

Received February 28, 2020, accepted March 31, 2020, date of publication May 4, 2020, date of current version May 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992112

# Practical Sender Authentication Scheme for In-Vehicle CAN With Efficient Key Management

TAEK-YOUNG YOUN<sup>1</sup>, YOUSIK LEE<sup>2</sup>, AND SAMUEL WOO<sup>3</sup>,

<sup>1</sup>Information Security Research Division, Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea

<sup>2</sup>ESCRYPT GmbH, Gyeonggi 13488, South Korea

<sup>3</sup>Department of Software Science, Dankook University, Gyeonggi 16891, South Korea

Corresponding author: Samuel Woo (samuelwoo@dankook.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant through the Korea Government (MSIT) under Grant NRF-2020R1G1A1004303, and in part by the Electronics and Telecommunications Research Institute (ETRI) Grant through the Korean Government (Development of Core Technologies For Trust Data Connectome to Realize a Safe Data Society) under Grant 20ZR1300

**ABSTRACT** Recently, Various security techniques are developed to provide security for In-vehicle CAN, but the limited characteristics of CAN protocol make it hard to apply them to a real vehicle. In this paper, we propose a sender authentication and key management schemes considering the limitations of In-vehicle CAN. Our proposed security scheme is designed considering the computing power of automotive ECU and the limited-size CAN data frame. Further, we suggest an efficient key management scheme causing no communication overhead in a session key update process. The security scheme has a structure that may be implemented without change of CAN standards. To evaluate the performance and security of the proposed scheme, we conduct hardware and network simulator based evaluation. Finally, through the analysis on the security and performance, we prove that our proposed scheme is suitable for solving the problem of In-vehicle CAN authentication.

**INDEX TERMS** In-vehicle CAN security, automotive security, key management, data authentication.

## I. INTRODUCTION

Various types of electronic control units (ECUs) are being loaded into modern vehicles for safety and convenience [1]. In particular, the number of ECUs loaded is increasing rapidly with the active development of autonomous vehicles. As a result, vehicles safety and convenience are being improved. However, security threats are also increasing sharply. The representative problem of an In-vehicle controller area network (CAN) is that it fails in providing entity and message authentications [2]. Accordingly, most vehicles are exposed defenselessly to impersonation and message replay attacks. Many studies with a view on solving the authentication problem of In-vehicle CAN have been conducted in the last decade [3], [4] [5]. However, the security countermeasures suggested by these studies have the following limitations.

- 1) Data frame overhead: a CAN data frame may transmit data of only 8 bytes at a time. To use a message authentication code (MAC), an additional data frame has to

be transmitted (data frame overhead occurred, bus load increased).

- 2) CAN standard modification: there are security countermeasures transmitting MAC using a cyclic redundancy check (CRC) field. The CRC field defined by the CAN standard cannot be used for anything else other than CRC. A new type of CAN standard should be developed to apply these countermeasures to CAN.
- 3) Tradeoff between security and availability: a countermeasure transmitting MAC was proposed using the Extended-ID (EXID) field in order to solve problems 1 and 2. However, the EXID field is too small to use a MAC of the size recommended by the security standard. When using a small sized MAC, a key update protocol has to be implemented frequently to enhance security, which results in rapidly increasing communication overhead.

Hence, carmakers still fail to give a perfect countermeasure for vulnerabilities of In-vehicle CAN. Such countermeasures may be applied to real vehicles only when the following requirements are met.

The associate editor coordinating the review of this manuscript and approving it for publication was IlSun You<sup>1</sup>.

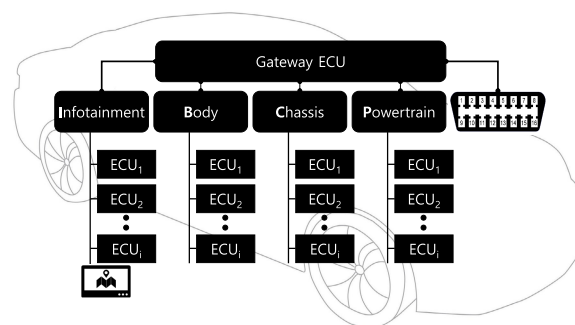
**TABLE 1.** Type and characteristics of the representative subsystems.

Subsystem	Characteristics	Network Protocol
Powertrain	Electronic control system producing the power of a vehicle. In case of malfunctioning, normal driving is not possible.	High Speed CAN, FlexRay, CAN-FD
Chassis	Electronic control system controlling vehicular motions. A safety system is included. In case of malfunctioning, normal driving is not possible.	High Speed CAN, FlexRay, CAN-FD
Body	Electronic control system controlling seats, doors, and mirrors. Mainly used for constructing convenient functions of cars.	Low Speed CAN, LIN
Infotainment	Electronic control system related to multimedia like navigation, CD/DVD/MP3 players.	MOST, Ethernet

- An additional data frame shall not be transmitted for authentication.
- A secured and efficient session key update scheme shall be provided.
- An authentication countermeasure shall be able to be used without modification of the CAN standard.

In this paper, we propose a sender authentication and key management schemes that satisfy these three points. Our proposed scheme using the pre-computable short authenticator(PreAuthCode) is a very practical scheme that can prevent the impersonation and replay attacks. The PreAuthCode can only generate legitimate entity and cannot reuse them. It transmits the PreAuthCode to the receiver using the EXID field, so it does not generate an additional data frame. Furthermore, our proposed key management scheme is very efficient in preventing both communication overheads and increases of the bus load. We conducted an evaluation to analyze security and performance of our proposed scheme. The evaluation environment was constructed using CANoe, which is mostly used at the development of the automotive Electrical and Electronic (E/E) system. We proved that our proposed security scheme ensures both security and availability through the results from the evaluation. The main contributions of this study are as follows.

- It presents a sender authentication scheme able to incapacitate the impersonation and replay attacks. To design a suitable countermeasure, we first analyze known attacks against In-vehicle CAN communication and define a security model that covers all possible attacks. Then, we examine the security of our scheme under the security model.
- It suggests the efficient key management scheme considering the limitations of In-vehicle CAN and ECU. Different from existing techniques, our scheme supports the seamless session key update scheme, which does not have high costs for operation and communication. The scheme does not require additional communication to update the session key and the key updating operations are sufficiently efficient to be implemented without influencing the performance of In-vehicle CAN communication.
- It proves the availability of the proposed scheme through a performance evaluation. We used XC2265N,



**FIGURE 1.** Vehicular E/E system and network environment.

ATmega328, and CANoe to build an experimental environment similar to a real vehicle. The XC2265N is a popular MCU for vehicular ECU manufacturing. The CANoe is the most used SW in vehicular network simulations.

## II. BACKGROUND

### A. AUTOMOTIVE E/E SYSTEM AND IN-VEHICLE NETWORK

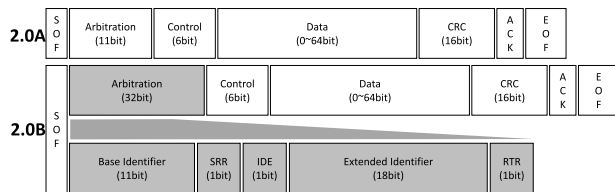
One or more ECUs construct an automotive E/E system performing electronic control. More than one E/E system constructs the main subsystems (e.g., powertrain, chassis, Body, and infotainment) [6]. Table 1. shows the types and characteristics of the representative automotive subsystems. Each subsystem uses different communication protocols depending on their functionality. In general, FlexRay or Media Oriented Systems Transport (MOST) are used for the automotive infotainment system that requires high-speed communication. Moreover, for the powertrain and chassis system requiring real-time control, high speed CAN is used. Figure 1 shows the In-vehicle network organization.

