT

| EXID | SVM | | NN | | | BDT | | |
|---|---|---|---|---|---|---|---|---|
| | Kernel function | | #Hidden nodes | | | #Classifiers | | |
| | Linear | RBF | 10 | 50 | 100 | 10 | 50 | 100 |
| $0000\ldots0000_{(2)}$ | 94.94 | 91.75 | 95.78 | 96.41 | **96.48** | 95.40 | 95.79 | **95.92** |
| $0101\ldots1010_{(2)}$ | **95.24** | 88.26 | 95.32 | 95.44 | 95.28 | 94.07 | 94.69 | 94.83 |
| $0111\ldots1111_{(2)}$ | 76.50 | 76.59 | 79.51 | 82.71 | 82.13 | 82.25 | 84.20 | 84.71 |

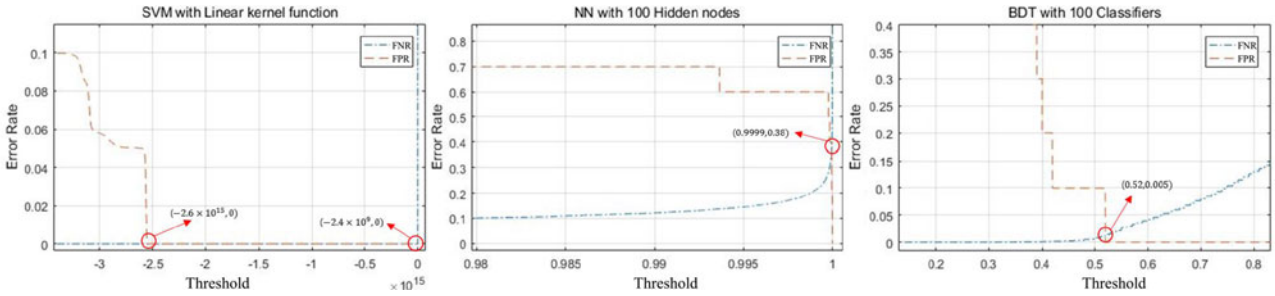



Fig. 9. FN and FP rate with varying threshold.

TABLE VII
THE AVERAGE MISCLASSIFICATION RATE FOR ECU IDENTIFICATION

| | Our method | [19] |
|---|---|---|
| Average misclassification rate (%) | 0.36 | 1.5 |

classification algorithms on electrical CAN signals with additional extended identifier of $00000000000000000_{(2)}$ and $01111111111111111_{(2)}$. Due to the fact that the IDE is 1, we can avoid the case that bit stuffing occurs at different bit locations when we define the most significant bit of the extended identifier as 0. In addition, we performed classification algorithms with typical parameters. Table VI shows the average success rate according to different parameters and different bit strings. We can see that linear kernel function, 100 hidden nodes, and 100 classifiers achieve the best performance of SVM, NN, BDT, respectively. In next subsection, we evaluate our method with only these parameters.

In addition, this result shows that the success rate varies according to the bit string given to the extended identifier field. Our method reaches the highest rate when the extended identifier was $00000000000000000_{(2)}$ and NN with 100 hidden layers was performed. Even though we did not evaluate more bit strings for the identifier field, the success rates for both $00000000000000000_{(2)}$ and $01010101010101010_{(2)}$ are acceptable. As a result, we conclude that we get better result when there are enough dominant states (i.e., 0).

