IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 4, 2018, accepted December 25, 2018, date of publication February 1, 2019, date of current version February 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2892961

CAN ID Shuffling Technique (CIST): Moving Target Defense Strategy for Protecting In-Vehicle CAN

SAMUEL WOO[1], DAESUNG MOON[1], TAEK-YOUNG YOUN[1], YOUSIK LEE[2], AND YONGEUN KIM[3]

[1]Information Security Research Division, Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea
[2]ESCRYPT GmbH, Gyeonggi 13488, South Korea
[3]Korea Automotive Technology Institute, Cheonan 31214, South Korea

Corresponding author: Daesung Moon (daesung@etri.re.kr)

This work was supported by Institute for Information and communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-00213, Development of Cyber Self Mutation Technologies for Proactive Cyber Defense).

ABSTRACT New vehicles have become increasingly targeted for cyber-attacks as their rate of digitalization is accelerated. Research on vehicle hacking has highlighted the security vulnerabilities of in-vehicle controller area networks (CANs) as the biggest problem. In particular, a CAN does not offer access control, authentication, or confidentiality, so it fails to prevent reconnaissance operations conducted by an adversary. Because its static configuration (CAN ID, data frame transmission cycle, and data field format) is used in an in-vehicle network environment, the adversary can conduct reconnaissance and easily acquire information to be used for an attack. One of the moving target defense strategies, network address shuffling (NAS), is an extremely practical security solution that can prevent in-vehicle CAN reconnaissance acts. In this paper, we propose a CAN ID shuffling technique using NAS. Our proposed security solution aims to increase the cost burden for the adversary to analyze CAN data frames. To evaluate the performance of the proposed security solution, we conducted an evaluation based on a labcar. Our proposed security solution may be implemented without altering the unique characteristics of the CAN standard. Hence, it can be used as a practical countermeasure to solve the problems affecting in-vehicle CANs.

INDEX TERMS Controller area network, in-vehicle network security, moving target defense, network address shuffling, vehicular cyber kill chain.

I. INTRODUCTION

Along with the development of Vehicle-Information and Communications Technology (ICT) convergence, various types of Electrical and Electronic (E/E) systems are being installed into vehicles. As the number of E/E systems installed into a vehicle increases, the complexity of the in-vehicle network increases [1], [2]. Robert Bosch GmbH developed the Controller Area Network (CAN) to construct an efficient in-vehicle network [3]. The CAN protocol allowed vehicle manufacturers to reduce the complexity of in-vehicle network wiring [4]. However, the CAN was designed only for a very closed network environment and, in this sense, presents serious security flaws. M.Wolf et al. pointed out that CAN vulnerabilities will cause the various threats in the ICT environment to be transferred to the vehicle environment [2]. The vulnerabilities of in-vehicle CAN are as follows [5].

- No Confidentiality: Data are transmitted in plaintext.
- No Authentication: Authentication for data and entity is not performed.
- Weak access control: Every entity that accesses a communication line can participate in communication.

Previous research has simulated practical cyber-attacks, such as the replay attack and impersonation attack, on an in-vehicle CAN [5] and determined that the vulnerabilities of entity and data authentication are the most serious problem [6]–[8]. Over the past decade, studies have been conducted to solve the problems of CAN [9]–[11]. However, the countermeasures suggested in previous works have demonstrated significant limitations [12].

1) The CAN data frame is too small to use the message authentication code (MAC). Additional transmission of a data frame for MAC causes authentication delays and increased bus load [5], [9], [11].

2) A security protocol that transmits MAC using the CRC field cannot be applied to the standard CAN [5].

This is why vehicle manufacturers have failed to completely solve the problems associated with in-vehicle CANs. If the basic vulnerabilities of in-vehicle CANs are not resolved, the number of vehicular hacking cases will increase in the future. Another reason why vehicles are vulnerable to hacking lies in system update methodology. In general, it is rare that hardware and software installed onto a vehicle are changed frequently. If the system remains in a static state for a significant period of time, an adversary would be given ample time to analyze the vulnerabilities of the system. In particular, the in-vehicle CAN environment uses a static configuration (CAN ID, transmission cycle of data frame, and data field format), so the adversary can easily obtain information for a cyber-attack through a reconnaissance act. As such, the adversary dominates in this asymmetric condition, which makes it very difficult to entirely defend a crucial system.

Recently, Moving Target Defense (MTD) was suggested to overcome the asymmetric relation between cyber-attacks and defense [13], [14]. MTD is an active defense strategy that moves the main attributes (Network address, Protocol, Platform, OS and so forth) of a target (victim) system in order to incapacitate a cyber-attack [15]. In the preparatory phase of a cyber-attack, the adversary performs reconnaissance act to acquire vulnerability information on the target system. The cost of the preparatory phase, including the reconnaissance act, accounts for 95% of the total cost required to perform cyber-attacks [16]. One of the MTD strategies, Network Address Shuffling (NAS) is a practical security solution that makes it difficult for an adversary to find targets by dynamically shuffling (moving) the network attack surface, which includes IP, MAC, and open ports [17]–[19].

In this paper, we propose a CAN ID shuffling technique (CIST) using the NAS. CAN is a sender-ID-based multi-master serial bus system. Therefore, in CAN, the sender ID is the attack surface. A reconnaissance act may be incapacitated by dynamically shuffling the sender ID. CIST can greatly increase the cost to accomplish a reconnaissance act. In order to apply directly to the modern vehicle, CIST is designed without changing the intrinsic properties of the CAN standard. The main contributions of this paper are summarized as follows:

•**Contributions**

1) Proposal of CAN ID shuffling technique (CIST) for secure in-vehicle CAN. CIST changes a sender ID whenever an ECU transmits a data frame using one-time ID. One-time ID is generated using a one-way hash function with a group session key and counter value. Hence, the adversary cannot infer and reuse one-time ID. If CIST used, it is possible to block a reconnaissance act by the adversary and prevent a replay attack and an impersonation attack at the same time.
2) Allocation of CAN ID that prevents duplication of the CAN ID. ECUs belonging to the same subnetwork should not use the same CAN ID at the same time. CIST newly defines CAN ID as PID (Priority ID) and DID (Dynamic ID) in order to ensure ID collision prevention.
3) Allocation of CAN ID that ensures the priority of predefined data frame transmission. In the CAN, the data frame transmission priority is determined by sender ID. Therefore, even if the CAN ID is dynamically changed, the priority of data frame transmission should not be changed. CIST use the PID in order to ensure the priority of data frame transmission.

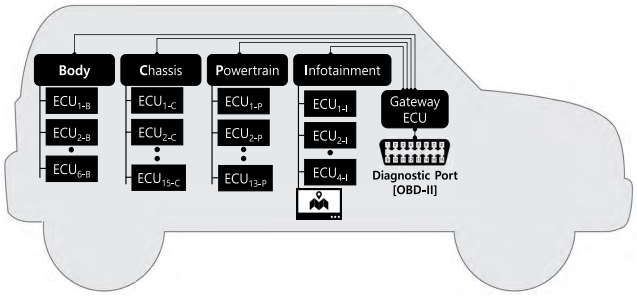


FIGURE 1. Conceptual Model for In-vehicle Network Environment.

This paper is organized as follows: Section II, we introduce the background of our work and the main concepts used. In section III summarizes the relevant previous research. The section IV describes the adversary model, and section V presents the details of the proposed CAN ID shuffling technique (CIST). Section VI describe the security and performance analysis of CIST.

II. BACKGROUND

A. IN-VEHICLE NETWORK AND CONTROLLER AREA NETWORK

The E/E system of a vehicle is composed of an ECU, a sensor, and an actuator. More than one E/E system constructs comprises the main subsystems. Representative main subsystems include the powertrain, chassis, body, and infotainment [20]. ECUs that belong to the subsystem comprise the independent subnetwork. A gateway ECU performs a router function in physically separated subnetworks. Fig. 1 shows a conceptual model of the in-vehicle network environment. In general, the in-vehicle networks use communication protocols such as CAN, CAN with Flexible Data-rate (FD), FlexRay, Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST), and Ethernet.

The CAN is the most representative network protocol for the in-vehicle network [21]. The CAN is a sender-ID-based multi-master broadcast bus system. It transmits a data frame using carrier-sense multiple access with a collision avoidance (CSMA/CA) technique [3], [22]. CAN is divided into two modes based on the length of the ID (arbitration) field. Fig. 2 shows the data frame format of CAN 2.0A and 2.0B. The ID field is changed dynamically for use at higher layer protocols such as J1939 [23]. CAN ID is used to determine the priority for data frame transmission in an arbitration process.

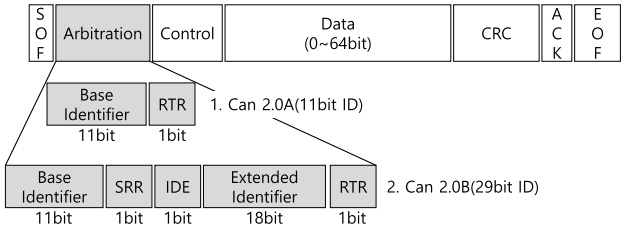


FIGURE 2. CAN data frame format.

The ID field of CAN 2.0B is 32 bits and may be divided into five parts (Base ID, Extended ID, SRR, RTR, IDE). The IDE bit determines whether to use an Extended ID or not. The ID bit (Base ID and Extended ID) should be unique at the single CAN bus. More than two nodes shall not use the same ID bit at the single CAN bus. Fig. 3 shows an example of an arbitration process performed by three nodes. Node A having the lowest value of ID bit acquired a priority for data transmission.

B. MOVING TARGET DEFENSE AND NETWORK ADDRESS SHUFFLING

The current ICT systems are built to operate in a static configuration (e.g. Network address, Protocol, Platform, OS, and etc). Therefore, an adversary could acquire time and information that is secure enough to analyze the vulnerabilities of a target (victim) system [17]. As such, the adversary dominates in this asymmetric condition, making it very difficult to entirely defend a crucial system. MTD is active security strategy that reverses this asymmetric condition between a cyber-attack and a defense. It prevents cyber-attacks by moving the main attributes (e.g., Network address, Protocol, Platform, OS, and etc.) of a crucial system. Gui-lin Cai et al. classified existing MTD strategies into three categories [13].