### B. CONTROLLER AREA NETWORK

A CAN is a message ID-based broadcast communication protocol [7]. The CAN uses CSMA/CD+AMP (Carrier Multiple Access with Collision Detection + Arbitration on Message Priority) for efficient communication [8]. Because CAN uses the ID field to perform the arbitration process, all messages shall use a unique ID field. The same ECU can send several messages with different IDs and receive different messages with different IDs. An ID field is to be changed dynamically

**TABLE 2. Characteristics and vulnerabilities of modern vehicles.**

Technology	Characteristics	Vulnerabilities
Increasing the use of E/E systems	Various communication protocols are used for communication among ECUs.	CANs are widely used but do not offer information security functions. Data frame sniffing and injection and replay attacks may be done for In-vehicle CAN. (Root cause of cyber-attack) [10] [11]
	A variety of diagnostic tool are used for automotive E/E systems and communication lines	A data frame used for forced controlling of certain ECU is stored in a diagnostic tool. A data frame used for the replay attack may be acquired easily. (acquisition of important information to be used for cyber-attack) [10] [11]
Increasing the use of third party devices	A vehicle owner may install various third-party devices to his/her vehicle	An attacker can manufacture/distribute a third-party device loading a malicious code. Malicious ECU can inject a data frame into In-vehicle CANs. (possible to manufacture/distribute an attack tool to be used for cyber-attack) [16] [17]
Commercialization of connected Car	Popularization of telematics ECU based connected car services	Remote control attacks are possible using vulnerabilities of connected car services. An attacker can install malicious software to telematics ECU remotely, whereby a malicious data frame may be injected into In-vehicle CANs. (possible to manufacture an attack tool to be used for cyber-attacks) [12] [13] [14] [15]
	Increase of auto service using Smart Phone Apps [16] [17]	An attacker can manufacture/distribute malicious Apps. Malicious Apps can inject a malicious data frame into In-vehicle CANs. (possible to manufacture an attack tool to be used for cyber-attacks) [18]



**FIGURE 2. CAN data frame format.**

to use at a higher layer protocols. CANs may be divided into two modes according to the length of the arbitration (ID) field.

- CAN 2.0A: 11bit arbitration field
- CAN 2.0B: 29bit arbitration field

In CANs, data of a maximum of 8 bytes may be transmitted using a single data frame. The CRC is used to detect a bit error occurred from a data frame transmission process. The other fields are not related to our work and hence will not be explained. Figure 2. shows a CAN data frame format.

**III. ATTACK MODEL**

ENISA classified the characteristics of car hacking into 4 categories [9]:

- 1) Remote attack: forced control attack on a car
- 2) Persistent vehicle alteration: car tuning and CAN data analysis attack
- 3) Theft: theft of a vehicle
- 4) Surveillance: surveillance of car position (closely related to privacy issues)

A remote attack is a very serious attack as it directly threatens the lives of drivers and passengers. Studies on car hacking carried out since 2010 have revealed that vehicles may be exposed to remote attack because of vulnerabilities of In-vehicle CANs. In particular, C. Miller *et al.* executed a

remote attack on a connected car. In this section, we analyze the characteristics and vulnerabilities of the latest vehicles and define the conceptual attack model.

**A. CHARACTERISTICS AND VULNERABILITIES OF MODERN VEHICLES**

With the increase of E/E systems loaded on vehicles, their security vulnerabilities increased along with their safety and convenience. Table 2. shows some characteristics and vulnerabilities of modern vehicles. Such characteristics and vulnerabilities are of great advantage to an attacker, allowing him/her to perform cyber-attacks.

**B. CONCEPTUAL ATTACK MODEL**

Vehicle hacking uses different attack surfaces but authentication vulnerabilities of CAN are always used at the final step of an attack. The method of injecting a malicious data frame into an In-vehicle CAN may be classified into three categories. Figure 3. shows a proposed conceptual attack model. Three attack types use authentication vulnerabilities of In-vehicle CAN and perform an impersonation attack and a replay attack.

- 1) Firmware modification of the devices composing the electronic control system of the vehicle (wired attack, wired + wireless attack).
- 2) Installation of malicious third party ECU after manufacturing (wired attack).
- 3) Wired/wireless access through the OBD2 terminal (wired attack, wired + wireless attack).

The first attack type is very difficult to prevent because it requires the firmware reverse engineering technique. If the firmware of the ECU is compromised, a sender authentication technique cannot prevent impersonation and replay attacks. (Note: There are different countermeasures such as digital sig-

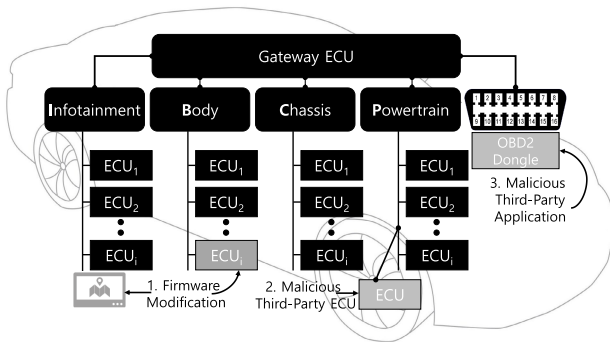


FIGURE 3. Conceptual attack model.

nature on firmware, secure boot, secure flashing etc. to prevent attackers from loading arbitrary firmware on ECUs. In this paper, we do not consider this type of technique.) In this study, we propose a sender authentication scheme to prevent the attack types 2 and 3. We do not consider denial-of-service or man-in-the-middle attacks on the In-vehicle CAN (an attacker cannot change the communication line of In-vehicle CAN).

#### IV. PRE-COMPUTABLE SHORT AUTHENTICATOR (PREAUTHCODE) BASED SENDER AUTHENTICATION SCHEME

##### A. DESIGN GOAL AND ASSUMPTIONS

The proposed scheme is provided to construct an In-vehicle CAN environment secured from impersonation and replay attacks. The proposed scheme must satisfy the following requirements:

- The unique characteristics of the CAN protocol shall not be damaged.
- The sender authentication process shall not increase the bus load.
- The sender authentication process shall guarantee real-time data processing.
- The session key to be used for sender authentication shall be managed securely.
- A secure and efficient session key update function shall be provided.
- The session key update process shall not increase the bus load.

In our proposed scheme, we make the following assumptions

- We consider the one-way communication between a sender  $ECU_s$  and a set of receivers  $RCV$ , which are suitable for the basic features of CAN communication.
- The  $ECU_s$  has a long-term secret key  $SK_s$ , shared with the  $RCV$  to generate and update a new key for each session in an authenticated way.
- A Long-term secret key is stored in the  $ECU_s$  and  $RCV$  by a vehicle manufacturer in the vehicle production process.
- Issues on Long-term secret key are not dealt with in this paper.

TABLE 3. Notation used for proposed scheme.

Notation	Description
$ECU_s$	Sender ECU, "s" is the message ID used by the sender ECU.
$ECU_r$	Receiver ECU, "r" is the message ID used by the Receiver ECU.
$RCV$	Set of a receiver ECU belonging to the same sub-network
$\alpha$	Number of receiver ECU ( $ECU_r$ ) in the same sub-network
$SEED_k$	Seed value of $k_{th}$ session. 32 bit SEED
$SK_s$	Long-term secret key between $ECU_s$ and $RCV$
$SSK_{s,0}$	Initial session key of the sender ECU ( $ECU_s$ )
$SSK_{s,k}$	Session key used for ( $ECU_s$ ) authentication tag generation in the $k_{th}$ session
$Auth_{s,k,j}$	$ECU_s$ authentication tag (PreAuthCode), When $ECU_s$ transmits the $j_{th}$ data frame in the $k_{th}$ session
$CTR_s$	Sender ECU ( $ECU_s$ ) data frame counter
$max_c$	$max_c$ is the upper bound of data frame counter ( $CTR_s$ )
$HKDF_x()$	Keyed one-way function used for key derivation $HKDF_x : \{0,1\}^* \times key \rightarrow \{0,1\}^{256}$
$HMAC_x$	Keyed one-way function using $x$ $HMAC_x : \{0,1\}^* \times key \rightarrow \{0,1\}^{256}$

- We assume that legitimate ECUs belonging to the same sub-network do not attack each other.
- The session key distribution has to be performed before a vehicle starts.
- The  $ECU_s$  and  $RCV$  manage a data frame transmission counter.

##### B. PROPOSED SCHEME

Our proposed scheme consists in three main steps. 1) initial session key distribution, 2) seamless session key update, and 3) sender ECU authentication using a PreAuthCode. In addition, we propose optimization methods to increase the efficiency of our proposed scheme. The notation used in this paper is listed in Table 3.