### D. Identifying Unknown ECUs (for Type I Adversary)

In the above subsection, we evaluated our method only considering Type II adversary model. To address a Type I adversary where an attacker attaches a new device to OBD-II port to connect to the in-vehicle CAN network, our method must classify the new device as belonging to the unknown class. As we mentioned in Section IV, identifying unknown ECUs is related to the novelty detection which is the identification of new or unknown data that a machine learning system has not been trained with [29]. We utilized a threshold-based approach which is a convenient and effective extension of our classification-based basic model. By using this approach, we can identify unknown ECUs while preserving our basic classification model. Accordingly, we define a threshold and perform the same classification methods as in the above basic experiment. In the threshold model, we consider that signals whose scores corresponding to known classes are all below the threshold are from the unknown class. The overall success rates of the threshold-based identification are less or equal to ones of the basic identification without a threshold, because the case that classification scores is less than a threshold is classified as an unknown class, even if the result is correct. FN (False Negative) refers to the case that an valid ECU is classified as an alien (i.e., an unknown ECU) and FP (False Positive) refers to the case that an unknown ECU is incorrectly classified as one of the valid ECUs. Fig. 9 shows both FN and FP rates corresponding to varying threshold. When we defined a higher threshold, FN rates went up, while FP rates went down. In case of SVM, the classification score is from negative infinity to 0. The score of 0 in SVM means that test data is perfectly matching to the corresponding class. In the other cases, the classification score is from 0 to 1. For NN and BDT, the score of 1 means that test data is perfectly matching to the corresponding class. We set the threshold depending on the ranges of the classification algorithm scores. Even though SVM did not output the best performance among three different classification algorithms in SubSection V-B, SVM can perfectly distinguish between known ECUs and unknown ECUs by utilizing a threshold of between $-2.6 \times 10^{15}$ and $-2.4 \times 10^{9}$ at which EER (Equal Error Rate) is 0. For NN, EER is 0.38 at a threshold of 0.9999, meaning that it is not available to identify an alien. As a result, we conclude our method outputs the most acceptable result when either SVM with linear kernel function or BDT with 100 classifiers is used.

Both CAN2USB-interface and ELM327 are typical devices that can be connected to the OBD-II port and adversaries can purchase those devices quite inexpensively [23], [24]. Accordingly, we assume that those devices are likely candidates for launching attacks. Using those devices, we sampled more than 1,000 observations or signals per device.

### E. Computational Time

To apply our method, we evaluate the computational time required. As can be seen in Algorithm 1, our method consists of several functions; however, we only evaluate the times required for the important functions: FEATURE_EXTRACTION, GET_CLASSIFIER, and CLASSIFICATION, which mainly affect the computational overhead compared to the other functions. We leveraged the MATLAB codes, which were conducted on an Intel i7 3.5 GHz dual core processor with 16 GB of RAM. In addition, the internal TIC and TOC functions in MATLAB were used, which enables to simply read elapsed times.

TABLE VIII
THE REQUIRED TIME FOR FEATURE EXTRACTION

| Domain | Feature | The required time (ms) |
|---|---|---|
| Time | Mean | 0.11 |
| | Std-Dev | 0.15 |
| | Average Deviation | 0.01 |
| | Skewness | 0.12 |
| | Kurtosis | 0.11 |
| | RMS Amplitude | 0.04 |
| | Lowest Value | 0.005 |
| | Highest Value | 0.005 |
| Frequency | Spec. Centroid | 0.22 |
| | Spec. Entropy | 0.03 |
| | Spec. Spread | 0.01 |
| | Spec. Skewness | 0.05 |
| | Spec. Kurtosis | 0.05 |
| | Spec. Brightness | 0.01 |
| | Spec. Roll off | 0.05 |
| | Spec. Irregularity | 0.03 |
| | Spec. flux | 0.02 |

Table VIII shows the measured times for 17 features extracted from a CAN electrical signal. A fast Fourier transformation (FFT), which is necessary for every frequency-domain feature, was included in the feature of Spec. Centroid.

GET_CLASSIFIER is the function requiring the longest computational time in our method, and depends on the number of training datasets applied. Because 900 training datasets per ECU were used in our experimental setup, we measured the time required for GET_CLASSIFIER using the same datasets. For CLASSIFICATION, we measured the execution time required to classify a set of features extracted from an electrical CAN signal into a class. Table IX shows the required times for GET_CLASSIFIER and CLASSIFICATION.