- Software transformations (ST): A technique that changes the structure and behavior of certain functions composing software in various ways.
- Dynamic Platform techniques (DPT): A technique that dynamically changes an attribute of a computing platform.
- Network Address Shuffling (NAS): A technique that dynamically changes network addresses (e.g. IP, Port, and MAC).

NAS is the technology used to prevent a reconnaissance act by moving the network address. [18]. It defines the network address as the attack surface [19]. NAS may be divided into two modes based on the frequency at which the network addresses are shuffled.

1) Periodic Shuffling: Periodic shuffling refers to a mode where hosts shuffle their network addresses at a fixed time.
2) Non-periodic Shuffling: Non-periodic shuffling refers to a mode where the MTD controller requests hosts to shuffle addresses when an attack is detected or when a certain event occurs.

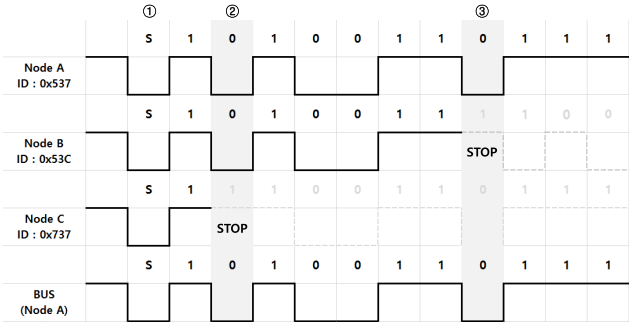


FIGURE 3. Example of an arbitration process: (1) Node A, B, and C start arbitration, (2) Node C loses the bus access, (3) Node B loses the bus access. Node A acquired a priority for data transmission.

The strongest NAS scheme uses a one-time address. Most NAS technologies allow MTD controller to decide when to shuffle the network address but the host itself can assume the role of the MTD controller if the one-time address is used. However, in a general ICT environment, changing the network address of the host to a highest frequency may cause severe performance degradation [18]. Hence, most NAS schemes use a periodic (using low frequency) shuffling mode with a non-periodic shuffling mode. Vehicular ECUs have very low computing power. For this reason, ST or DPT cannot be applied to an in-vehicle network. NAS is a very effective defense method that makes it difficult to perform the first step of cyber kill chain by changing the in-vehicle network properties. In this paper, we propose a method of using a one-time address with low performance degradation of the in-vehicle network. Our proposed scheme uses one-time ID (one-time address) so the ECU itself plays the role of MTD controller with the attack surface shuffling whenever transmitting a data frame.

C. CYBER KILL CHAIN

Lockheed Martin Corporation suggested a general procedure to be applied to cyber-attacks [24]. They analyzed system intrusion phases from the perspective of an adversary and defined them as the cyber kill chain. It consists of seven phases: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command and Control, and Actions on objectives. Because the cyber kill chain consists of a series of phases, an attack fails if a certain phase is broken.

1) Reconnaissance: An adversary investigates, identifies, and selects an objective or a target.
2) Weaponization: An adversary prepares a cyber weapon to attack an objective after selecting it from the reconnaissance step.
3) Delivery: The prepared cyber weapon is delivered to an objective.
4) Exploitation: Malicious code is operated after delivery of the cyber weapon to the target system.
5) Installation: An adversary installs a trojan horse, a backdoor to the target system, and creates an environment in order for them to work there for some time.

6) Command and Control: An adversary constructs a channel to control a target system from the outside.
7) Actions on objectives: An adversary achieves an intended goal.

III. RELATED WORK

A. RESEARCH ON THE VEHICULAR HACKING

K. Koscher et al. was the first to present research findings on vehicular hacking using a real vehicle [6]. They analyzed the vulnerabilities of the in-vehicle CAN and performed an experiment forcibly controlling certain functions of a vehicle. Their experiments have shown that replay attacks can easily be performed on the in-vehicle CAN. In addition, they demonstrated that a remote attack can be successful if the vulnerabilities of a telematics ECU and a wireless communication protocol are exploited [27]. Their experimental findings may be the first case where a commercialized connected car service was hacked. Based on findings of vehicular hacking published by Koscher *et al.* there have been various types of vehicular hacking-related studies.

Koscher *et al.* conducted research similar to that of [6]. They published very detailed technical documents on technological contents related to vehicular hacking. Findings by C. Miller and C. Valasek show that the latest vehicles are not secured from hacking any more.

- In [7], they published a technical document analyzing the vulnerabilities of ECU and in-vehicle CAN and explain how to hack a vehicle in a wired environment.
- In [31], they presented findings from analysis on automobile models vulnerable to hacking based on a variety of documents published by auto manufacturers.
- In [8], they published research findings from hacking into the commercial connected car service.

Woo *et al.* [5] proposed a practical wireless hacking model using a malicious smartphone application and demonstrated an experiment using real vehicles. They showed that if a driver used a malicious self-diagnostic application, the vehicle could be exposed to a very threatening attack. Their experiment shows four types of vehicle hacking: distortion of the dash board, engine stop, handle control, and acceleration. Furthermore, they published findings from hacking into Android OS-based Audio, Visual, and Navigation (AVN) systems [32]. They analyzed the vulnerability of the AVN firmware that uses the Android OS, and developed a method to install malicious firmware. They performed an experiment to track the location of a vehicle using the vulnerabilities of Android OS for vehicles.

B. RESEARCH ON SECURITY SOLUTIONS FOR IN-VEHICLE CAN

ALONG with vehicular hacking research, many studies have been conducted on in-vehicle CAN security. The existing research published for the last 5 years may be classified as follows.

1) Data frame authentication techniques for preventing an impersonation attack and a replay attack.
2) CAN ID hopping techniques for preventing a Denial-of-Service (DoS) attack and replay attack.
3) Intrusion detection system (IDS) for detecting an impersonation attack and a replay attack.

IDS and CIST differ in their purpose and function. This subsection presents previous studies except for those related to IDS. In section VI, we describe the results of the comparative evaluation between the existing studies explained in this subsection and CIST.

- **Truncated MAC:** Woo et al. proposed a CAN data frame authentication technique using truncated MAC [5]. Extended ID field and CRC field were used to transmit 32 bit truncated MAC. **Truncated MAC** generates no additional data frame so there is no increase in bus load. However, it is not possible to use the CRC field for data frame authentication. Their research suggests the need for a new CAN standard with an extended data frame for data frame authentication. In fact, the CAN with Flexible Data-rate (FD) standard that solved limitations of CAN was presented in 2015. In the CAN-FD, a data field was extended to 64 byte. It allowed us to use a variety of data frame authentication techniques. However, there is no commercial vehicle using the CAN-FD because it is a new communication standard. In other words, modern vehicles still use CAN. Accordingly, a security scheme for CAN is indispensable.
- **Mini-MAC:** Jackson et al. proposed a CAN data frame authentication technique using MAC [11]. **Mini-MAC** uses part of a data field to transmit MAC. They analyzed a CAN data frame while driving a Toyota Prius. According to their analytic findings, approximately 60% of the total data frames for the analysis used for the data field was under 4 bytes. They suggested a technique to use the remaining 8 bytes of unused data field for MAC transmission. However, their analysis was performed for a certain vehicle, so it does not necessarily apply to situations for all vehicles. In addition, as shown from their analytic findings, the data field was fully utilized for approximately 35% of the data frames. That is, their proposed scheme cannot be applied to every vehicle and every ECU. In addition, their technique has the potential to increase bus load.
- **ID-Hopping:** Abdulmalik et al. proposed a CAN ID Hopping scheme using an offset generated by a trusted party [33]. When there is a DoS attack on certain subnetwork, the trusted party generates an offset value and delivers it to every ECU that belongs to the subnetwork. The ECUs that received offset values generate new IDs by adding an offset value to their ID value. This technique is not perfectly effective for defending against DoS. As they described in the assumption, the adversary uses a lower ID than te target ECU. In case a new ID is generated by adding an offset, a value larger than that of the adversary is used for the ID. In addition, the ID may be changed using an offset, but the adversary can analyze an offset value easily. A shuffled ID may be determined easily by monitoring the data field of data frames transmitted by ECUs. In their proposed attack model, a malicious ECU can monitor data frames exchanged by ECUs.

• **ID-obfuscation:** Martin et al. proposed an ID obfuscation scheme where vehicles from the same model use a different CAN ID system to minimize the expansion of cyber-attacks [34]. However, a CAN ID used for an attack may be obtained easily each vehicle is analyzed. In particular, the meaning of a data frame may be analyzed with ease if using characteristics of the data field rather than those of CAN ID; this causes their ID-obfuscation scheme to be ineffective. Therefore, it is difficult for their proposed scheme to completely defend against cyber-attacks on an in-vehicle CAN.

IV. ADVERSARY MODEL AND SECURITY REQUIREMENT

In general, research on vehicle hacking is being conducted in three major fields.

(A) Hacking into a vehicle's E/E system (ECU forced actuation attack)
(B) Hacking into a vehicle's smart key (smart key copy)
(C) Hacking into a vehicle's sensor (sensor malfunction: Rader, Rider, and Tire Pressure Monitoring System)

The research field (A) related with hacking results using a vulnerabilities of in-vehicle CAN. In this section, We define an adversary model based on research field (A). First, we analyze the characteristics and vulnerabilities of modern vehicles. Second, we analyze representative research findings on hacking into the E/E system of a vehicle. Based on the analysis results, we illustrate the vehicular cyber kill chain of our adversary model. In this paper, we do not consider the case (B) and (C).

A. CHARACTERISTICS AND VULNERABILITIES OF MODERN VEHICLE

The E/E system of a vehicle leads to various types of security issues. We presented analytic findings on the characteristics and vulnerabilities of a modern vehicle in our previous work. The research findings are listed as follows [12].