###### 1) INITIAL SESSION KEY DERIVATION

After starting a vehicle, the  $ECU_s$  selects a seed ( $SEED_0$ ) and computes an authentication code ( $MAC_1$ ) for the  $SEED_0$  as in (1). Then, the  $ECU_s$  computes the initial session key as in (2). We use an 32-bit truncated MAC for  $MAC_1$ .

$$MAC_1 = HMAC_{SK_s}(SEED_0) \quad (1)$$

$$SSK_{s,0} = HKDF_{SK_s}(SEED_0 || 0) \quad (2)$$

After generating the initial session keys, the  $ECU_s$  broadcasts  $MAC_1$  and  $SEED_0$ . When every  $ECU_r$  receives  $MAC_1$  and  $SEED_0$ , they perform an initial session key derivation process with  $ECU_s$  in a fixed order. The  $ECU_r$  verifies the given  $MAC_1$  using the shared key  $SK_s$  and given seed  $SEED_0$ . If the given  $MAC_1$  is correct, the receiver accepts  $SEED_0$  as a valid seed information for the initial session key and computes the initial session key as in (2). The  $ECU_r$  generates  $MAC_r$  and sends it to the  $ECU_s$ . We use an 64-bit truncated

MAC for  $\text{MAC}_r$ . If a given  $\text{MAC}_r$  is correct, the  $\text{ECU}_s$  finishes the initial session key derivation process with the  $\text{ECU}_r$ .

## 2) SEAMLESS SESSION KEY UPDATE

To permit  $\text{ECU}_s$  and  $\text{ECU}_r$  to update their session key, we support the uni-directional session key update protocol instead of giving the interactive session key establishing protocol.

In existing works, the session key could be established by performing interactive protocols, which are not suitable for seamless data frame authentication in CAN communication because they require bi-directional message transmission, which is not easy to be implemented in CAN communication. In our approach, a session key is used for  $\text{max}_c$  data frames, where  $\text{max}_c$  is the upper bound of the data frame counter. It uses the session key and counter to authenticate the sender who sent each data frame. We will explain how to generate a PreAuthCode in the next session; in this section, we will describe the session key update scheme. In our approach, for the  $k$ th session, the new session key  $\text{SSK}_{s,k}$  is derived from the previous session key  $\text{SSK}_{s,k-1}$ . To determine the timing of the session key update, we maintain a data frame counter  $\text{CTR}_s$ , which is increased by 1 at each data frame generation. If the current counter reaches the upper-bound, i.e.,  $\text{CTR}_s = \text{max}_c$ , the  $\text{ECU}_s$  updates the session key as in (3)

$$\text{SSK}_{s,k} = \text{HKDF}_{\text{SK}_s}(\text{SSK}_{s,k-1} || k) \quad (3)$$

Then, the  $\text{ECU}_s$  initializes the counter by setting  $\text{CTR}_s = 0$  and uses the new session key for the upcoming data transmissions until the initialized counter reaches the upper-bound. Recall that  $\text{ECU}_s$  and  $\text{ECU}_r$  can maintain the common counter  $\text{CTR}_s$ , which guarantees the correct timing of the old session key update. Then, the  $\text{ECU}_s$  can use the new session key without confirming whether all the  $\text{ECU}_r$  have updated the session key.

## 3) PRE-COMPUTABLE SHORT AUTHENTICATOR(PREAUTHCODE)

We support a sender authentication by using a one-time token(PreAuthCode) computed from the session key and the data frame counter. For each data frame, we will include the PreAuthCode to prove that the data frame is generated by the session key holder for the data frame, which is identified by the data frame counter. Because the  $\text{ECU}_s$  and  $\text{RCV}$  can correctly maintain the current counter, we can guarantee the entity authenticity by generating a PreAuthCode for the current counter based on the pre-shared session key. Specifically, the procedure is as follows. Let  $\text{CTR}_s$  be the current counter in the  $k$ th session; then, the unit  $\text{ECU}_s$  computes a short authenticator for the  $j$ th data frame in the  $k$ th session as in (4).

$$\text{Auth}_{s,k,j} = \text{HMAC}_{\text{SSK}_{s,k}}(\text{CTR}_s) \quad (4)$$

The  $\text{ECU}_s$  sends the authenticator with a data frame. Our proposed scheme uses the EXID field for  $\text{Auth}_{s,k,j}$ . We use an 18-bit truncated MAC to transmit data and  $\text{Auth}_{s,k,j}$

one time. The  $\text{ECU}_r$  can verify a given short authenticator ( $\text{Auth}_{s,k,j}$ ) by comparing it with  $\text{HMAC}_{\text{SSK}_{s,k}}(\text{CTR}_s)$ .

Based on the above procedure, each  $\text{ECU}_r$  is certain that the given data frame is actually the  $j$ th data frame in the  $k$ th session. To accelerate the authentication procedure, we can pre-compute a set of PreAuthCode ( $\text{Auth}_{s,k,j}$ ) before generating the data frame, because the  $\text{Auth}_{s,k,j}$  does not include data frames related information.

## 4) OPTIMIZATION

Recall that we generate a short authenticator for each data frame transmission. The fundamental feature required for the security of the scheme is the unpredictability of the token. We utilize a hash function to generate a secret information, and truncate 18-bits of the output of the function, because an 18-bit randomness is sufficient against on-line guessing attacks. Here, we will show a trick that can improve the performance of the proposed scheme. As already mentioned, we use only 18-bits of the output. However, the remaining part is still unpredictable from an adversary's viewpoint. Our strategy is to use the remaining part for the authentication of subsequent data frames. When we use a SHA-256, we can use an output of the function to authenticate 14 data frames. As a result, our simple trick can reduce the cost of hash evaluation by 92.9%.

We can also improve the performance of cryptographic operations. We describe our scheme using hash-based primitives such as HMAC and naive hash function; encryption based primitives can be applied for higher performance. As shown in [19], VMAC(AES)-128 supports tens of times efficient message authentication code generation than the above mentioned hash-based functions. In our scheme, the HMAC can be replaced by a VMAC(AES)-128. When the VMAC(AES)-128 is used as a one-way function instead of the naive hash function HMAC to generate 18-bits authenticating information, an output can be used for 7 data frames.

## V. SECURITY AND PERFORMANCE ANALYSIS

### A. SECURITY ANALYSIS

In this section, we analyze the security of our proposed scheme. Recall that, as we explained in section 3, an adversary's goal is to insert a valid data frame, which is not generated by any of the legitimate units (impersonation and replay attacks). Hence, to prove the security of the proposed scheme, we will show that the proposed technique is secure against an impersonation attack and a replay attack.

*Theorem:* The proposed authenticating scheme is secure against impersonation attack (including the replay attack) conducted by CAN-adversary if the underlying hash function and MAC are secure under standard notions of security. Specifically, in an adversary's viewpoint, the only way to perform the impersonation attack is to guess  $\ell$ -bit authenticating message without knowing secret input information ( $\text{SSK}_{s,k}$ ) with probability higher than  $1/2^\ell$ .

In this proof, we will consider the case where the adversary tries to forge a message of the unit  $\text{ECU}_s$ . Recall that the goal

**(A) Initial Session Key Derivation**ECU<sub>s</sub>

Select a random seed SEED<sub>0</sub>  
 Generate MAC<sub>1</sub> = HMAC<sub>SK<sub>s</sub></sub>(SEED<sub>0</sub>)  
 Compute SSK<sub>s,0</sub> = HKDF<sub>SK<sub>s</sub></sub>(SEED<sub>0</sub>||0)

MAC<sub>1</sub>, SEED<sub>0</sub>ECU<sub>r</sub>

Verify MAC<sub>1</sub>  
 If MAC<sub>1</sub> is invalid, stop the protocol  
 Compute SSK<sub>s,0</sub> = HKDF<sub>SK<sub>s</sub></sub>(SEED<sub>0</sub>||0)  
 Generate MAC<sub>r</sub> = HMAC<sub>SSK<sub>s,0</sub></sub>(r)

MAC<sub>r</sub>

Verify MAC<sub>r</sub> for all r  
 If MAC<sub>r</sub> is invalid for any r, stop the protocol

**(B) Session Key Update (k<sup>th</sup> Session for ECU<sub>s</sub>)**ECU<sub>s</sub>

If CTR<sub>s</sub> == max<sub>c</sub>,  
 compute SSK<sub>s,k</sub>

ECU<sub>r</sub>

If CTR<sub>s</sub> == max<sub>c</sub>,  
 compute SSK<sub>s,k</sub>

where SSK<sub>s,k</sub> = HKDF<sub>SK<sub>s</sub></sub>(SSK<sub>s,k-1</sub>||k)

**FIGURE 4. Proposed key management protocols.**

of an adversary  $\mathcal{A}$  is to impersonate an unit ECU<sub>s</sub> without possessing (SSK<sub>s,k</sub>) and the adversary succeeds in the attack if it can generate a valid PreAuthCode for a new data frame. To evaluate the adversary's advantage, we define **Succ** as the event that  $\mathcal{A}$  succeeds in the impersonation attack. The impersonation attack against CAN communication can be classified into two types depending on the forged data frame. An already used or new data frame can be used as a forged data frame. To deal with each case, we define the following two events. Let  $E_e$  be the event that the forged data frame belongs to **List** and  $E_n$  be the event that the forged data frame does not belong to **List**, where **List** is the list of all data frames with PreAuthCode generated by the unit ECU<sub>s</sub>.