Although GET_CLASSIFIER requires a significant amount of time, this function can be applied in advance; accordingly, it is not included in the computational overhead of our method through a pre-computation. The computational times for FEATURE_EXTRACTION and CLASSIFICATION should be minimized because our method needs the CAN messages to be analyzed in real time. The execution time required for the extraction of 17 features and classification using a linear SVM is 1.067 ms (1.02 ms + 0.047 ms). According to ISO-11898, CAN supports baud rates from 40 Kbit/s to 1 Mbit/s. Among these baud rates, a 500 Kbit/s CAN network is the most typically used in an automotive environment. To ensure that no message gets lost, the CAN bus load should not be filled to 100% [32]; thus, in-vehicle CAN networks maintain their bus load at around 50% [17], [33]. Considering a 50% bus load in a 500 Kbit/s CAN network, it is assumed that two CAN message are transmitted on average every 1 ms. It should be noted that about 2 $\mu$s is needed to transmit one bit in a 500 KBit/s CAN network. We actually checked the validity of this assumption by measuring the average time interval between two consecutive CAN messages from All New SOUL 2014, where one CAN message is transmitted every 0.478 ms. Because our method requires 1.067 ms to address one CAN message, our method seems insufficient to analyze all CAN messages in real time. However, our method is able to handle this problem in the following two ways. First, it can be designed to analyze only CAN messages related with safety-critical operations (e.g., engine control) and to not analyze less critical CAN messages (e.g., door control). Furthermore, many ECUs are not critical to in-vehicle CAN networks, and the protection of non-critical ECUs can be considered optional [33]. Second, our method can be implemented using multi-thread processing, which enhances the computational performance if multiple processors are supported. As more high-performance devices continue to be installed in new vehicles, we expect that our method will analyze two or three CAN messages every millisecond through multi-thread processing. Thus, we believe that our method enables a secure in-vehicle CAN network in practice.

### *E. Functionality*

In this subsection, we analyze our method when it comes to functionality. We focus on 5 functionalities: Arbitration decision, Aperiodic message, Needed # of messages, Hardware modification, and False identification rate. By the CAN standard, electrical CAN signal can be generated by more than 2 ECUs until the arbitration decision process ends. The signal that is generated by multiple ECUs cannot be used for correct ECU identification. To avoid this part of signal, our method examine the electrical CAN signal corresponding to extended identifier field. In case of [19], they examined the signal corresponding to identifier field; and hence their method is not able to identify ECUs when signal is generated by more than two ECUs. Since our method examines every CAN message and its electrical CAN signal, it is possible to identify ECUs per every CAN message. However, the method of [34] needed multiple CAN messages that the same ECU transmits in order to identify the ECU. It means that identification time of their method is longer than our method because it takes time to collect multiple CAN messages. Furthermore, the method of [34] is only available for periodic CAN messages. According to [17], an adversary inject just six non-periodic CAN messages to kill the engine of the target vehicle. This means that the method of [34] is not able to detect the attack proposed in [17]. Because our method complies with the CAN standard, our method does not require hardware modification of the current ECUs. Finally, false identification rate refers to misclassification rate which is mentioned in Section V-B. Our method has approximate 4 times as less misclassification rate compared to [19]. Table X shows functionality comparison between our method and the existing methods [19], [34]. As a result, we conclude that our method outperforms the other existing methods.

## VI. RELATED WORKS

Fingerprinting is a well-established biometrics techniques for identifying human beings [35], [36]. Beginning in the 20th century, the concept of fingerprinting has been used to identify devices as well. In particular, World War II made it both important and popular to try to identify radar, radios, and other wireless communications [37]. Many researchers have