1) The CAN protocol has no information security function. An adversary can conduct attacks including sniffing, injection, and replay of a data frame to in-vehicle CAN [2], [5], [6].
2) A diagnostic tool is used to diagnose the E/E system of vehicle. A data frame that can control a certain ECU compulsorily is stored in the vehicle diagnostic tool. Various types of control commands may be acquired using a vehicle diagnostic tool [5], [6].
3) A vehicle owner may install a third-party device to his/her vehicle in addition to a manufacturer-provided original device. An adversary can manufacture and distribute the malicious code-loaded third-party device (Malicious ECU). A malicious ECU can inject a malicious data frame into in-vehicle CAN.
4) There are more than 200 different vehicular applications being sold on Google Play Store (As of July, 2018). An adversary can manufacture and distribute malicious apps. The malicious apps can inject a malicious data frame into the in-vehicle CAN.

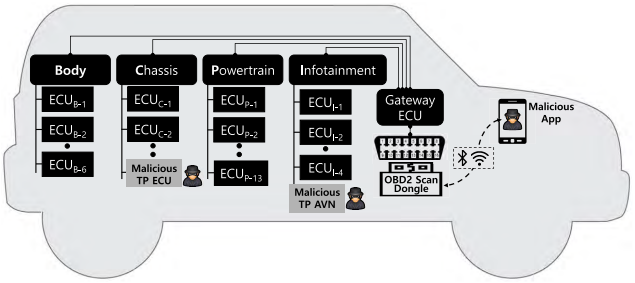


FIGURE 4. Conceptual Adversary Model for in-vehicle CAN.

B. ADVERSARY ABILITY

An adversary can easily acquire official information for hacking. They can manufacture a cyber weapon for target vehicle hacking using official information. The kinds of official information related to the vehicle's E/E system are as follows.

1) Network line structure: Identifiable from official manuals, such as diagnosis instructions.
2) Vehicle Diagnostic Tool (used to acquire ECU control data frame): Purchasable by general people.
3) SDK for developing vehicular application: Downloadable from web sites (e.g. ELM327, PLX KiWi) [25], [26].

C. ADVERSARY MODEL WITH VEHICULAR CYBER KILL CHAIN

K. Koscher et al. gave a presentation on hacking experiments using a real vehicle for the first time [6]. Based on their research findings, research on vehicle hacking has been conducted in various ways. Table 1 shows the representative research on vehicle hacking from 2010 to now. Most related research specified in Table 1 pointed out the vulnerability of CAN as the most serious problem. Injection of a malicious data frame into the in-vehicle CAN has three modes. Fig. 4 shows the conceptual adversary model that can be caused by the vulnerability of CAN and modern vehicles.

1) Modification of firmware of the AVN system (wired attack, wireless attack).
2) Manufacture of the malicious third party ECU and installation into a target vehicle (wired attack).
3) Wired/wireless access through an OBD2 terminal (wired attack, wireless attack).

Table 2 shows the vehicular cyber kill chain based on the above analysis. The vehicular cyber kill chain consists of seven phases, which can be classified into preparation, intrusion, and attack. In the adversary model, we do not consider DoS or man-in-the-middle attacks on in-vehicle CAN (An adversary cannot change the communication line of in-vehicle CAN).

D. ATTACK SURFACE AND SECURITY REQUIREMENT

Every research specified in Table 1 hacked a vehicle using a replay attack and an impersonation attack. To implement

TABLE 1. Characteristics of vehicle hacking research.

Research	Attack Interface	Research Contents	Vulnerability
[6]	Wired	• Experimented with hacking while first using a real vehicle. • First presented how to analyze the vulnerabilities of in-vehicle CAN	• In-vehicle CAN
[27]	Wireless	• First successful attempt at hacking into a commercial connected car service	• In-vehicle CAN • Connected car service
[7]	Wired	• Gave detailed information on an analytic process of in-vehicle CAN vulnerabilities • Conducted based on research [6]	• In-vehicle CAN
[8]	Wireless	• The most accurate and realistic vehicle hacking • Performed a wireless attack on a commercial connected car service	• In-vehicle CAN • Connected car service
[5]	Wireless	• First successful wireless attack using the smart phone of a driver. • Conducted based on research [6]	• In-vehicle CAN • Third Party Application

TABLE 2. Vehicular Cyber Kill Chain of our adversary model.

Preparation	Reconnaissance	CAN ID, CAN Data
	Weaponization	Malicious Smartphone Application, Third Party Device, and Malicious Firmware
Intrusion	Delivery	App Market, Third Party Market, and USB (SD Card)
	Exploitation	Exploiting a vulnerability to execute code on a target vehicle (device)
	Installation	Installing a weapon on the target vehicle (device)
Attack	Command and Control	ECU forced actuation attack through the weapon
	Action on Objectives	

the replay attack and the impersonation attack, the adversary first analyzes a sender (target) ID through a reconnaissance act. When performing the replay attack, the adversary uses ID field (Network Address) and data field information from a collected data frame. When implementing the impersonation attack, the adversary uses ID field information from a collected data frame. Everyone can participate in communication using an ID (Network Address) of a legitimate ECU because the CAN does not offer access control and entity authentication. The adversary executes the replay attack and the impersonation attack using such vulnerability. Therefore, in a CAN, the sender ID becomes the attack surface. Three types of vulnerabilities should be eliminated to paralyze the reconnaissance act at the vehicular cyber kill chain.

- No Confidentiality: Data are transmitted in plaintext.
- No Authentication: Authentication for data and entity is not performed.
- Weak access control: Every entity that accesses a communication line can participate in communication.

Confidentiality may be easily ensured by encryption. The most serious problems include authentication and access control. CAN is not able to ensure physically perfect access control. Furthermore, its restrictive data payload prevents it from using a practical data authentication technique. In our previous research, we analyzed the limitations of existing studies on CAN data frame authentication [12]. We also mentioned that it is impossible to apply a CAN data frame authentication scheme that satisfies both security and availability to in-vehicle CAN. The in-vehicle CAN is vulnerable to impersonation attacks and replay attacks because of the problems of access control and authentication mentioned above. Security requirements for the secure in-vehicle CAN environment may be divided into three items.

1) CONFIDENTIALITY

The CAN is a broadcast system and thus an adversary can easily conduct data frame sniffing. It is possible to analyze and acquire an ECU control command when using a vehicular diagnostic tool. The CAN provides confidentiality for a secure in-vehicle CAN.

2) PREVENT IMPERSONATION ATTACK

The CAN does not provide entity authentication. An adversary can conduct an impersonation attack using a legitimate sender ID. A proposed security solution has to provide entity authentication to construct an in-vehicle CAN that is secure from an impersonation attack. Logical access control is needed for a legitimate entity to transmit/receive meaningful data frames.

3) PREVENT REPLAY ATTACK

CAN uses a CRC code to detect a transmission error. CRC code is not able to prevent a replay attack. A proposed security solution must provide data frame authentication to construct an in-vehicle CAN that can defend against a replay attack.

V. PROPOSED MECHANISM (CIST)

In this section, we describe a new security technology concept that is able to construct an in-vehicle CAN that can defend

TABLE 3. Notation used for proposed mechanism.

Notation	Description
GECU	Gateway ECU
ECU_{i-j}	ECU using identity "i" and belonging to the subnetwork "j"
PID	Priority ID for arbitration
DID	Dynamic ID for ID shuffling
CTR_{i-j}	ECU_{i-j} data frame counter
$DID_{i-j-c-k}$	ECU_{i-j} Dynamic ID. When ECU_{i-j} transmits the c_{th} data frame in the k_{th} session
SK_{i-j}	Long-term symmetric key between GECU and ECU_{i-j}. (authentication key used for session key derivation)
KGK_j	Long-term symmetric key between GECU and all ECU belonging to the subnetwork "j". (key generation key used for session key derivation)
GSK_k	Group session key for One-Time key (OTK_{c-k}) generation. In the k_{th} session
GEK_k	Group Session key for Data Encryption. In the k_{th} session
OTK_{c-k}	One-Time key for dynamic ID generation. When ECU_{i-j} transmits the c_{th} data frame in the k_{th} session
$Seed_k$	Seed value of k_{th} session
C	Ciphertext
M	Plaintext
α	Number of ECUs in the same subnetwork
$HKDF_x()$	Keyed one-way hash function used for key derivation $HKDF_x : \{0,1\}^* \times key \rightarrow \{0,1\}^{256}$
$H_x()$	Keyed one-way hash function using x $H_x : \{0,1\}^* \times key \rightarrow \{0,1\}^{256}$

against an impersonation attack and a replay attack using a CAN ID shuffling technique (CIST). The purpose of CIST is to maximize the cost of a reconnaissance act performed by an adversary. The adversary that initiates an unsuccessful reconnaissance act will not be able to perform an impersonation attack and a replay attack. In our proposed security solution, we assume the following.

- Long-term symmetric key (SK_{i-j}, KGK_j) is stored in every ECU_{i-j} including GECU. Long-term symmetric key distribution is performed on a secure channel.
- Sender and receiver manage a synchronized data frame transmission counter (In CAN, the transmission state of the CAN data frame is checked reciprocally by senders and receivers using the ACK bit field. Therefore, a data frame counter can be managed and synchronized between the sender and the receiver. [5]).
- GECU is a trusted party.
- Device certificate is stored in GECU and an external device.

CIST is divided into seven phases. The main contributions are in phases *A*, *B*, *D* and *E*. Phases *C*, *F*, and *G* use a well-known security technique. This paper does not provide detailed descriptions of technology related to the management of a long-term symmetric key and a device certificate. The notation used in this paper is listed in Table 3.

A. CIST SYSTEM ARCHITECTURE

The purpose of CIST is to block a reconnaissance act by the adversary and prevent a replay attack and an impersonation attack at the same time. In CIST, the attack surface (sender ID) is to be shuffled using one-time ID whenever transmitting a data frame. CIST does not need an additional entity to decide the shuffling frequency of an ID because it uses one-time ID. In CIST, ECUs generate one-time IDs, so the ECUs themselves play the role of MTD controller. Fig. 5 shows the CIST system architecture. Gateway ECU (GECU) has a function to distribute group session keys (GSK) to be used for generating one-time ID. In addition, it connects a diagnostic device to a vehicle with certificate-based mutual authentication. GECU and every ECU that executed a group session key distribution process come to acquire the GSK. Every ECU possessing the GSK generates and uses a new one-time ID whenever transmitting a data frame. Every ECU plays the role of an MTD controller. Legitimate ECUs possessing the GSK do not have to conduct an additional communication to share the one-time key (OTK) because they can generate the OTK for themselves.