Because an attack belongs to either  $E_e$  or  $E_n$ , we have  $\Pr(E_e) = p$  and  $\Pr(E_n) = 1 - p$  for some  $p \in [0, 1]$ . Then, the advantage of the adversary  $\mathcal{A}$  can be defined as in (5), and it can be rewritten as in (6)

$$\text{Adv}(\mathcal{A}) = \Pr(\text{Succ}), \quad (5)$$

$$\text{Adv}(\mathcal{A}) = p \cdot \Pr[\text{Succ}|E_e] + (1-p) \cdot \Pr[\text{Succ}|E_n] \quad (6)$$

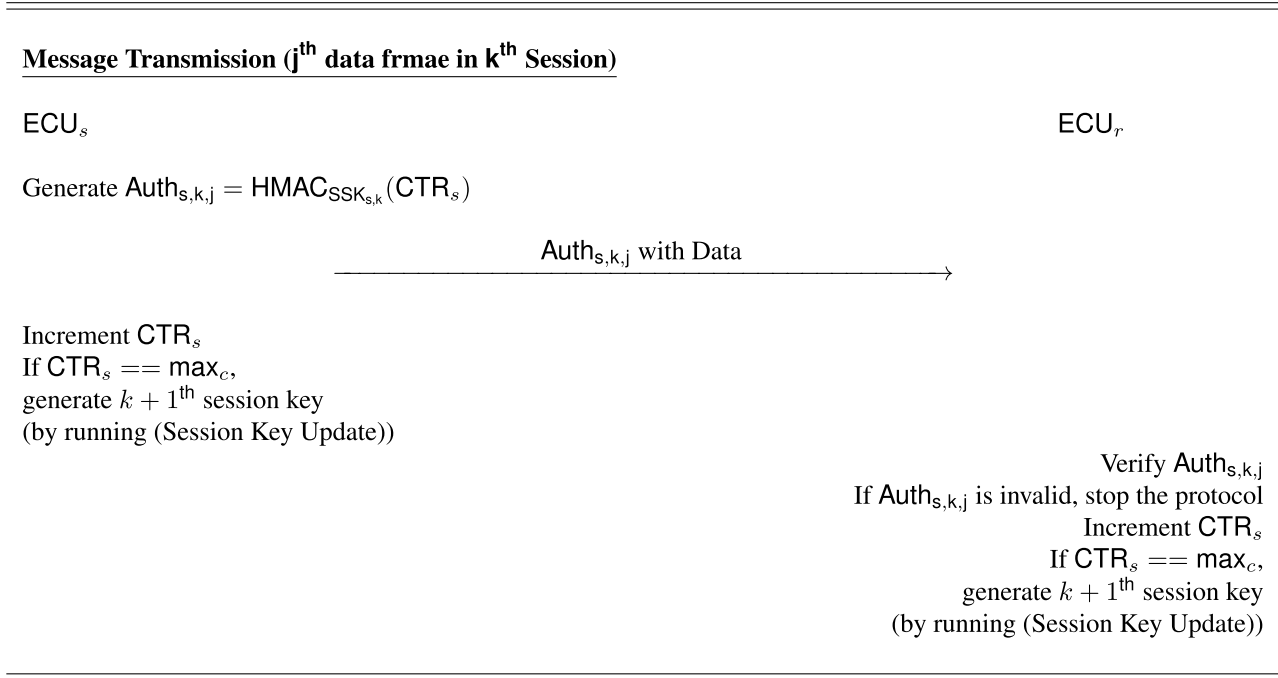
In the formula,  $\Pr[\text{Succ}|E_e]$  is the probability of successful attack with an existing data frame with a PreAuthCode. Similarly,  $\Pr[\text{Succ}|E_n]$  is the probability of successful attack with a new data frame with a PreAuthCode. In the rest of the section, we evaluate the possibility of successful attack for each case.

**[Use Existing data frame for Impersonation Attack]**

In this case, the adversary's strategy to mount the impersonation attack is to monitor the transmitted data frame and use one of them as a forged data frame. Before mounting the attack, the adversary can collect sufficient data frames with the corresponding PreAuthCode. Because the size of the PreAuthCode is not sufficiently large to guarantee the collision resistance, there may be collisions such that:

$$\text{Auth}_{s_1, k_1, j_1} = \text{Auth}_{s_2, k_2, j_2} \quad \text{for } \{s_1, k_1, j_1\} \neq \{s_2, k_2, j_2\} \quad (7)$$

Note that, the existence of collisions implies that the adversary can use an existing PreAuthCode for later data frames. For simplicity, we assume that the adversary obtains  $N \in$



**FIGURE 5. Proposed authentication protocol.**

[1, 2<sup>ℓ</sup>] data frame with different PreAuthCode, which means that there are  $N$  data frames in List. Suppose that an adversary uses an existing PreAuthCode Auth<sub>s,k',j'</sub> ∈ List as a forgery for the  $j$ -th data frame in the  $k$ -th session such that T<sub>s,k,j</sub> > T<sub>s,k',j'</sub> where T<sub>α,β,γ</sub> is the time when the PreAuthCode for the data frame Data<sub>α,β,γ</sub> is generated. Let Auth<sub>s,k,j</sub> be the correct PreAuthCode for the  $j$ -th data frame in the  $k$ -th session. The probability Pr[Succ|E<sub>e</sub>] can be rewritten as in

$$\Pr[\text{Succ}|E_e] = \Pr[\text{Auth}_{s,k',j'} = \text{Auth}_{s,k,j}] \quad (8)$$

Recall that the adversary's strategy is to use an existing packet in List, and thus we have

$$\Pr[\text{Auth}_{s,k',j'} = \text{Auth}_{s,k,j} | \text{Auth}_{s,k,j} \notin \text{List}] = 0 \quad (9)$$

Hence, we can see that Pr[Succ|E<sub>e</sub>] is identical with

$$\Pr[\text{Auth}_{s,k,j} \in \text{List}] \cdot \Pr[\text{Auth}_{s,k',j'} = \text{Auth}_{s,k,j} | \text{Auth}_{s,k,j} \in \text{List}] \quad (10)$$

Recall that we use a hash function to generate the PreAuthCode, and the probability Pr[Auth<sub>s,k,j</sub> = σ] = 2<sup>-ℓ</sup> for all σ ∈ {0, 1}<sup>ℓ</sup> since the hash function works as a random oracle. Hence, we have

$$\Pr(\text{Auth}_{s,k,j} \in \text{List}) = \frac{N}{2^\ell} \quad (11)$$

and

$$\Pr(\text{Auth}_{s,k',j'} = \text{Auth}_{s,k,j} | \text{Auth}_{s,k,j} \in \text{List}) = \frac{1}{N} \quad (12)$$

Therefore, we can see that

$$\Pr(\text{Succ}|E_e) = \frac{1}{2^\ell} \quad (13)$$

**[Generate New Packets for Impersonation Attack]**

Recall that, in our scheme, we use ℓ-bit information as a PreAuthCode which is computed for the  $j$ -th data frame in  $k$ -th session as following:

$$\text{Auth}_{i,j,k} = H(\text{SSK}_{i,j} || k) \quad (14)$$

As we can see in the above equation, to generate a valid PreAuthCode, a session key for the  $k$ -th session is required. Because the hash function works as a random oracle, it is not possible to guess results of the function before running the function on an input. Moreover, in the generation of the PreAuthCode, the  $k$ -th session key secretly kept by the unit ECU<sub>s</sub> is required as an input. Therefore, without the secret input value, it is not possible to guess even a single bit. In the event E<sub>n</sub>, the adversary can succeed in the impersonation attack if it can generate a valid forgery without using existing data frames. Hence, in the adversary's viewpoint, the only way is to guess a ℓ-bit PreAuthCode due to the randomness of the PreAuthCode generation method. If the adversary can generate valid PreAuthCode without randomly guessing the value, it implies the contradictory statement that the adversary can also predict the output of the random oracle implemented using the underlying hash function. Hence, we have

$$\Pr[\text{Succ}|E_n] = \frac{1}{2^\ell} \quad (15)$$

As shown in the above, we have Pr(Succ|E<sub>e</sub>) = 1/2<sup>ℓ</sup> and Pr[Succ|E<sub>n</sub>] = 1/2<sup>ℓ</sup>, thus, we obtain

$$\text{Adv}(\mathcal{A}) = p \cdot \frac{1}{2^\ell} + (1 - p) \cdot \frac{1}{2^\ell} = \frac{1}{2^\ell} \quad (16)$$