TABLE IX
THE REQUIRED TIME FOR GET_CLASSIFIER AND CLASSIFICATION

| The required time (ms) | SVM | | NN | | | BDT | | |
|---|---|---|---|---|---|---|---|---|
| | Kernel function | | #Hidden nodes | | | #Classifier | | |
| | Linear | RBF | 10 | 50 | 100 | 10 | 50 | 100 |
| GET_CLASSIFIER | 2,574 | 4,458 | 197 | 285 | 453 | 465 | 2,204 | 4,570 |
| CLASSIFICATION | 0.047 | 0.586 | 0.006 | 0.007 | 0.009 | 0.015 | 0.071 | 0.140 |

TABLE X
FUNCTIONALITY COMPARISON

| Arbitration decision | Our method Considered | [19] Unconsidered | [34] Unconsidered |
|---|---|---|---|
| Needed # of CAN messages | Single | Single | Multiple |
| Aperiodic message | Available | Available | Unavailable |
| Hardware modification | No | No | No |
| False identification rate | Mid (0.36%) | High (1.5%) | Low (0.055%) |

continued to study identification of wireless signals right up to the present [38]–[45]. These methods are usually based upon minor variations in analog hardware in transmitters, utilizing the individual characteristics of the transmitters [46]. Fundamentally, the approach to identifying devices according to their unique characteristic recognizes that physical device are all different in the way they produce signals, even when they are produced by the same manufacture.

Some previous works has identified network devices by using the clock skew of the devices. On a network such as the Internet closck skew describes the difference in time shown by the clocks at different nodes on the network. Accordingly, it is possible to distinguish devices through TCP and ICMP timestamps [47]. Previous research has studied not only hardware-based fingerprinting, but also software-based fingerprinting. Software-based fingerprinting uses patterns traceable to a user. For example, by analyzing month-logs, users can be tracked or identified [48]. Browser history can also be used for tracking or identifying users [49]. Software-based fingerprinting uses a uniqueness of users' patterns such as current configuration of the system (e.g., screen resolution, or list of fonts installed in a system) [50].

Recently, since various smart devices including smartphones are being developed, there has been a substantial amount of work done on smart device fingerprinting. In order to identify smart devices, researchers use either smart device sensor or unique personalized configurations [25], [51]–[57]. The Sensors that work in smart devices can be divided into 3 groups: camera sensors, acoustic sensors, and motion sensors. Based on the camera sensors's unique pattern noise, it is possible to distinguish images for the same scene because the images include unique differences because camera sensors are all different from one another [56], [57]. The embedded acoustic components of smartphones have also been used to identify smartphones [51], [52]. Moreover, different microphones convert the same sound source into different electronic signals, or conversely loudspeakers convert the same audio signals into different sounds in the air. By using the unique characteristics of each device, it is possible to identify microphones and loudspeakers when analyzed separately or together.

Another approach to identifying sensors in smartphones is to use motion sensors such as accelerometer and gyrometer [25], [51], [53], [54]. Accelerometers measure 3-axis acceleration and regulate, for example, smart phone screen rotation. For the same movement, the measured values are a little bit different in each accelerometer; and hence identification of smart devices is available. Furthermore, accelerometers can use gravitational acceleration (i.e., 9.8 m/s$^2$) that is always detectable, which means that an artificial same source is not needed such as the same image or the same sound.

Similar to our method, there are methods to identify ECUs in the in-vehicle CAN network [19], [34]. The authors of [19] proposed a method for source identification using signal characteristics in the CAN network. They used the matched filter (known as a North filter) to compute the correlation between a target signal and a reference signal and thereby identify the devices. Their work was the first approach to identify ECUs using inimitable characteristics of signals. The authors of [34] proposed another method to identify ECUs in the in-vehicle CAN network. They used unique delayed time (clock skew) when ECUs transmit CAN messages. Because there is no the field for timestamp information in the CAN frame format, their method is only able to identify ECUs by using periodic CAN messages. This means that an adversary who transmits aperiodic CAN messages is not detected by their method.