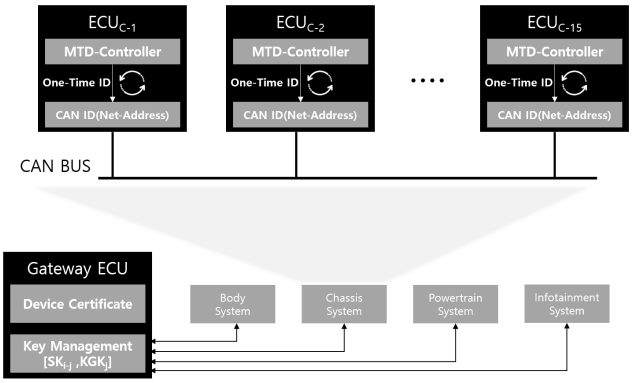


FIGURE 5. CIST system architecture.

B. CAN ID ALLOCATION FOR ID COLLISION RESISTANCE AND DATA FRAME PRIORITY

CIST is designed with two considerations:

- In the CAN, the data frame transmission priority is determined by the ID of the sender. Therefore, even if the sender ID is dynamically changed, the priority of data frame transmission should not be changed. CIST uses a PID to ensure the priority of data frame transmission.
- ECUs that belong to the same subnetwork are not able to use an identical CAN ID at the same time. CIST uses a PID to ensure ID collision resistance.

In the CAN 2.0B, the CAN ID consists of a base ID and an extended ID. A pre-designated CAN ID is used at a general vehicle and may not be changed. However, the dynamic allocation of CAN ID is permissible in the CAN protocol. CIST defines CAN ID as a priority ID (PID) and a dynamic ID (DID) in a new manner. PID is never changed once it is designated, but DID is continuously changed. The size of the PID and the DID is determined as follows.

- α is the number of ECUs in the same subnetwork.
- Positive integer α may be expressed as n bit.

$$(2^{n-1} \leq \alpha \leq 2^n\text{-}1)$$

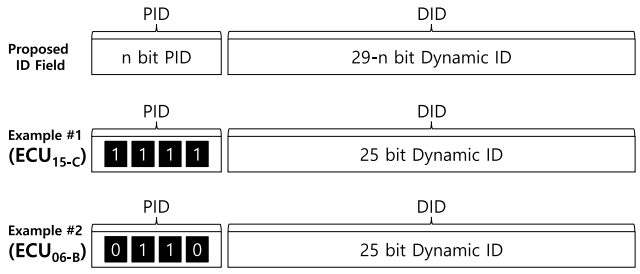


FIGURE 6. CAN ID allocation for data frame priority and collision resistance.

- PID bit = n bits, DID bit = 29-n bits.

The following are examples of the proposed CAN ID allocation scheme. In Fig. 1, the chassis system consists of 15 ECUs (α = 15). Here, the size of PID is 4 bits and the size of DID is 25 bits.

- Positive integer $\alpha(15)$ may be expressed as *4* bit. $(2^{4-1} \leq 15 \leq 2^4\text{-}1)$
- PID bit = *4* bits, Dynamic ID bit = 25 bits.

Fig. 6 shows an example of the proposed CAN ID allocation scheme. Because PID uses a fixed value, even though the same DID is assigned to two or more ECUs at the same time, the entire ID field will have different values. Furthermore, because PID is positioned far ahead at the front of the ID field, it does not affect the data frame transmission priority of ECUs.

C. ENTITY AUTHENTICATION AND GROUP SESSION KEY DISTRIBUTION

We use Authenticated Key Exchange Protocol 2 (AKEP2) for entity authentication and group session key distribution [28]. ECUs that belong to the same subnetwork perform AKEP2 in a fixed order. As shown from Fig. 7-(A), the process of AKEP2 performance consists of a 3-way handshake. The output size of $\mathsf{HKDF}_x()$ is 256 bits. The leftmost 128 bits are used as GSK and the rightmost 128 bits as GEK.

- $\mathsf{GSK_k}||\mathsf{GEK_k} = \mathsf{HKDF}_{\mathsf{KGK_j}}(\mathsf{SEED_k})$

GECU and every ECU which executed a group session key distribution process come to acquire the GSK.

D. DYNAMIC ID GENERATION AND DATA ENCRYPTION FOR SECURE COMMUNICATION

The sender ECU generates a DID and a ciphertext before transmitting a data frame. CIST uses a truncated HMAC to generate the DID. We use a one-time key (OTK) for DID generation to ensure security of truncated HMAC. Legitimate ECUs possessing a GSK can generate the OTK without an additional communication between the sender and the receiver. That is, only legitimate entities possessing the GSK can generate the OTK in safety without an additional communication because of three characteristics of the one-way hash function. Kannan Balasubramanian et al. described three characteristics of a one-way hash function as follows. [29]

- *Given H and α (any given input), it is easy to computer message digest H(α).*
- *Given H and H(α), it is computationally infeasible to find α.*
- *Given H and H(α), it is computationally infeasible to find α and α' such that H(α) = H(α').*

The ciphertext is generated using the AES-128 algorithm. Fig. 7-(B) shows the secure communication process when $\mathsf{ECU_{13\text{-}j}}$ transmits the $\mathsf{8_{th}}$ data frame in the $\mathsf{k_{th}}$ session.

•Data Frame Transmission

The sender generates a one-time key ($\mathsf{OTK_{8\text{-}k}}$) to be used for sending the 8th data frame using the group session key ($\mathsf{GSK_k}$). In the Fig. 7-(B), equation (4) shows how to generate the one-time key. The output size of $\mathsf{H}_x()$ is 256 bits. Here, the leftmost 128 bits are used as OTK. The OTK generation may be cut in half when using the leftmost 128 bits of $\mathsf{H}_x()$ output as $\mathsf{OTK_{c\text{-}k}}$ and the rightmost 128 bits as $\mathsf{OTK_{c+1\text{-}k}}$.

- $\mathsf{OTK_{8\text{-}k}} = \mathsf{H}_{\mathsf{GSK_k}}(\mathsf{OTK_{7\text{-}k}}||\mathsf{CTR_{13\text{-}j}})$

$\mathsf{DID_{13\text{-}j\text{-}8\text{-}k}}$ is then generated using $\mathsf{OTK_{8\text{-}k}}$. The output size of $\mathsf{H}_x()$ is 256 bits. Here, the leftmost n bits is used as DID.

- $\mathsf{DID_{13\text{-}j\text{-}8\text{-}k}} = \mathsf{H}_{\mathsf{OTK_{8\text{-}k}}}(\mathsf{DID_{13\text{-}j\text{-}7\text{-}k}}||\mathsf{CTR_{13\text{-}j}})$

The sender generates ciphertext C using group session key ($\mathsf{GEK_k}$) and data frame counter ($\mathsf{CTR_{13\text{-}j}}$).

- $\mathsf{C} = \mathsf{E}_{\mathsf{GEK_k}}(\mathsf{CTR_{13\text{-}j}}) \oplus \mathsf{M}$

The sender sends ciphertext(C) with $\mathsf{DID_{13\text{-}j\text{-}8\text{-}k}}$ and increments $\mathsf{CTR_{13\text{-}j}}$.

We use $\mathsf{SEED_k}$ as an input parameter when generating $\mathsf{OTK_{0\text{-}k}}$ and $\mathsf{DID_{i\text{-}j\text{-}0\text{-}k}}$ to be used first in the $\mathsf{k_{th}}$ session. Because $\mathsf{GEK_k}$ distributed from the group session key distribution process is a value which an adversary cannot infer, he/she cannot infer $\mathsf{OTK_{0\text{-}k}}$ and $\mathsf{DID_{i\text{-}j\text{-}0\text{-}k}}$ if $\mathsf{SEED_k}$ is used as the input parameter. It also comes from characteristics of the one-way hash function.

•Data Frame Reception

The receiver verifies the $\mathsf{DID_{13\text{-}j\text{-}8\text{-}k}}$ by using the DID generated in a previous data reception process. A simple comparison is performed only in the verification process. The data frame is dropped if verification fails. A decryption process is performed after DID verification is finished normally. After completing the DID verification and data decryption process, the receiver generates the $\mathsf{OTK_{9\text{-}k}}$ and the sender's DID ($\mathsf{DID_{13\text{-}j\text{-}9\text{-}k}}$) to be used next. Finally, the receiver increments the frame counter from sender data ($\mathsf{CTR_{13\text{-}j}}$).

E. ADVANCED TECHNIQUE FOR DYNAMIC ID GENERATION AND VERIFICATION

The sender and receiver can generate and save DID in advance before sending and receiving data frames. Because the parameters used to generate DID are predictable values, it is fully possible to pre-generate the DID of the sender. In Fig. 7-(B), the sender and receiver are able to process No.2, No.3, No.11, and No.12 when they are not communicating. Using this method, the sender and receiver can reduce the time required

(A) 3-Way Handshake for the Session Keys(GSK, GEK)

No.	Entity	Operation
1	All	Start
2	ECU_{i-j}	Selects random number $\mathbf{R_{i-j}}$
3	ECU_{i-j}	Transmits $\mathbf{R_{i-j}}$ to GECU
4	GECU	Receives $\mathbf{R_{i-j}}$ to GECU
5	GECU	Selects random number $\mathbf{Seed_k}$
6	GECU	Generates $\mathbf{MAC_1}$ as in (1)
7	GECU	Transmits $\mathbf{MAC_1||Seed_k}$ to ECU_{i-j}
8	ECU_{i-j}	Receives $\mathbf{MAC_1||Seed_k}$
9	ECU_{i-j}	Verifies $\mathbf{MAC_1}$ as in (1) if($\mathbf{MAC_1}$ == Incorrect) then goto 17 endif
10	ECU_{i-j}	Computes $\mathbf{GSK_k}$, $\mathbf{GEK_k}$ as in (2)
11	ECU_{i-j}	Generates $\mathbf{MAC_2}$ as in (3)
12	ECU_{i-j}	Transmits $\mathbf{MAC_2}$ to GECU
13	GECU	Receives $\mathbf{MAC_2}$
14	GECU	Verifies $\mathbf{MAC_2}$ as in (3) if($\mathbf{MAC_2}$ == Incorrect) then goto 17 endif
15	GECU	Computes $\mathbf{GSK_k}$, $\mathbf{GEK_k}$ as in (2)
16		End
17		Fail