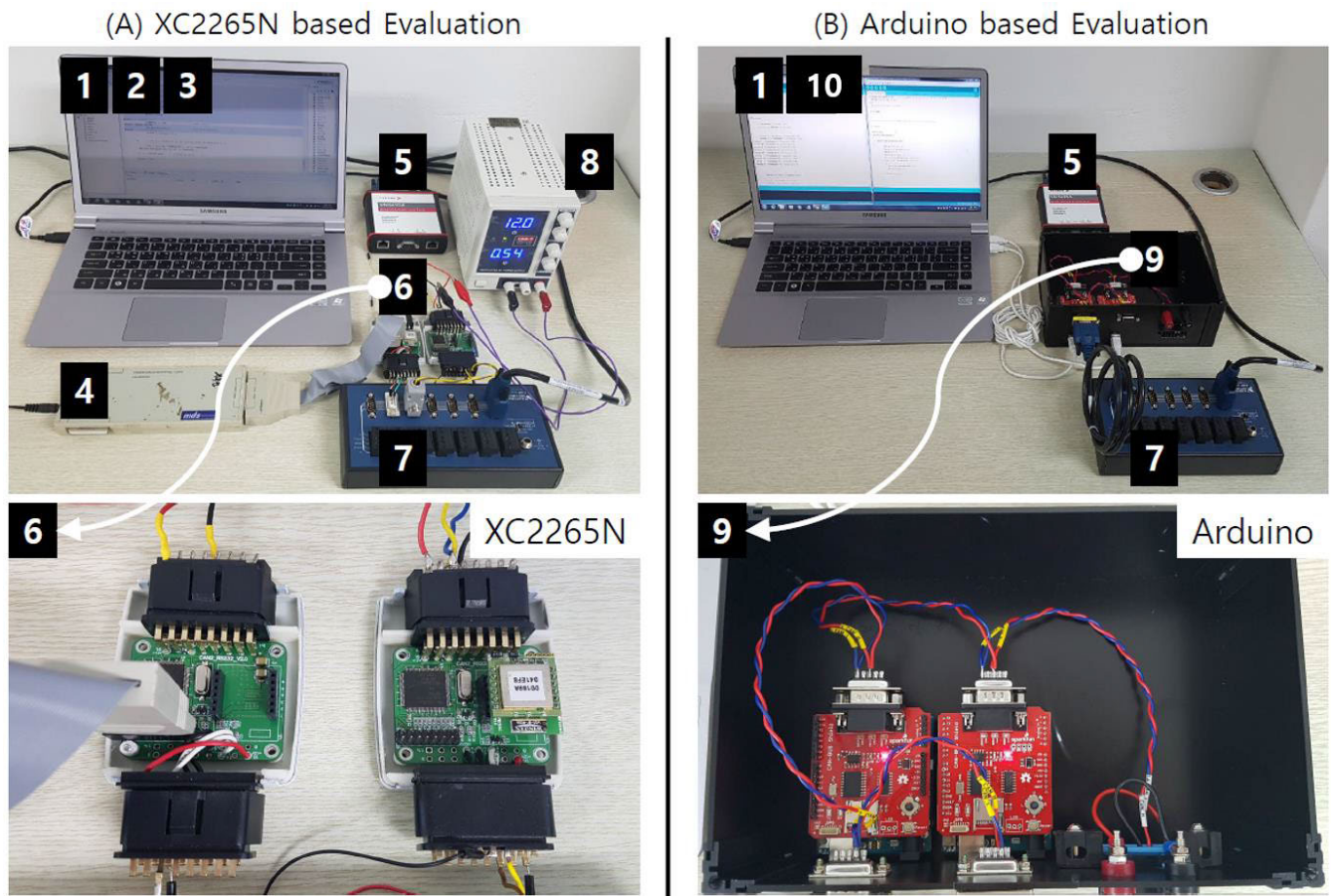


FIGURE 6. Performance evaluation environment.

The equation implies that the adversary's advantage is identical with the probability of random guessing. Hence, the proposed scheme guarantees  $\ell$ -bits security against impersonation attack.

In the above theorem, we proved the security level of the proposed technique against impersonation attacks. However, until now, it is not clear whether the size is sufficient for the security of the proposed scheme because  $\ell$ -bits PreAuthCode seem too short from the viewpoint of modern cryptography. However, in cryptography, we already utilized a number of short secret values as authentication information. For example, we use 40-bits information as passwords in many online services. The size of passwords is very small compared with ordinary cryptographic private information, but it guarantees sufficient security in real applications because the number of adversarial trials against a password is limited. If an adversary fails to forge a password 3 or more times, the service provider can protect the security by blocking the use of the password. Smaller information can be used for authentication. In OTP systems, a random password is used only once, unlike passwords. Hence, in the literature, it is known that 6-decimals one-time tokens can be used in practical applications, including security sensitive services such as e-banking [20]. Because  $10^6 \approx 2^{19}$ , we can see that

the security level of the proposed scheme is sufficient for resisting impersonation attacks in CAN communication.

## B. PERFORMANCE EVALUATION

### 1) EVALUATION ENVIRONMENT

To evaluate the performance of the proposed scheme, we performed a hardware-based performance evaluation and a network simulator-based performance simulation. The performance evaluation environment is shown in Figure 6 and the characteristics of the hardware and software used for the performance evaluation are listed in Table 4.

#### [Hardware-based performance evaluation:]

using an XC2265N-based and an ATmega328-based evaluation board, we measured the execution time of the cryptographic algorithm used in the proposed scheme.

#### [Network simulator-based performance evaluation:]

using the CANoe and the evaluation board used for the hardware-based performance evaluation, we measured the performance of the proposed key management method and the data frame authentication scheme. To evaluate the performance of the key management scheme, we measured the time taken by the initial session key distribution process. To measure the performance of the data frame authentication scheme, the communication delay of the process of data

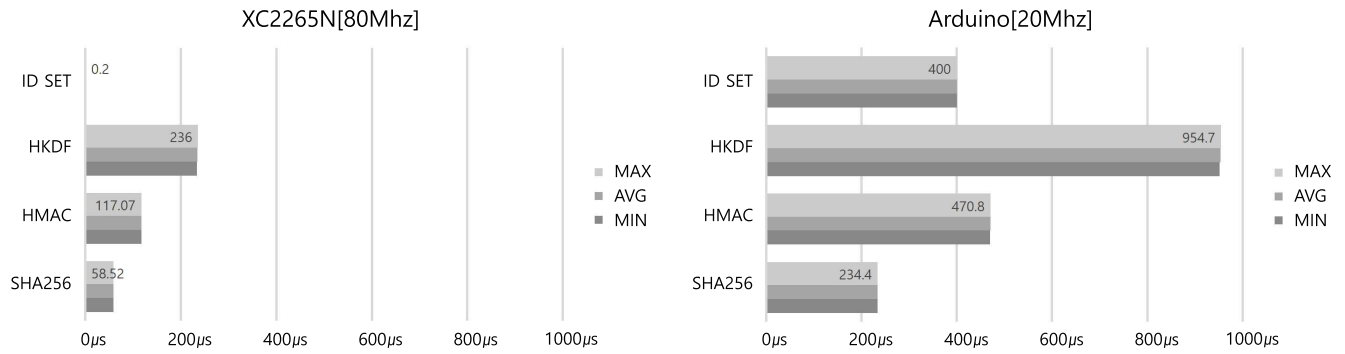


FIGURE 7. Execution time for cryptographic algorithm and CAN ID change.

TABLE 4. Hardware and software used for evaluation.