## VII. LIMITATION AND FUTURE WORKS

In this section, we discuss limitations of our method and future work to improve the method. Our method uses the extended frame format instead of base frame format. Even though this frame format is part of the standard CAN protocol, the in-vehicle networks currently function solely with the base frame format. Thus, our method can be used without changing ECU hardware, but ECU firmware would need to be updated in order to use the extended frame format. This firmware update would involve new programming in new vehicles or reprogramming in older vehicles, but again, replacement of existing ECU hardware is not required. If it is possible to get enough long fixed bit string from base frame format not extended frame format, we do not have to update existing firmware. As a result, we plan to do future work to develop a method without using the extended frame format. In addition, our plans for future works contain more

extended modeling and analysis. For example, in the current work of the paper, the complexity issue was insufficient, and we are planning to take into account such an issue as well with more extended model and analysis.

## VIII. Conclusion

We have presented our new method to identify ECUs based on inimitable characteristics of signals emitted from different ECUs. Since our method only requires installation of a monitoring unit to the in-vehicle CAN network, without any change in ECU hardware, our method can be directly and transparently applied into the current system. In addition, our method adequately simulates the CAN standard and is compatible with it. Thus, we have minimized the required cost of applying this security method to the existing in-vehicle CAN network. We identified the limitations in the work of [19], improving it to be effective in a real environment. Our method is more than twice as accurate as the previous research of [19], as reflected in the low false positive rate. Moreover, we tested our method under the same environment as the actual in-vehicle CAN network. As a result, we conclude our proposed method is a feasible and effective approach that can realistically be applied into in-vehicle CAN network.

## References


[1] J. G. Kassakian, H.-C. Wolf, J. M. Miller, and C. J. Hurton, "Automotive electrical systems circa 2005," *IEEE Spectr.*, vol. 33, no. 8, pp. 22–27, Aug. 1996.

[2] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 447–462.

[3] *CAN Specification Version 2.0*, Robert Bosch GmbH Std., 1991.

[4] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. USENIX Security Symp*. San Francisco, CA, USA, 2011. [Online]. Available: http://static.usenix.org/events/sec11/tech/full_papers/Checkoway.pdf

[5] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," Black Hat USA, 2015.

[6] R. Akhtar, S. Leng, I. Memon, M. Ali, and L. Zhang, "Architecture of hybrid mobile social networks for efficient content delivery," *Wireless Pers. Commun.*, vol. 80, no. 1, pp. 85–96, 2015.

[7] H. J. Jo, W. Choi, S. Y. Na, S. Woo, and D. H. Lee, "Vulnerabilities of android OS-based telematics system," *Wireless Pers. Commun.*, vol. 92, pp. 1511–1530, 2017. [Online]. Available: https://link.springer.com/article/10.1007/s11277-016-3618-9

[8] I. Memon, "Authentication users privacy: An integrating location privacy protection algorithm for secure moving objects in location based services," *Wireless Pers. Commun.*, vol. 82, no. 3, pp. 1585–1600, 2015.

[9] I. Memon and Q. A. Arain, "Dynamic path privacy protection framework for continuous query service over road networks," *World Wide Web*, vol. 20, pp. 639–672, 2017. [Online]. Available: https://link.springer.com/article/10.1007/s11280-016-0403-3

[10] I. Memon, L. Chen, A. Majid, M. Lv, I. Hussain, and G. Chen, "Travel recommendation using geo-tagged photos in social media for tourist," *Wireless Pers. Commun.*, vol. 80, no. 4, pp. 1347–1362, 2015.

[11] M. H. Memon, J.-P. Li, I. Memon, and Q. A. Arain, "Geo matching regions: Multiple regions of interests using content based image retrieval based on relative locations," *Multimedia Tools Appl.*, vol. 76, pp. 15377–15411, 2017.

[12] I. Memon, Q. A. Arain, H. Memon, and F. A. Mangi, "Efficient user based authentication protocol for location based services discovery over road networks," *Wireless Pers. Commun.*, vol. 95, no. 4, pp. 3713–3732, 2017.