Handshake ECU_{i-j} ——— R_{i-j} ———→ GECU

Handshake ECU_{i-j} ←——— $MAC_1||Seed_k$ ——— GECU

Handshake ECU_{i-j} ——— MAC_2 ———→ GECU

In the k_{th} session

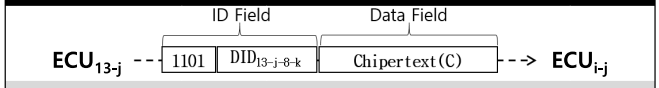
(1) $MAC_1 = H_{SKi-j}(ECU_{i-j} ||GECU|| R_{i-j}|| Seed_k)$

(2) $HKDF_{KGKj}(Seed_k) = GSK_k || GEK_k$

(3) $MAC_2 = H_{SKi-j}(ECU_{i-j} || Seed_k)$

(B) Data Transmission and Reception (8_{th} data frame in the k_{th} Session)

No.	Entity	Operation
1	All	Start
2	ECU_{13-j}	Generates OTK_{8-k} as in (4)
3	ECU_{13-j}	Generates $DID_{13-j-8-k}$ as in (5)
4	ECU_{13-j}	Generates Ciphertext (C) as in (6)
5	ECU_{13-j}	Transmits [PID+$DID_{13-j-8-k}$ || C] to Receiver ECU_{i-j}
6	ECU_{13-j}	Increments CTR_{13-j} (CTR_{13-j} + 1)
7	Receiver	Receives [PID+$DID_{13-j-8-k}$ || C]
8	Receiver	Verifies $DID_{13-j-8-k}$ use the pre-generated $DID_{13-j-8-k}$ if(MAC == Incorrect) then **goto 14** endif
9	Receiver	Decryption and obtains plaintext (M) as in (7)
10	Receiver	Increments CTR_{13-j} (CTR_{13-j} + 1)
11	Receiver	Generates OTK_{9-k} as in (4)
12	Receiver	Generates $DID_{13-j-9-k}$ as in (5)
13		End
14		Fail



8_{th} data frame in the k_{th} session

(4) $H_{GSK_k}(OTK_{7-k} || CTR_{13-j}) = OTK_{8-k}$

(5) $H_{OTK_{8-k}}(DID_{13-j-7-k} || CTR_{13-j}) = DID_{13-j-8-k}$

(6) $E_{GEK_k}(CTR_{13-j}) \oplus M = C$

(7) $E_{GEK_k}(CTR_{13-j}) \oplus C = M$

FIGURE 7. Session Key Distribution and Secure Communication.

to generate the DID of the sender when sending and receiving data frames.

F. SHARING A SESSION KEY WITH AN EXTERNAL DEVICE

We suggested the session key sharing scheme based on a device certificate from our previous research [5]. External devices authenticated by an auto manufacturer may ensure a session key after executing certificate-based mutual authentication. The certificate-based mutual authentication and session key sharing are very general skills, so we omit a detailed description.

G. ACCESS CONTROL POLICY

Every ECU that accesses an in-vehicle CAN bus can transmit/receive a data frame. An abnormal entity may also participate in communication when attempting to access the CAN bus. In fact, it is impossible to execute physically perfect access control in an in-vehicle CAN. Although it is possible to find abnormal connections using an intrusion detection system (IDS), network IDS is a complex classification problem [30], so it is difficult to apply it to the vehicular ECUs.

Accordingly, logical access control shall be conducted that prevents abnormal entities from executing meaningful communication, although they access the communication line. In order to execute logical access control, access control policies shall be operated in in-vehicle CAN as follows.

1) Every ECU that accesses an in-vehicle CAN should perform entity authentication and session key derivation. Legitimate ECUs can acquire a session key only.
2) An external device authenticated by an auto manufacturer should access in-vehicle CAN only.
3) A vehicle and an external device should execute mutual authentication and session key derivation.
4) Legitimate entities may execute data frame encryption and CAN ID shuffling using a session key.
5) Abnormal entities may not analyze the meaning of an encrypted data frame.
6) Abnormal entities may not execute an impersonation attack because they cannot use CIST.

VI. SECURITY AND PERFORMANCE ANALYSIS

A. SECURITY ANALYSIS

In this subsection, we analyze the security of CIST according to the security requirement defined in section III. The main concept of MTD is to increase the complexity of attacks so that an adversary pays much more cost to mount an attack. Thus, to measure the merit of the proposed method from the viewpoint of the MTD technique, we measure the increased complexity of attacks caused by the adoption of the proposed method. Because the goal of the MTD technique is to improve the cost of attacks *Att*, it is reasonable to define the *effect* of a countermeasure *CM* for a system *Sys* as

$$EF^{CM}_{Sys,Att} = \frac{\text{Complexity of } Att \text{ on } Sys \text{ with } CM}{\text{Complexity of } Att \text{ on } Sys \text{ without } CM}$$

The value of Complexity of Att on Sys without CM to 1, when the security method is not applied to the system (e.g. If the attack success probability is $\frac{1}{2^k}$, we need 2^k for the value of Complexity of *Att* on *Sys* without *CM*).

The security of an in-vehicle CAN can be broken if one of the security features are damaged, and thus we can defined the *effect* of CIST as

$$EF^{\mathsf{CIST}}_{IVC,ATT} = \min\{EF^{\mathsf{CIST}}_{IVC,Con}, EF^{\mathsf{CIST}}_{IVC,Imp}, EF^{\mathsf{CIST}}_{IVC,Rep}\},$$

where *IVC* represents the in-vehicle CAN, *Con* is the attack to break confidentiality, *Imp* is an impersonation attack, *Rep* is a replay attack, and $ATT = \{Con, Imp, Rep\}$.

Now, we measure *effect*s of CIST against three security threats. Recall that security functions are not available in the in-vehicle CAN. That means that in a in-vehicle CAN without CIST, any adversary can see any data to break confidentiality, disguise ECUs by generating their packets, and reuse any data frames. Therefore, we set the value of Complexity of *Att* on *Sys* without *CM* to 1 and analyze the security of CIST.

Theorem 1: The *effect* of CIST in terms of confidentiality is $EF^{\mathsf{CIST}}_{IVC,Con} = 2^{128}$.

Proof: To measure the *effect* of CIST in terms of confidentiality, we examine the increased difficulty of obtaining information from communicating messages compared with naive CAN communication. Differently from the naive CAN, the proposed scheme uses AES-128 to encrypt the data part. Only legitimate entities may acquire $\mathsf{GEK_k}$ used for the generation of a ciphertext. Abnormal entities cannot decipher plaintext even if they are acquiring a ciphertext because they have no $\mathsf{GEK_k}$ [35]. Hence, the capability of an adversary is restricted by the difficulty of the underlying encryption scheme. Because the encryption scheme can guarantee 2^{128}-bit security, we have $EF^{\mathsf{CIST}}_{IVC,Con} = 2^{128}$. ■

In the following theorems, we assume that it is difficult to extract private information such as OTK from collected data frames. Because we use secret information as a part of the input for a cryptographic hash function, which guarantees pre-image resistance and only the output of the function are revealed to an adversary, it is reasonable to assume this statement. Hence, from now, we exclude the case where an adversary extracts any information from data frames and uses it to improve the success probability.

Theorem 2: The effect of CIST against an impersonation attack is $EF^{\mathsf{CIST}}_{IVC,Imp} = 2^{29-n}$.

Proof: In the proposed security scheme, only legitimate entities can generate DID. To mount an impersonation attack, an adversary needs valid DID because the receiver ECU can identify an abnormal data frame using DID. From the viewpoint of the adversary, the only way to obtain valid DID is to guess the value because it cannot be computed without secret information OTK. Hence, the adversary can mount the attack only if the guess DID is correct. A guessed DID is correct with probability 2^{29-n} because $|\mathsf{DID}| = 29-n$. Hence, we can see that $EF^{\mathsf{CIST}}_{IVC,Imp} = 2^{29-n}$. ■

Theorem 3: The effect of CIST against replay attack is $EF^{\mathsf{CIST}}_{IVC,Rep} = 2^{29-n}$.

Proof: To mount a replay attack, an adversary may collect a number of data frames and use one of them as a valid data frame. To be a successful attack, the selected data frame should include valid DID for the chosen target ECU and the current counter. Even if the adversary collects a sufficient quantity of data frames, it is not easy to guess the correct DID for a specific data frame. This is because the DID is computed from secret information and a message counter using a cryptographic hash function, which guarantees randomness in the outputs. Thus, the only way to mount a successful replay attack is to choose correct DID from collected DIDs. Because it seems possible to collect 2^{29} data frame, we can assume that an adversary collect all possible DIDs. Hence, the difficulty of executing a replay attack is determined by the size of DID because it influences the probability of correct guess. Because DID can be correctly guessed without input data with probability 2^{29-n}, we have $EF^{\mathsf{CIST}}_{IVC,Rep} = 2^{29-n}$. ■

Note that the first theorem explains the *effect* of the proposed scheme in the offline scenario where an adversary can use as much computing power as needed. However, Theorem 2 and 3 explain the *effect* of CIST in the on-line scenario where an adversary should generate correct values in the on-line phase. When high-performance devices are available for attacks, we can improve the performance of an attack in off-line scenario, but not in an on-line scenario. In some instances, it is better to give the *effect* of CIST in both viewpoints. CIST supports 2^{128} and 2^{29-n} *effect*s against off-line and on-line attacks, respectively. In practice, an off-line problem with 2^{128} *effect* is more difficult than an on-line problem with 2^{29-n} *effect*, and thus we can say that CIST guarantees 2^{29-n} *effect* in the on-line problem.

B. SECURITY COMPARISON

In this subsection, we compare CIST to the security solution proposed in [5], [11], [33], and [34]. Table 4 shows the results of a comparative evaluation. The four previous studies use the ID, Data, and CRC fields to implement their proposed countermeasure. Fig. 8 shows the field information used in each technique to apply a security solution.

TABLE 4. The results of comparative evaluation.