	Type	Model Name	Note
1	SW1	CANoe V8.5	CAN Network Simulator
2	SW2	Tasking C/C++	For XC2265N, Compiler
3	SW3	Trace32 Powerview	For XC2265N, Debugger
4	HW1	JTAG Emulator	For XC2265N
5	HW2	CANcaseXL	CANoe to CAN BUS
6	MCU1	XC2265N	Clock speed 80 MHz
7	HW3	CAN Breakout Box	CAN BUS Terminal
8	HW4	Power supply	For XC2265N
9	MCU2	ATmega328 (Arduino Uno Rev3)	Clock speed 20 MHz
10	SW4	Arduino IDE	For Arduino, Compiler

frame transmission and receipt by the sender and receiver was measured.

## 2) HARDWARE-BASED EVALUATION

To measure the time taken for the PreAuthCode generation and verification and session key generation, we performed a hardware-based performance evaluation. In our proposed scheme, a SHA-256-based HMAC and HKDF are used for the PreAuthCode and session key tag generation. Because the PreAuthCode is applied to the ID field, the ECU<sub>s</sub> must register the PreAuthCode in its transmission ID (TXID) register. To receive the data frames transmitted by the ECU<sub>s</sub>, the ECU<sub>r</sub> must also register the PreAuthCode of the ECU<sub>s</sub>, which is calculated beforehand, in his receive ID (RXID) register. For ECU<sub>s</sub> and ECU<sub>r</sub> to use the PreAuthCode, it is necessary to use the PreAuthCode to update the ID register of the microcontroller each time a data frame is transmitted and received. We applied the proposed scheme to two types of microcontrollers and then carried out the performance evaluation. The SHA-256, HMAC, and HKDF were used to implement the proposed scheme. To produce accurate results, the three algorithms were run 100,000 times each, measuring the average, maximum, and minimum execution times. Further, the time taken to update the register of the microcontroller was measured.

### [Analysis of execution time for the PreAuthCode generation/verification]

The operations carried out by ECU<sub>s</sub> and ECU<sub>r</sub> transmitting and receiving the  $j_{th}$  data frame in the  $k_{th}$  session and the In-vehicle CAN requirements are the following. ECU<sub>s</sub>: the ECU<sub>s</sub> generates a PreAuthCode (Auth<sub>s,k,j</sub>), updates the TXID register, and then transmits the data frame. Here, the ECU<sub>s</sub> performs the HMAC operation and register update once. ECU<sub>r</sub>: the ECU<sub>r</sub> verifies the PreAuthCode of the ECU<sub>s</sub> and then receives the data. In the process of Auth<sub>s,k,j</sub> verification, only a simple comparison operation is carried out using the Auth<sub>s,k,j</sub>, which was generated beforehand. After the normal receipt of the data, the ECU<sub>r</sub> calculates the Auth<sub>s,k,j+1</sub> beforehand to prepare to receive the next data frame. Here, the ECU<sub>r</sub> performs the HMAC operation and register update once. Figure 7. shows the time required for the HMAC operation and register update. In the case of XC2265N, commonly used for manufacture of automotive ECU<sub>s</sub>, it can be seen that up to 118μs are required for the data transmission and receiving processes. Even without using the proposed optimization scheme, the execution time required for transmission and receipt of one data frame does not exceed 240μs. These results indicate that our scheme may be applied to general In-vehicle CAN environments. In particular, the use of an ECU with integrated HSM, such as the TC275, allows faster use of the proposed authentication scheme [21]. Therefore, the proposed scheme can be said to be a security method that is well suited for use in contemporary vehicles. It was found that for ATmega328, the time required for register update was very long compared to that for XC2265N. Thus, long as the time required for register update can be shortened, the proposed scheme may also be used without particular difficulty in ATmega328-based ECU.

### [Analysis of execution time for session key generation and updating]

In the proposed scheme, the ECU<sub>s</sub> and ECU<sub>r</sub> use a session key to generate a PreAuthCode. The session key is derived through a 2-way handshake process based on a long-term symmetry key. The HKDF is used for session key generation.

While the 2-way handshake process is carried out, the ECU<sub>s</sub> and ECU<sub>r</sub> carry out one MAC generation, one MAC verification, and one key generation. When the session

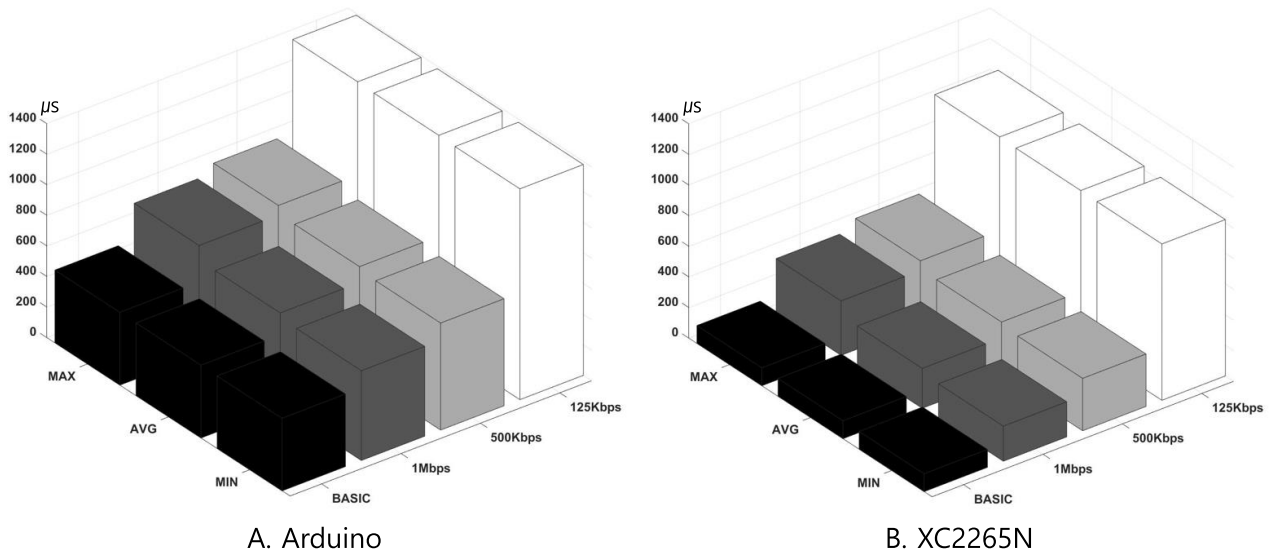


FIGURE 8. Execution time for secure communication.

key is updated, the 2-way handshake is not carried out. The  $ECU_s$  and  $ECU_r$  carry out the update themselves, using the HKDF. As shown in Figure 7, the execution time for the cryptographic algorithm accompanying the session key derivation and update processes can be analyzed. In the case of ATmega328, up to  $3800\mu s$  are required for execution of the cryptographic algorithm accompanying the 2-way handshake process.

For the same operation, the XC2265 takes up to  $940\mu s$ . For session key updates, the ECUs and  $ECU_r$  need to only execute the HKDF operation, without the need for a 2-way handshake process. Performing the HKDF once takes the ATmega328 up to  $960\mu s$  of execution time, while the XC2265N takes  $240\mu s$ .

### 3) NETWORK SIMULATOR-BASED EVALUATION

Using the CANoe and an evaluation board, we implemented an evaluation environment similar to an actual In-vehicle network environment. The CANoe is the network simulator most commonly used for In-vehicle network development and testing [22]. We applied the cryptographic algorithm execution times derived from the hardware-based performance evaluation to the CANoe and performed a network simulator-based performance evaluation. The evaluation board played the role of the  $ECU_s$ , and the virtual nodes generated in the CANoe played the role of the  $ECU_r$ .