[13] B. Groza and S. Murvay, "Efficient protocols for secure broadcast in controller area networks," *IEEE Trans. Ind. Inf.,* vol. 9, no. 4, pp. 2034–2042, Nov. 2013.

[14] C.-W. Lin and A. Sangiovanni-Vincentelli, "Cyber-security for the controller area network (CAN) communication protocol," in *Proc. Cyber Security (CyberSecurity),* Dec. 2012, pp. 1–7.

[15] A.-I. Radu and F. D. Garcia, "LeiA: A lightweight authentication protocol for can," in *Proc. Int. Conf. Eur. Symp. Res. Comput. Security*, 2016, pp. 283–300.

[16] A. Van Herrewege, D. Singelee, and I. Verbauwhede, "Canauth-a simple, backward compatible broadcast authentication protocol for CAN bus," in *Proc. ECRYPT Workshop Lightweight Cryptography*, vol. 2011, 2011. [Online]. Available: https://www.esat.kuleuven.be/cosic/publications/article-2086.pdf

[17] S. Woo, H. J. Jo, and D. H. Lee, "A practical wireless attack on the connected car and security protocol for in-vehicle can," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 993–1006, Apr. 2015.

[18] T. Ziermann, S. Wildermann, and J. Teich, "Can+: A new backward-compatible controller area network (CAN) protocol with up to 16× higher data rates." in *Proc. IEEE Des., Autom. Test Eur. Conf. Exhib.*, 2009, pp. 1088–1093.

[19] P.-S. Murvay and B. Groza, "Source identification using signal characteristics in controller area networks," *IEEE Signal Process. Lett.* vol. 21, no. 4, pp. 395–399, Apr. 2014.

[20] R. M. Gerdes, M. Mina, S. F. Russell, and T. E. Daniels, "Physical-layer identification of wired ethernet devices," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 4, pp. 1339–1353, Aug. 2012.

[21] M. Di Natale, "Understanding and using the controller area network," New York, NY, USA: Springer, 2008, [Online]. Available: Inst. Eecs. Berkeley. Edu/~ ee249/fa08/Lectures/handout_canbus2. pdf

[22] D. Adu-Gyamfi, Y. Wang, F. Zhang, M. K. Domenic, I. Memon, and Y. H. Gustav, "Modeling the spreading behavior of passive worms in mobile social networks," in *Proc. IEEE 6th Int. Conf. Inf. Manag., Innovation Manag. Ind. Eng.,* 2013, vol. 1, pp. 380–383.

[23] "Elm327," [Online]. Available: http://www.iobd2.org/elm327-iphone/. Accessed on: Mar. 9, 2017.

[24] "Obd2usb," [Online]. Availble: http://www.ebay.com/bhp/obd2-usb. Accessed on: Mar. 9, 2017.

[25] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "Accelprint: Imperfections of accelerometers make smartphones trackable," in *Proc. 21st Annu. Netw. Distrib. Syst. Security Symp.,* San Diego, CA, USA, Feb. 23–26, 2014. [Online]. Available: https://pdfs.semanticscholar.org/38cb/8060ad77cc4b50e7701592d273a8f65cdb5b.pd

[26] "Libxtract: Feature extraction library documentation," 2007. [Online]. Available: http://libxtract.sourceforge.net. Accessed on: Mar. 9, 2017.

[27] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 27–66, 2012.

[28] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Comput. Security*, vol. 28, no. 1, pp. 18–28, 2009.

[29] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[30] T. Nolte, H. Hansson, and L. Bello, "Automotive communications-past, current and future," in *Proc. 10th IEEE Conf. Emer. Technol. Factory Autom.,* Sep. 2005, vol. 1, pp. 8–992.

[31] MATLAB,*Version 8.5.0 (R2015a)*. Natick, MA, USA: The MathWorks Inc., 2015.