	CIST	Truncated-MAC	Mini-MAC	ID-Hopping	ID-Obfuscation
Confidentiality	O	O	X	X	X
No Communication overhead	O	O	X	O	O
No Standard Change	O	X	O	O	O
Prevent Impersonation attack	O	O	O	X	X
Prevent Replay attack	O	O	O	X	X
Prevent DoS attack	X	X	X	X	X

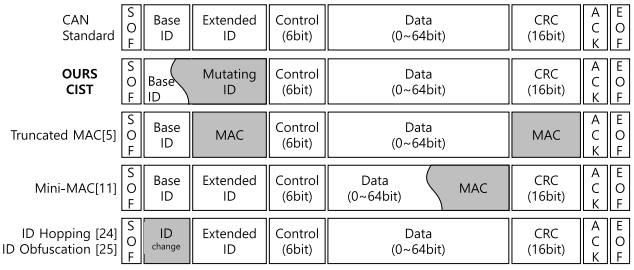


FIGURE 8. Comparison of data frame usage for security solutions.

CIST, ID-hopping, and ID-obfuscation use a similar concept that CAN ID is to be shuffled to protect in-vehicle CAN. However, each technique differs in the way it shuffles CAN IDs. In addition, each technology has different security effects. How and why CIST, ID-Hopping and ID-obfuscation shuffle a CAN ID is as follows.

• **ID Hopping:** ID-Hopping aims to resist a DoS attack by shuffling the attack surface when a DoS attack occurs. To shuffle an ID, a new ID is generated by adding a value of offset to that of an ID used currently (NEW ID = Previous ID + offset). A fixed new ID is used for some time. In other words, the attack surface does not move for some time. An ID generation method proposed by ID-Hopping cannot detect resued IDs, so a replay attack and an impersonation attack cannot be prevented.

• **ID obfuscation:** ID-obfuscation aims to minimize the spread of an attack by allowing vehicles to use different ID systems. In general, internal ECUs use the same ID system in the same vehicle model. ID-obfuscation assigns an ID for ECUs loaded to every vehicle to use different ID systems. However, a pre-assigned ECU ID is not shuffled for a long time. That is, the attack surface does not move for a long time. The ID generation method proposed by ID-obfuscation cannot detect when an ID has been reused. Hence, both a replay attack and an impersonation attack cannot be prevented.

• **CIST:** The purpose of CIST is to prevent a reconnaissance act by an adversary and construct an in-vehicle CAN environment secured from a replay attack and an impersonation attack. In CIST, the attack surface is to be shuffled using a one-time ID whenever transmitting a data frame. An ID will be shuffled using the fastest frequency that can be used in the in-vehicle CAN. When a one-time ID is generated, it is possible to detect a reused ID because one-way hash function with a group session key and counter value is used. In other words, reconnaissance acts, replay attacks, and impersonation attacks can be prevented. However, an ID generation method purposed by ID-Hopping and ID-obfuscation schemes cannot detect reused IDs. Accordingly, neither a replay attack nor an impersonation attack can be prevented.

It is said that both ID-Hopping and ID-obfuscation schemes can resist a DoS attack, but it is not possible to resist a DoS attack in a CAN. In addition, CIST cannot resist a DoS attack. The reason why the three schemes including CIST cannot completely resist DoS attacks is explained as follows. First, in a CAN bus system, the dominant bit always overwrites the recessive bit. An adversary can perform a DoS attack using a method that involves continuously sending dominant bits. This prevents other nodes from sending any type of message on the bus. B. Groza et al. proposed four kinds of DoS attacks using these characteristics. These attacks can be implemented with CAN-based protocols such as CANopen, TTCAN, SAE J1939. In order to prevent these types of attacks, it is necessary to add physical devices, such as a firewall to the in-vehicle CAN environment [36]. Second, a DoS attack on an in-vehicle CAN is more influenced by a data frame transmission cycle than by the CAN ID priority. To construct an in-vehicle CAN that can protect against a DoS attack, a firewall is to be used [37]. However, applying the firewall method by [37] to an in-vehicle CAN environment requires the installation of a significant quantity of firewalls. Third, a malicious ECU can infer the altered CAN ID using the data field format. It is possible to analyze the altered CAN ID through a very simple comparative operation.

Truncated-MAC and Mini-MAC offer data frame authentication to prevent replay and impersonation attacks. However, it is not recommended to use the data field to transmit a message authentication code because of the communication overhead. Truncated-MAC is also impossible to apply to a real vehicle environment. In order to use the CRC field for other purposes, it is necessary to modify the CAN standard. All schemes except CIST and Truncated-MAC do not provide data confidentiality. If confidentiality is not guaranteed, it is possible to analyze the meaning of certain data frames. As explained in section V, if an adversary can analyze the meaning of the data frame, the adversary can disable the

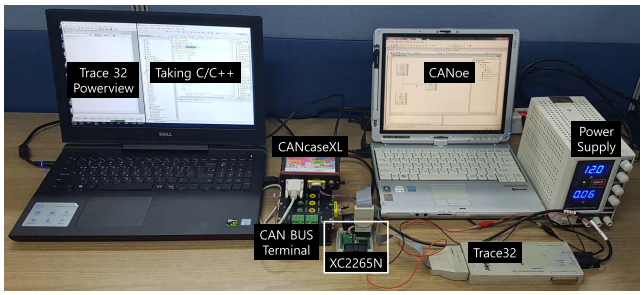


FIGURE 9. Labcar environment for performance evaluation.

TABLE 5. Tools used for the evaluation.

Tool	Model Name	Note
MCU 1	Infineon XC2265N	clock speed: 80 MHz
Compiler	Tasking C/C++	For XC2265N
Debugger	Trace32 Powerview	For XC2265N
MCU reg setting	Dave	For XC2265N
Network simulator	CANoe	For CAN Communication
CAN BUS Switch	CAN BUS terminal	For CAN Communication
Interface Unit	CANcaseXL	CANoe to CAN BUS

ID-Hopping and ID-Obfuscation. CIST provides confidentiality. In addition, because CIST uses counters between the sender and receiver for DID generation, replay and impersonation attacks are impossible. The CAN standard allows for the value of the ID field to be changed during communication. CIST is a practical security technique that can disable a reconnaissance act by using the CAN ID shuffling function allowed by the CAN standard.

C. PERFORMANCE EVALUATION

In this subsection, we analyze CIST performance. In CIST, a one-time ID is used to dynamically shuffle the attack surface. We conducted a hardware-based evaluation and measured the time needed to generate a one-time ID. In the hardware-based evaluation, the execution time of the cryptography algorithms and the CAN ID register setting were measured. In addition, with the network simulator based evaluation, we measured the timing of the communication delay when conducting communication using one-time ID. In the network simulator based evaluation, communication delay and group session key distribution delay were measured. We proved that CIST can be applied to an in-vehicle CAN environment through the experiment. Table 5 shows the hardware and software used to construct the labcar environment. The labcar environments are as shown in Fig. 9.

1) HARDWARE-BASED EVALUATION

We used a XC2265N-based evaluation board to measure the execution time spent in the cryptography algorithms

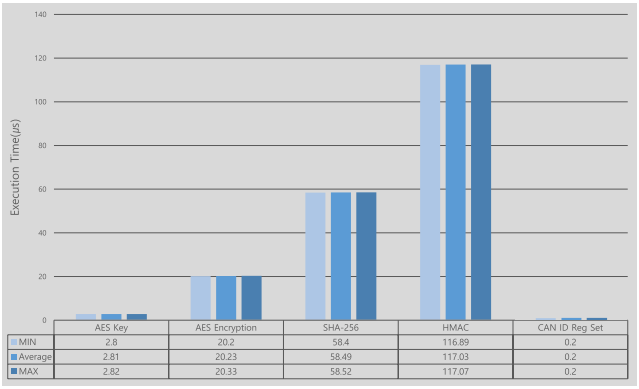


FIGURE 11. Execution time of the cryptography algorithm and the CAN ID register setting.

(HMAC, SHA256, AES-128) and the CAN ID register setting. We applied an AES-128 algorithm, which does not use a pre-computed lookup table. For the HMAC algorithm, we used a source code provided by Openssl. Fig. 10 shows the execution time of cryptography algorithms. All ECUs perform DID generation and data encryption (decryption) during the communication process. A sender ECU generates the DID and the ciphertext for secure communication. A receiver ECU needs to verify the DID of the sender ECU and decrypt the ciphertext. Here, the sender and the receiver perform the HMAC operation twice and the AES operation once.

Through experiments, we confirmed that the sender was able to execute DID generation, CAN ID register setting, and data encryption within 270μs. As shown in subsection V-D, for efficient communication, DID generation may be carried out in advance before sending/receiving a data frame. In other words, ECUs only have to perform encryption/decryption during the sending and receiving of data frames. A simple comparison operation is performed only during the verification process. As shown in subsection V-C, OTK generation cost may be also decreased by half. Furthermore, CIST may be applied more efficently if we use an HSM-embedded ECU, such as TC275 [12].

2) NETWORK SIMULATOR-BASED EVALUATION

CANoe was used for the evaluation based on the network simulator. This network simulator is used for developing or testing embedded systems for vehicles [38]. CIST was also implemented on a CANoe virtual ECU node. CIST was manufactured with DLL and applied to CANoe. We also implemented the results of the hardware-based evaluations of execution time delay for the network simulator based evaluation.

a: COMMUNICATION DELAY

An evaluation scenario for measuring the communication delay is described as follows (The CAN ID of receiver-ECU was fixed. XC2265N was used for a sender-ECU and CANoe was used for a receiver-ECU.).