#### [Communication Response Time]

We measured the communication delay and varying communication speed. For the communication delay measurement, we defined the following scenario and carried out the performance evaluation. The scenario involves transmitting a  $j_{th}$  data frame in the  $k_{th}$  session as follows:

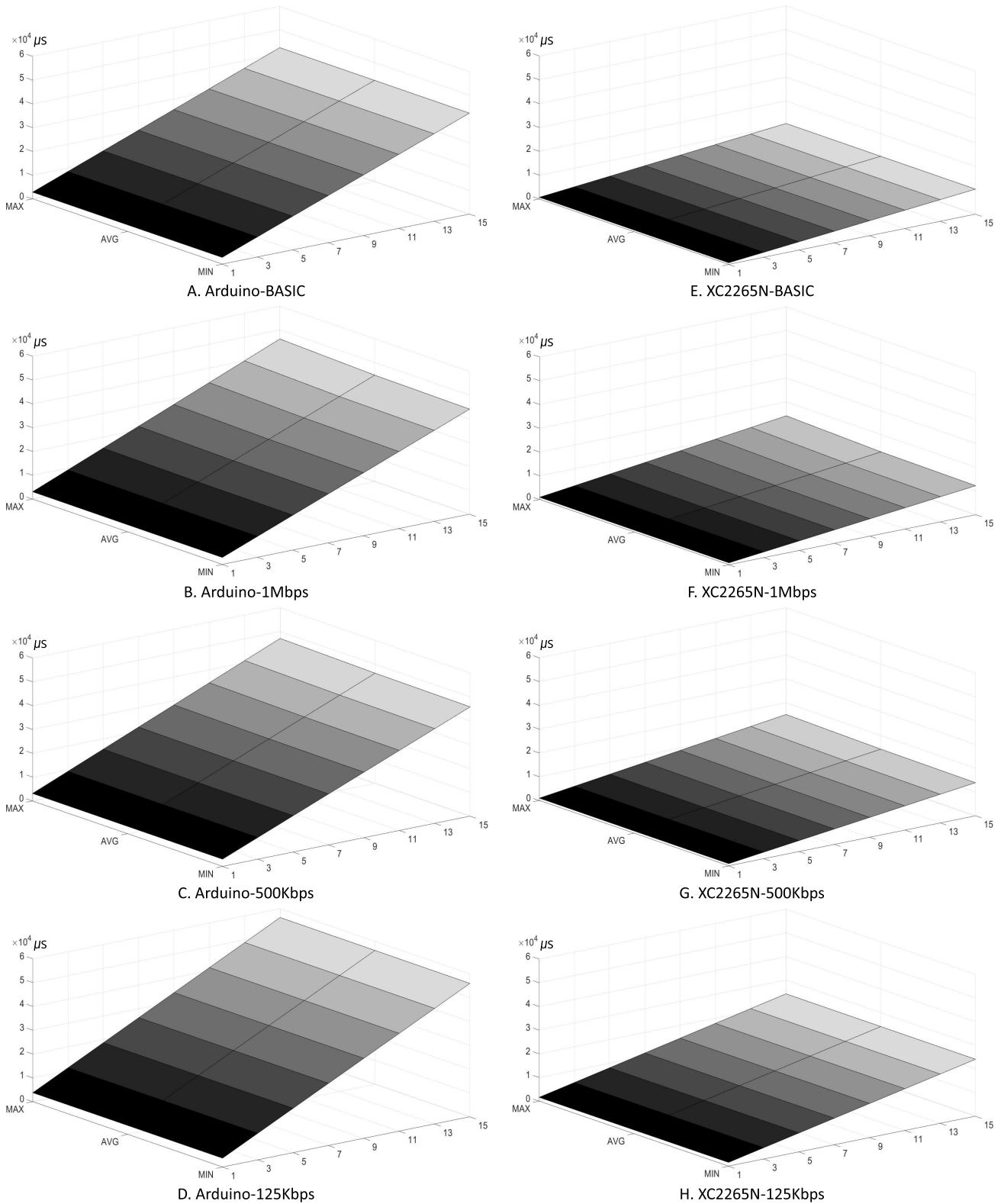
- 1) The  $ECU_s$  generates  $Auth_{s,k,j}$  and updates the TXID register.
- 2) The  $ECU_s$  transmits a request data frame.
- 3) The  $ECU_r$  verifies the request data frame. (Verification is carried out using the  $Auth_{s,k,j}$  generated upon receiving the previously generated data frame)

- 4) After completing the  $Auth_{s,k,j}$  verification, the  $ECU_r$  generates an  $Auth_{s,k,j+1}$  for receipt of the next data frame, then updates the RXID register.
- 5) The  $ECU_r$  generates a response data frame. (In the response data frame, the PreAuthCode is not used)
- 6) The  $ECU_r$  transmits the response data frame.
- 7) The  $ECU_s$  receives the response data frame.

We measured the time taken from the instant the  $ECU_s$  generates the  $Auth_{s,k,j}$  to when the response data frame is received. The results of the performance evaluation are shown in Figure 8. There are two major causes of communication delays when the proposed method is applied to an In-vehicle CAN environment:

- 1) Time taken for PreAuthCode generation and verification by the ECU.
- 2) Time taken for register updates in the ECU.

Of the two, the greatest delay is caused by the operations of generating and verifying the PreAuthCode in the ECU. (The PreAuthCode verification operation is identical to the precomputable short authenticator generation operation). For the  $ECU_s$  to generate the PreAuthCode, the HMAC operation must be carried out once. When the  $ECU_r$  verifies the PreAuthCode as well, the HMAC operation is carried out once. In our experimental scenario, only the  $ECU_s$  uses the PreAuthCode. In the process of data frame transmission/receipt, a total of two HMAC operations are carried out. In an environment having  $ECU_s$  equipped with XC2265N communicating at a speed of 500 Kbps, it was found that carrying out the scenario we defined results in up to  $600\mu s$  of communication delay. Analyzing this result based on the test results of V.B.2, it can be seen that the communication delay caused is equal to the cost of HMAC execution. Using the optimization scheme proposed in section IV.B.4, it is possible to reduce the average time required for the PreAuthCode generation and verification by up to 92.9%.



**FIGURE 9.** Execution time for session key distribution.

That is, it is possible to maintain communication speeds similar to conventional environments in which the PreAuthCode are not used. The second cause of communication delays is the operation of the ECU updating the register. In the

XC2265N and the ATmega328 we used for testing, changing the CAN ID requires updating the CAN ID registers.

In an ECU with XC2265N, the time taken for CAN ID updates is approximately  $0.2\mu s$ , which is negligible

compared to the HMAC execution time. In the case of the ECU with ATmega328, it was found that a substantial delay was caused by updating the CAN register. The process of updating the register is a unique function of the MCU and there is no way of optimizing this process. Considering the results of the hardware-based and software-based performance evaluations together, it is found that the proposed method can be applied to low-spec ECU commonly used in vehicle environments to implement secure communication environments.

### [Session Key Distribution and Updating Time]

While varying the communication speed and number of ECU<sub>r</sub>, we measured the time taken for session key derivation. In the proposed scheme, a 2-way-handshake is carried out for session key derivation. The time required for session key distribution are shown in Figure 9. When a session key distribution process is carried out in a group having 15 ECU<sub>s</sub> with low speed (125Kbps), up to 50400 $\mu$ s and 22800 $\mu$ s are taken by ECU groups with ATmega328 and XC2265N, respectively. By analyzing this results based on the test results of V.B.2, it can be seen that most of the time taken for session key derivation is accounted for by cryptographic algorithm execution.

Thus, shortening this execution time (e.g., by using an ECU with a high CPU clock rate or an ECU with HSM) can help to reduce the time required for the session key derivation. In the proposed session key update method, the ECU<sub>s</sub> and ECU<sub>r</sub> use the HKDF to generate a session key to be used in the following session. In this process, a 2-way handshake is not carried out. Instead, the ECU<sub>s</sub> and ECU<sub>r</sub> independently perform HKDF. The execution time required for a session key update is identical to the time required to perform an HKDF.

### C. SECURITY COMPARISON

In the present section, a comparative analysis is carried out for the scheme we have proposed and the three previous studies analyzed under the related work section. As shown in Table 5, all four methods provide data frame authentication and are able to block impersonation and replay attacks. The Mini-MAC method uses the data payload to transmit the MAC, resulting in data frame overhead. In the other three methods, the CRC and EXID fields are used to transmit the MAC causing no data frame overhead.

However, the Woo-Auth and DDA methods, which use the CRC field, have the serious problem of requiring a change of CAN standard. The other three methods transmit the data and MAC using a single data frame and, therefore, have short communication delays (the communication delay is equal to the time required for MAC generation and verification). However, as authentication is carried out for four data frames at once in the DDA, a substantial communication delay is caused. Lastly, seamless session key update is provided only in the method we propose.

The Woo-Auth method involves repeated execution of 3-way handshakes and HMAC for session key updates requiring a long time for session key updates. Mini-MAC and

TABLE 5. Security comparison.