[32] F. Yang, "A bus off case of CAN error passive transmitter," EDN Tech. Paper, 2009. [Online]. Available: https://www.researchgate.net/profile/Fuyu_Yang2/publication/265825188_A_Bus_off_Case_of_CAN_Error_Passive_Transmitter/links/5760b81708ae227f4a3f2c82.pdf

[33] S. Nürnberger and C. Rossow, "–vatican–vetted, authenticated can bus," in *Proc. Int. Conf. Cryptographic Hardware Embed. Syst.*, 2016, pp. 106–124.

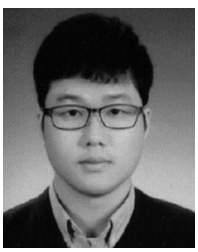
[34] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. 25th USENIX Security Symp.,* USENIX Association, Berkeley, CA, USA, 2016, pp. 911–927.

[35] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern Recognit. Lett.*, vol. 24, no. 13, pp. 2115–2125, Sep. 2003. [Online]. Available: http://dx.doi.org/10.1016/S0167-8655(03)00079-5

[36] P. Tuyls and J. Goseling, "Capacity and examples of template-protecting biometric authentication systems," in *Biometric Authentication*, (Lecture Notes in Computer Science). vol. 3087. D. Maltoni and A. Jain, Eds. Berlin, Germany: Springer, 2004, pp. 158–170. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-25976-3_15

[37] R. V. Jones, *Most Secret War*. London, U.K: Penguin, 2009.

[38] M. Barbeau, J. Hall, and E. Kranakis, "Intrusion detection and radio frequency fingerprinting in mobile and wireless networks," School of Comput. Sci., Carleton Univ., Ottawa, ON, Canada, Tech. Rep., 2003.
[39] K. Ellis and N. Serinken, "Characteristics of radio transmitter fingerprints," *Radio Sci.*, vol. 36, no. 4, pp. 585–597, 2001.
[40] P. J. Ferrell, "Method and apparatus for characterizing a radio transmitter," U.S. Patent 5,005,210, Apr. 2, 1991.
[41] M. B. Frederick, "Cellular telephone anti-fraud system," U.S. Patent 5,448,760, Sep. 5, 1995.
[42] J. Hall, M. Barbeau, and E. Kranakis, "Detection of transient in radio frequency fingerprinting using signal phase," presented at the Intl. Conf. Wireless Opt. Commun., 2003.
[43] D. L. Mensa, A. C. Bati, K. Vaccaro, L. D. To, R. E. Dezellem, and D. P. Wirtz, "Radar signature evaluation apparatus," U.S. Patent 6,529,157, Mar. 4, 2003.
[44] O. Ureten and N. Serinken, "Detection of radio transmitter turn-on transients," *Electron. Lett.*, vol. 35, no. 23, pp. 1996–1997, 1999.
[45] Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device fingerprinting in wireless networks: Challenges and opportunities," *IEEE Commun. SurveysTuts*, vol. 18, no. 1, pp. 94–104, First Quarter 2016.
[46] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw.*, 2008, pp. 116–127.
[47] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Trans. Depend. Secure Comput.,* vol. 2, no. 2, pp. 93–108, Apr.–Jun. 2005.
[48] T.-F. Yen, Y. Xie, F. Yu, R. P. Yu, and M. Abadi, "Host fingerprinting and tracking on the web: Privacy and security implications," in *Proc. 19th Annu. Netw. Distrib. Syst. Security Symp.*, 2012. [Online]. Available: https://pdfs.semanticscholar.org/0557/5e85d4b0c09c09552921b2ee0db79e5e9cf9.pdf
[49] L. Olejnik, C. Castelluccia, and A. Janc, "Why johnny can't browse in peace: On the uniqueness of web browsing history patterns," in *Proc. 5th Workshop Hot Topics Privacy Enhanc. Technol.*, 2012.
[50] G. Acar *et al.*, "Fpdetective: Dusting the web for fingerprinters," in *Proc. 2013 ACM SIGSAC Conf. Comput. Commun. Security.*, 2013, pp. 1129–1140.
[51] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," arXiv:1408.1416, 2014.
[52] A. Das, N. Borisov, and M. Caesar, "Do you hear what i hear?: Fingerprinting smart devices through embedded acoustic components," in *Proc. 2014 ACM SIGSAC Conf. Comput. Commun. Security.*, 2014, pp. 441–452.
[53] A. Das, N. Borisov, and M. Caesar, "Tracking mobile web users through motion sensors: Attacks and defenses," in *Proc. 23rd Annu. Netw. Distrib. Syst. Security Symp.*, 2016. [Online]. Available: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/tracking-mobile-web-users-through-motion-sensors-attacks-defenses.pdf
[54] A. Das, N. Borisov, E. Chou, and M. H. Mughees, "Smartphone fingerprinting via motion sensors: Analyzing feasibility at large-scale and studying real usage patterns," arXiv:1605.08763, 2016.
[55] A. Kurtz, H. Gascon, T. Becker, K. Rieck, and F. Freiling, "Fingerprinting mobile devices using personalized configurations," *Proc. Privacy Enhanc. Technol.*, vol. 2016, no. 1, pp. 4–19, 2016.
[56] C.-T. Li, "Source camera identification using enhanced sensor pattern noise," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 2, pp. 280–287, Jun. 2010.
[57] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Trans. Inf. Forensics Security,* vol. 1, no. 2, pp. 205–214, Jun. 2006.