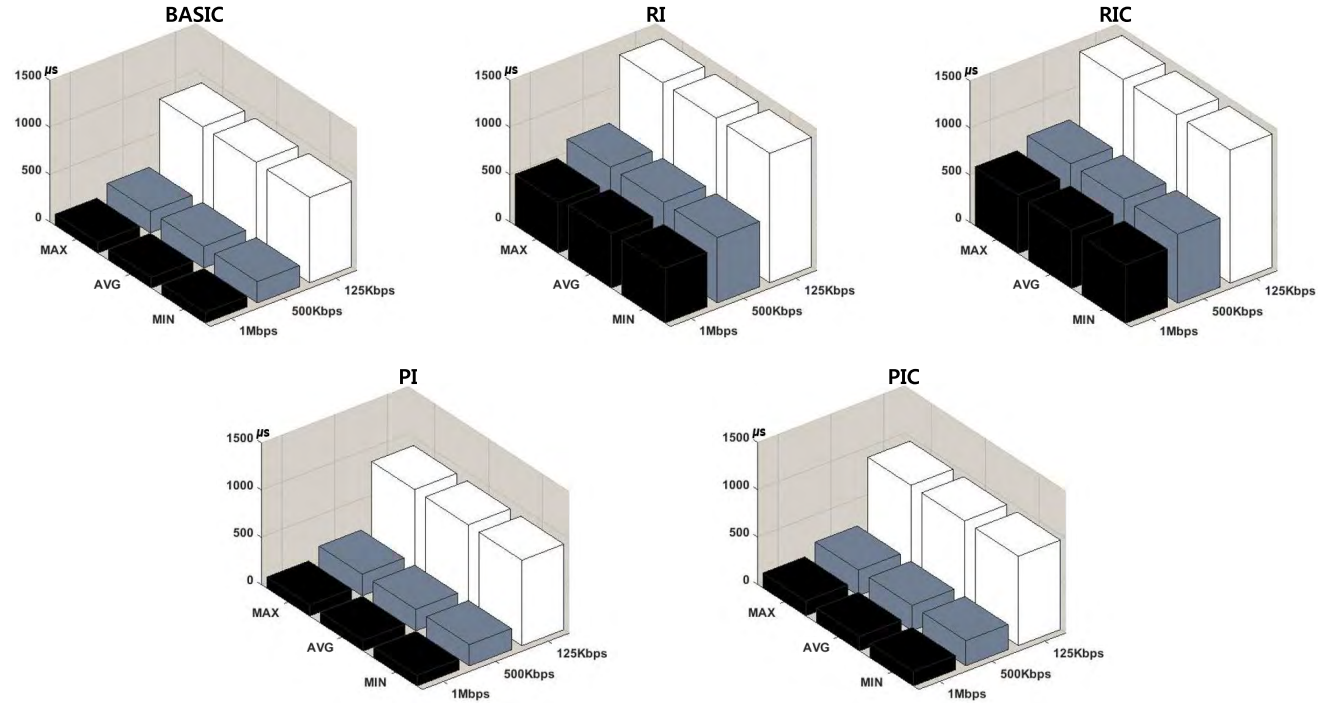


FIGURE 10. Communication Delay in term of the baud rate.

1) sender-ECU generates a request message. (DID generation and data encryption)
2) sender-ECU transmits a request message.
3) receiver-ECU receives a request message. (DID verification[simple comparison], data decryption and generate the sender's DID to be used next.)
4) receiver-ECU transmits a response message.
5) sender-ECU receives a response message.

The step from 1 to 3 performs the process from 2 to 11 presented in Fig. 7-(B). In step 4, the receiver-ECU sends a response message without using the DID and the ciphertext. We measured the execution time from the time a sender-ECU generates a request message to the time it receives a response message. We defined four options and conducted the evaluation.

1) **RI** (**R**eal Time generated **I**D): The sender-ECU and receiver-ECU perform DID generation whenever sending/receiving a data frame. The sender-ECU generate DID immediately before sending a data frame. The receiver-ECU generate sender's DID to use when receiving the next data frame after receiving a data frame.
2) **RIC** (**R**eal Time generated **I**D with **C**iphertext): A DID generating condition identical to **RI** is used. Data encryption/decryption operations are conducted in the process of sending/receiving a data frame.
3) **PI** (**P**re-generated **I**D): The sender-ECU and receiver-ECU generate and store sender's DID in advance before sending/receiving a data frame. A simple comparison operation is performed only in the reception process.
4) **PIC** (**P**re-generated **I**D with **C**iphertext): A DID generating condition identical to **PI** is used. Data encryption/decryption operations are conducted in the process of sending/receiving a data frame.

Fig. 11 shows the communication delay in term of the baud rate. BASIC shows the result of the communication delay when the sender-ECU and receiver-ECU use only a two-way exchange for a message without executing CIST. There was little difference between the maximum delay and minimum delay. There are three causes of delay when applying CIST to an in-vehicle CAN environment.

1) DID and ciphertext generation (Algorithm execution delay).
2) CAN ID change (Register setting delay).
3) Baud rate of the CAN bus (Data transmission delay).

DID generation and the baud rate causes the greatest delay. A sender-ECU performs the HMAC operation two times in order to generate DID. The receiver-ECU should also conduct HMAC two times in order to generate the sender's DID. In our experimental scenario, HMAC operation is to be conducted four times in the process of sending/receiving a data frame. In the evaluations using options RI and RIC, It takes approximately 1380μs and 1420μs to perform the request and response operation once. In the case of preprocessing DID generation and verification (options PI and PIC), it is possible for ECUs having performance under 80 MHz to send/receive a data frame within 952μs. The time required for newly setting the CAN ID register in XC2265N is 0.2μs. Compared to the HMAC operation time, it is a negligible time delay. Our experiment shows that the communication delay

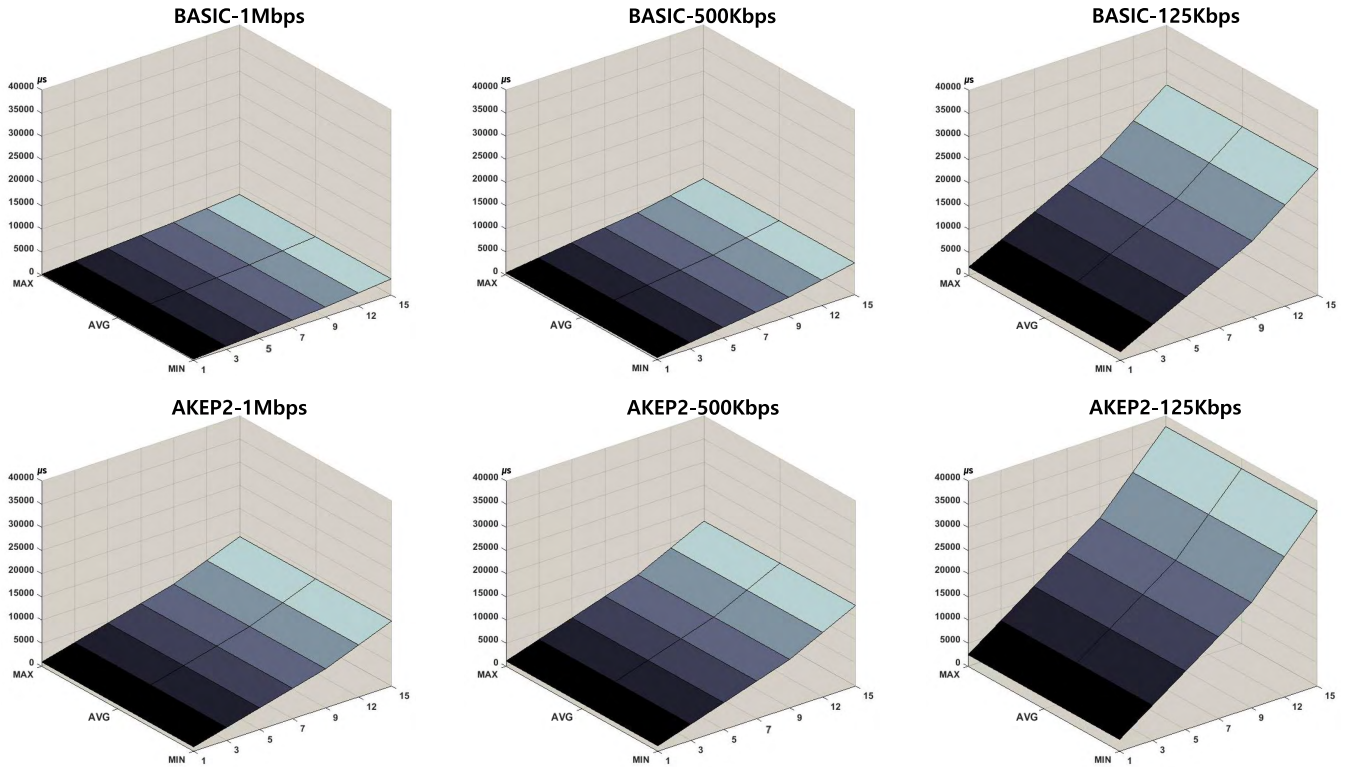


FIGURE 12. Group session key distribution delay in term of the number of ECUs and the baud rate.

is strongly influenced by DID generation and the baud rate when applying the proposed scheme to an in-vehicle CAN environment.

b: GROUP SESSION KEY DISTRIBUTION DELAY

Changing **the number of ECUs** and **the baud rate of the CAN bus**, we measured the session key distribution delay. In this experiment, the XC2265N was used for the GECU and the CANoe virtual node was used as the ECUs participating in communication. The AKEP2 is a long-term symmetric key-based entity authentication and key distribution protocol. Therefore, in the group session key distribution process, GECU and all ECUs do not use CIST. While performing the AKEP2 protocol, GECU and ECU conduct HMAC operations two times for MAC generation and verification, HKDF operations one time for a group session key derivation. Fig. 12 shows the results for a group session key distribution delay from our experiment.

BASIC shows the communication delay when all ECUs and GECU perform a three-way exchange for a message without AKEP2. The most of the time spent in the group session key derivation was due to HMAC operation and baud rate of the CAN bus. In particular, the baud rate has a great influence on the key distribution delay. The time spent in processing the cryptography algorithm may be reduced when using an ECU with a high CPU clock rate or an ECU loaded with HSM [12]. As shown in the results of Fig. 12, for 125 Kbps, the time for 15 ECUs to complete the session key derivation process is less than 38ms.

3) PRACTICAL IMPLICATIONS OF CIST

In sections I and II, we stated the reason why the existing data authentication schemes cannot be applied to an in-vehicle CAN environment. The reason is as follows. The CAN data frame is too small to use the message authentication code (MAC). The additional transmission of a data frame for MAC causes an authentication delay and increased bus load. Hence, using a data authentication scheme to resist (defend against) replay and impersonation attacks creates a trade-off between security and efficiency. A security scheme that does not create data frame transmission overhead or increase bus load shall be used to resist replay and impersonation attacks in an in-vehicle CAN environment. In addition, it is necessary to minimize the communication delay that occurs when sending and receiving a data frame.