	OURS	Woo-Auth	Mini-MAC	DDA
Prevent Impersonation attack	O	O	O	O
Prevent Replay attack	O	O	O	O
No data frame overhead	O	O	X	O
No standard change	O	X	O	X
Communication delay	LOW	LOW	LOW	HIGH
Seamless session key update	O	X	?	?

DDA do not propose a specific method for key management. In cases where data is authenticated using a truncated MAC, as in Mini-MAC, session key updates must be carried out at regular intervals to ensure MAC safety. Despite this, the Mini-MAC does not propose a seamless session key update method. In the method we have proposed, the transmitter and the receiver perform session key update individually using synchronized counters, making it possible to carry out seamless session key updates.

### VI. RELATED WORK

With the fast progress of Vehicle-ICT convergence over the last decade, various types of vehicle hacking studies have been published. In these vehicle hacking studies, it was pointed out that hacking was possible because In-vehicle CANs did not provide data authentication. Once this problem was reported, studies to resolve the problem followed. In this section, relevant studies proposing data frame authentication to In-vehicle CAN are introduced.

DDA [5]: D.K. Nilsson *et al.* proposed a Delayed Data Authentication (DDA) method taking the CAN data frame format into consideration. In the DDA method, a method of MAC transmission using CRC fields was proposed. The data transmission-receipt process using the DDA method is as follows:

- 1) The sender-ECU generates a 64-bit MAC for data frame  $n$ ,  $n+1$ ,  $n+2$ , and  $n+3$ .
- 2) After splitting the MAC into 16 bit units, they are inserted into the CRC fields for data frames  $n+4$ ,  $n+5$ ,  $n+6$ , and  $n+7$ .
- 3) The sender-ECU transmits data frames  $n$ ,  $n+1$ ,  $n+2$ , and  $n+3$ .
- 4) Then, data frames  $n+4$ ,  $n+5$ ,  $n+6$ , and  $n+7$  are transmitted.
- 5) The receiver-ECU, after receiving data frames  $n$ ,  $n+1$ ,  $n+2$ , and  $n+3$ , waits for data frame  $n+4$ ,  $n+5$ ,  $n+6$ , and  $n+7$ .
- 6) After receiving data frame  $n+4$ ,  $n+5$ ,  $n+6$ , and  $n+7$ , a MAC verification process is carried out for data frame  $n$ ,  $n+1$ ,  $n+2$ , and  $n+3$ .

It can be seen that the DDA method is designed for data payloads of limited size. However, a substantial delay is caused, as receipt of all data frames up to data frame  $n+7$  must

be completed in order to complete the verification of data frame  $n$ . This is a method that is difficult to apply to an In-vehicle CAN environment, where real-time data processing is crucial. Further, to use the CRC field as the MAC, it is necessary to change the CAN standard. That is, the use of DDA requires the replacement of both the hardware and software of conventional ECUs. As a result, the DDA method cannot be used in In-vehicle CAN environments.

**WooAuth [3]:** Woo *et al.* proposed a data authentication method wherein a truncated MAC is transmitted using an EXID field and a CRC field. The data frame authentication method they propose guarantees real-time data processing and, as no additional data frames are generated, the bus load does not increase. However, the method they propose has two major problems. The first is that the CRC field is used for data frame authentication. As explained in the foregoing, it is practically impossible to use the CRC field for another use. The second is that the key management method they propose is unable to provide seamless session key updates. Further, a 3-way-handshake is carried out for session key update. Here, HMAC execution is carried out some eight times, requiring a very long time.

**Mini-MAC [4]:** Jackson *et al.* proposed a data frame authentication method in which a data field carrying data and the MAC simultaneously is transmitted. Their method ensures real-time data processing, as both the MAC and data are carried in a single data frame. However, using Mini-MAC causes the problem of increased bus load. In the Mini-MAC, part of the data field is used for MAC transmission. As a result, the volume of data frames transmitted inevitably increases. That is, the method has an inherent potential to increase bus load. For communication safety, the bus load must be maintained at no more than 50% in an In-vehicle CAN. Using a 32-bit MAC will double the number of data frames transmitted, sharply increasing the bus load.

## VII. CONCLUSION

Many studies have focused on the increase of attacks on In-vehicle CAN communication. However, due to computational and environmental limitations of ECU units and CAN communication, the existing techniques have many constraints to practical use. By analyzing the existing techniques, we found three requirements for the practical application of In-vehicle CAN communication. In this study, we propose a sender authentication technique. Unlike the existing techniques, our protocol protects In-vehicle CAN communication from attacks without modifying CAN standard. In other words, our technique can be easily implemented under CAN standard. The technique is also efficient in the sense that it does not influence the performance of CAN communication, which means that the use of security tool does not slow down the In-vehicle CAN communication as a result of heavy computation or the use of additional messages. Finally, to prove our assertions, we implemented our protocol in a CANoe simulator and XC2265N and ATmega328 and showed that it works in practice. Our technique is the practical

authentication technique that guarantees the above explained features.

## REFERENCES

- [1] R. N. Charett, "This car runs on code," *IEEE Spectrum*, vol. 46, no. 3, p. 3, Feb. 2009.
- [2] M. Wolf, A. Weimerskirch, and T. Wollinger, "State of the art: Embedding security in vehicles," *EURASIP J. Embedded Syst.*, vol. 2007, no. 1, 2007, Art. no. 074706.
- [3] 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.
- [4] 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.
- [5] D. K. Nilsson, U. E. Larson, and E. Jonsson, "Efficient in-vehicle delayed data authentication based on compound message authentication codes," in *Proc. IEEE 68th Veh. Technol. Conf.*, Sep. 2008.
- [6] 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.
- [7] *CAN Specification, Specification Version 2.0*, Bosch, Gerlingen, Germany, 1991.
- [8] K. H. Johansson, M. Torngrén, and L. Nielsen, "Vehicle applications of controller area network," in *Handbook Networked Embedded Control Systems*. Boston, MA, USA: Birkhäuser, 2005, pp. 741–765.
- [9] *Cyber Security and Resilience of Smart Cars—Good Practices and Recommendations*, Eur. Union Agency Netw. Inf. Secur., Heraklion, Greece, 2016.
- [10] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, May 2010, pp. 447–462.
- [11] C. Miller and C. Valasek, "Adventures in automotive networks and control units," in *Proc. DEF CON*, 2013, pp. 260–264.
- [12] S. Checkoway, D. McCoy, D. Anderson, B. Kantor, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. 19th Conf. USENIX Sec.*, Washington, DC, USA, 2011, pp. 447–462.
- [13] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *Proc. Black Hat*, 2015, pp. 1–91.
- [14] C. Miller and C. Valasek, "Asurvey of remote automotive attack surfaces," in *Proc Black Hat*, 2014, pp. 1–94.
- [15] H. J. Jo, W. S. 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.
- [16] *ELM Electronics*. Accessed: May 10, 2020. [Online]. Available: <https://www.elmelectronics.com>
- [17] *PLX Device*. Accessed: May 10, 2020. [Online]. Available: <https://www.plxdevices.com>
- [18] S. Woo, J. Lee, Y. Song, H. Moon, and D. H. Lee, "Enhanced Android app-repackaging attack on in-vehicle network," *Wireless Commun. Mobile Comput.*, vol. 2019, Feb. 2019, Art. no. 5650245.
- [19] *Crypto++ 5.6.0 Benchmarks*. Accessed: 2009. [Online]. Available: <https://www.cryptopp.com/benchmarks.html>
- [20] *Mirae Technology*. Accessed: 2018. [Online]. Available: <https://www.cryptopp.com/benchmarks.html>
- [21] 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.
- [22] *Vector, Webpage*. Accessed: May 10, 2020. [Online]. Available: <https://www.vector.com/>



**TAEK-YOUNG YOUN** received the B.S., M.S., and Ph.D. degrees from Korea University, in 2003, 2005, and 2009, respectively. He is currently a Senior Researcher with the Electronics and Telecommunications Research Institute. Since 2016, he serves as an Associate Professor with the University of Science and Technology, Daejeon, South Korea. 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 19 years, especially ten years in automotive security. He specializes in consulting, development and standardization 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, security risk analysis, cyber security management system (CSMS), and evaluation methodologies for automotive security.



**SAMUEL WOO** received the Ph.D. degree in information security from Korea University, Seoul, South Korea, in 2016. He was a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. He is currently an Assistant Professor with the Department of Software Science, Dankook University, Jukjeon, South Korea. His research interests include cryptographic protocols in authentication, security and privacy in vehicular networks, and controller area network security.

• • •