**Wonsuk Choi** received the B.S. degree in mathematics from the University of Seoul, Seoul, South Korea, in 2008, and the M.S. degree in information security from Korea University, Seoul, South Korea in 2013. He is currently working toward the Ph.D. degree in information security at the Graduate school of Information Security, Korea University. His research interests include security for body area network, usable security, applied cryptography, and smart car security.

**Hyo Jin Jo** received the B.S. degree in industrial engineering and the Ph.D. degree in information security from Korea University, Seoul, South Korea, in 2009 and 2016, respectively. He is currently a Postdoctoral Researcher with the Department of Computer and Information System, University of Pennsylvania, Philadelphia, PA, USA. His research interests include cryptographic protocols in authentication, applied cryptography, security and privacy in ad hoc networks, and smart car security.

**Samuel Woo** received the Ph.D. degree in information security from Korea University, Seoul, South Korea, in 2016. He was with the Electronics and Telecommunications Research Institute (ETRI) in 2016. He is currently a Senior Researcher with the Network Security Research Laboratory, ETRI. His research interests include cryptographic protocols in authentication, applied cryptography, security and privacy in vehicular networks, and moving target defense.

**Ji Young Chun** received the B.S. degree from the Department of Mathematics, Ewha Womans University, Seoul, South Korea, in 1997. She received the M.S. and Ph.D. degrees from the Graduate School of Information Management and Security, Korea University, Seoul, South Korea, in 2006 and 2011, respectively. She is currently a Research Professor with the Graduate School of Information Security, Korea University, Seoul, South Korea. Her research interests include cryptographic protocols, database security, and RFID/USN security.

**Jooyoung Park** received the B.S. degree in electrical engineering from Seoul National University, Seoul, South korea, in 1983 and the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin, Austin, TX, USA, in 1992. In 1993, he was with Korea University, Seoul, South korea, where he is currently a Professor with the Department of Control and Instrumentation Engineering. His research interests include machine learning and control theory.

**Dong Hoon Lee** received the B.S. degree from Korea University, Seoul, South Korea, in 1985 and the M.S. and Ph.D. degrees in computer science from the University of Oklahoma, Norman, OK, USA, in 1988 and 1992, respectively. Since 1993, he has been with the Faculty of Computer Science and Information Security, Korea University. He is currently a Professor and the Director with the Graduate School of Information Security, Korea University. His research interests include cryptographic protocol, applied cryptography, functional encryption, software protection, mobile security, vehicle security, and ubiquitous sensor network security.