We proved that CIST is a practical security solution through the performance evaluation and the security analysis. CIST is the scheme that dynamically shuffles the attack surface using the one-time ID. It shuffles an ID whenever transmitting a data frame, so it shuffles the attack surface at the fastest frequency. That is, it is possible to set the highest frequency to shuffle the attack surface. Moreover, as shown from the performance evaluation, CIST does not increase bus load. In particular, algorithms composing CIST can work fast even when paired with a low-performance ECU. As shown in Fig. 10, it is possible to use algorithms to generate a one-time ID even with a low-performance ECU. In cases where an HSM-loaded ECU such as TC275 is used, the time spent to generate a one-time ID may be reduced further [12]. There are not many

operations required to generate and verify a one-time ID, thus real-time data processing is possible. In addition, because pre-computation is possible for the one-time ID proposed by CIST, the communication delay that occurs when sending and receiving a data frame has no significant effect on the performance of the entire network. As described above, CIST is the practical security scheme used to guarantee real-time data processing and resist replay and impersonation attacks effectively without increasing bus load.

VII. CONCLUSION

MTD is a new concept of security technology in which Information & ICT infrastructures actively change their form to defend against various types of cyber-attacks. One MTD strategy known as NAS is a practical security solution that makes it difficult for reconnaissance acts to be successful. It is possible to significantly increase the cost of executing a reconnaissance act if the NAS technique is applied to an in-vehicle CAN. In this paper, we proposed a new defense solution that implements shuffling of the CAN ID. Our proposed CAN ID shuffling technique (CIST) is a practical security scheme that prevents a reconnaissance act, a replay attack, and an impersonation attack. In order to directly apply our technique to modern vehicles, CIST is designed without changing the intrinsic properties of the CAN standard. We performed a labcar-based evaluation to analyze the performance of CIST. The evaluation has proven that the CIST has sufficient availability.

REFERENCES

[1] R. N. Charette, "This car runs on code," *IEEE Spectr.*, vol. 46, no. 12, p. 7, Dec. 2009.
[2] M. Wolf, A. Weimerskirch, and T. Wollinger, "State of the art: Embedding security in vehicles," *EURASIP J. Embedded Syst.*, vol. 2007, p. 074706, Jun. 2007.
[3] *CAN Specification Version 2.0*, BOSCH, Stuttgart, Germany,1991.
[4] K. H. Johansson, M. Törngren, and L. Nielsen, "Vehicle applications of controller area network," in *Handbook of Networked and Embedded Control Systems*. Boston, MA, USA: Birkhäuser, 2005, pp. 741–765.
[5] 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.
[6] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, May 2010, pp. 447–462.
[7] C. Miller and C. Valasek, "Adventures in automotive networks and control units," in *Proc. DEF CON*, vol. 21, 2013, pp. 260–264.
[8] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *Proc. Black Hat*, 2015, pp. 1–91.
[9] B. Groza and S. Murvay, "Efficient protocols for secure broadcast in controller area networks," *IEEE Trans. Ind. Informat.*, vol. 9, no. 4, pp. 2034–2042, Nov. 2013.
[10] EVITA. *Homepage of the EVITA Project*. Accessed: 2008. [Online]. Available: http://evitaproject
[11] J. Schmandt, A. T. Sherman, and N. Banerjee, "Mini-MAC: Raising the bar for vehicular security with a lightweight message authentication protocol," *Veh. Commun.*, vol. 9, pp. 188–196, Jul. 2017.
[12] S. Woo, H. J. Jo, I. S. Kim, and D. H. Lee, "A practical security architecture for in-vehicle CAN-FD," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2248–2261, Aug. 2016.
[13] G.-L. Cai, B.-S. Wang, W. Hu, and T.-Z. Wang, "Moving target defense: State of the art and characteristics," *Frontiers Inf. Technol. Electron. Eng.*, vol. 17, no. 11, pp. 1122–1153, 2016.
[14] *Moving Target Defense*. Accessed: 2013. [Online]. Available: https://www.dhs.gov/science-and-technology/csd-mtd
[15] H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein, "Finding focus in the blur of moving-target techniques," *IEEE Security Privacy*, vol. 12, no. 2, pp. 16–26, Mar./Apr. 2014.
[16] J. Lowry, R. Valdez, and B. Wood, "Adversary modeling to develop forensic observables," in *Proc. Air Force Res. Lab's Digit. Forensic Res. Workshop*, 2004, pp. 11–13.
[17] V. Heydari, "Moving target defense for securing SCADA communications," *IEEE Access*, vol. 6, pp. 33329–33343, 2018.
[18] S. Yan, X. Huang, M. Ma, P. Zhang, and Y. Ma, "A novel efficient address mutation scheme for ipv6 networks," *IEEE Access*, vol. 5, pp. 7724–7736, 2017.
[19] J. Sun and K. Sun, "DESIR: Decoy-enhanced seamless IP randomization," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr. 2016, pp. 1–9.
[20] H. Gustavsson and J. Axelsson, "Evaluating flexibility in embedded automotive product lines using real options," in *Proc. 12th Int. Softw. Product Line Conf.*, Sep. 2008, pp. 235–242.
[21] T. Nolte, H. Hansson, and L. L. Bello, "Automotive communications-past, current and future," in *Proc. IEEE Conf. Emerg. Technol. Factory Automat.*, vol. 1, Sep. 2005, pp. 1–8.
[22] (2004). *BOSCH CAN*. [Online]. Available: http://www.can.bosch.com
[23] *J1939: Recommended practice for a serial control and communications vehicle network*, Standard J1939_201308, 2013, pp. 1–2.
[24] (2014). *The Lockheed Martin Cyber Kill Chain*. [Online]. Available: http://cyber.lockheedmartin.com/hubfs/Gaining_the_Advantage_Cyber_Kill_Chain.pdf
[25] *ELM Electronics*. Accessed: 2019. [Online]. Available: https://www.elmelectronics.com
[26] *PLX Device*. Accessed: 2015. [Online]. Available: https://www.plxdevices.com
[27] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. 19th Conf. USENIX Secur. Symp.*, Washington, DC, USA, 2011, pp. 77–92.
[28] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Proc. Conf.-CRYPTO*, 1993, pp. 232–249.
[29] K. Balasubramanian, "Hash functions and their applications," in *Algorithmic Strategies for Solving Complex Problems in Cryptography*. Pennsylvania, PA, USA: IGI Global, 2018, pp. 66–77.
[30] B.-C. Zhang, G.-Y. Hu, Z.-J. Zhou, Y.-M. Zhang, P.-L. Qiao, and L.-L. Chang, "Network intrusion detection based on directed acyclic graph and belief rule base," *Electron. Telecommun. Res. Inst. J.*, vol. 39, pp. 592–604, Aug. 2017.
[31] C. Miller and C. Valasek, "A survey of remote automotive attack surfaces," BlackHat, Las Vegas, NV, USA, Tech. Rep., 2014.
[32] 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, no. 4, pp. 1511–1530, 2017.
[33] H. Abdulmalik and B. Luo, "Using ID-hopping to defend against targeted DoS on CAN," in *Proc. 1st Int. Workshop Safe Control Connected Auton. Vehicles*, 2017, pp. 19–26.
[34] L. Martin, P. Mundhenk, and S. Steinhorst, "Security-aware obfuscated priority assignment for automotive CAN platforms," *ACM Trans. Des. Automat. Electron. Syst.*, vol. 21, no. 2, pp. 1–27, 2016.
[35] A. Hodjat and I. Verbauwhede, "Minimum area cost for a 30 to 70 Gbits/s AES processor," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Feb. 2004, pp. 83–88.
[36] P.-S. Murvay and B. Groza, "DoS attacks on controller area networks by fault injections from the software layer," in *Proc. 12th Int. Conf. Availability, Rel. Secur.*, 2017, p. 71.
[37] C. Patsakis, K. Dellios, and M. Bouroche, "Towards a distributed secure in-vehicle communication architecture for modern vehicles," *Comput. Secur.*, vol. 40, pp. 60–74, Feb. 2014.
[38] *Vector*. Accessed: 2010. [Online]. Available: http://www.vector-informatik.com

SAMUEL WOO received the Ph.D. degree in information security from Korea University, Seoul, South Korea, in 2016. He joined the Electronics and Telecommunications Research Institute, in 2016, where he is currently a Senior Researcher with the Network Security Research Laboratory. His research interests include cryptographic protocols in authentication, security and privacy in vehicular networks, and controller area network security.

DAESUNG MOON received the M.S. degree from the Department of Computer Engineering, Busan National University, Busan, South Korea, in 2001, and the Ph.D. degree in computer science from Korea University, Seoul, South Korea, in 2007. He joined the Electronics and Telecommunications Research Institute, in 2000, where he is currently a Principal Researcher. His research interests include network security, data mining, and biometrics.

TAEK-YOUNG YOUN received the B.S., M.S., and Ph.D. degrees from Korea University, in 2003, 2005, and 2009, respectively. Since 2016, he has been serving as an Associate Professor with the University of Science and Technology, Daejeon, South Korea. He is currently a Senior Researcher with the Electronics and Telecommunications Research Institute. His research interests include cryptography, information security, authentication, and data privacy.

YOUSIK LEE is currently pursuing the Ph.D. degree in information security with Korea University. He has been in the cyber security industry for over 17 years, especially eight years in automotive security. He specializes in consulting and development for application security, PKI, cryptography, and automotive security. He is currently a Security Consultant of ESCRYPT GmbH. His research interests include in-vehicle network security, V2X, and evaluation methodologies.

YONGEUN KIM received the B.S., M.S., and Ph.D. degrees in electrical engineering from Chonbuk National University, Jeonju, South Korea, in 2005, 2007, and 2010, respectively. He is currently a Researcher with the Vehicle-IT Convergence R&D Center, Korea Automotive Technology Institute, Cheonan, South Korea. His current research interests include electric machinery and its drives, micro controller, and the applications of motor drives such as electric vehicles.