EC-SVC: Secure CAN Bus In-Vehicle Communications With Fine-Grained Access Control Based on Edge Computing

Donghyun Yu, *Graduate Student Member, IEEE*, Ruei-Hau Hsu, *Member, IEEE*, Jemin Lee, *Member, IEEE*, and Sungjin Lee

***Abstract*— In-vehicle communications are not designed for message exchange between the vehicles and outside systems originally. Thus, the security design of message protection is insufficient. Moreover, the internal devices do not have enough resources to process the additional security operations. Nonetheless, due to the characteristic of the in-vehicle network in which messages are broadcast, secure message transmission to specific receivers must be ensured. With consideration of the facts aforementioned, this work addresses resource problems by offloading secure operations to high-performance devices, and uses attribute-based access control to ensure the confidentiality of messages from attackers and unauthorized users. In addition, we reconfigure existing access control based cryptography to address new vulnerabilities arising from the use of edge computing and attribute-based access control. Thus, this paper proposes an edge computing-based security protocol with fine-grained attribute-based encryption using a hash function, symmetric-based cryptography, and reconfigured cryptographic scheme. In addition, this work formally proves the reconfigured cryptographic scheme and security protocol, and evaluates the feasibility of the proposed security protocol in various aspects using the CANoe software.**

Manuscript received July 9, 2021; revised November 9, 2021 and December 24, 2021; accepted January 23, 2022. Date of publication February 16, 2022; date of current version April 5, 2022. This work was supported in part by the Institute of Information and Communications Technology Promotion (IITP) Grant by the Korean Government through the Ministry of Science, ICT and Future Planning (MSIP) (Development of attack response and intelligent RSU technology for vehicle security threat prevention) under Grant 2021-0-01277; in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government through the Ministry of Science and ICT (MSIT) under Grant NRF-2018R1A5A1060031; in part by the NRF Grant by the Korean Government through MSIP under Grant 2020R1A2C2008878; in part by the Ministry of Science and Technology of Taiwan under Grant 109-2221-E-110-041-MY3, Grant 109-2923-E-011-006-MY3, and Grant 110-2218-E-110-007-MBK; and in part by the Information Security Research Center, National Sun Yat-sen University, Taiwan. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. George Loukas. *(Corresponding author: Jemin Lee.)*

Donghyun Yu and Sungjin Lee are with the Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu 42988, South Korea (e-mail: xaos4715@dgist.ac.kr; sungjin.lee@dgist.ac.kr).

Ruei-Hau Hsu is with Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan (e-mail: rhhsu@mail.cse.nsysu.edu.tw).

Jemin Lee is with the Department of Electrical and Computer Engineering, Sungkyunkwan University (SKKU), Suwon 16419, South Korea (e-mail: jemin.lee@skku.edu).

Digital Object Identifier 10.1109/TIFS.2022.3152405

***Index Terms*— In-vehicle security, access control, attribute-based encryption, edge computing.**

I. Introduction

With the noticeable improvements in vehicles, internal devices in a vehicle start to share an amount of essential data of the car with each other. In addition, vehicles are allowed to share this data with data consumers outside of in-vehicle networks and to accept the control commands from outsiders to electronic control units (ECUs). The most significant advantage of data sharing is to improve their data processing performance. For instance, ECU, which is the most common machine sharing data in a car, can exchange information in order to make an important decision rapidly [1].

Therefore, it is definitely indispensable for ECU to increase the amount of shared data for its decision performance in the car. However, the in-vehicle networks can be exposed to outside attackers. Hence, more frequent data sharing between ECUs can provide more information to potential attackers, who may eavesdrop or manipulate the data for misbehaved operations, as also shown in the experimental attacks such as [2]–[5]. Despite of those security risks, the typical in-vehicle networks do not provide data confidentiality and message authentication, which are the most basic requirements for secure communications. Thus, secure in-vehicle communications are essential to prevent unauthorized accesses and exposure of messages of in-vehicle networks [2], [6]–[8].

In particular, even the most representative in-vehicle communications protocol, that is, CAN protocol, has already been known not to satisfy the important security requirements [9] that are necessary as the CAN bus is accessible externally. This is because the connectivity of vehicle-to-vehicle or vehicle-to-infrastructure was not necessarily considered since CAN is only intended for in-vehicle communications when it was developed, where external attackers do not exist. However, as the data from CAN bus exchanged among vehicles has been facilitated, the security of CAN becomes more crucial since in-vehicle communications still considerably rely on CAN. Moreover, controller area network (CAN) is widely used in industrial control system including electronic equipments for aviation and navigation, medical devices and equipments, as well as vehicles. All of these systems are close to real-time systems, which can be fatal if security issues occur.

To solve the security issues of in-vehicle networks, especially CAN, many researchers have proposed new security protocols for CAN. The details of previous work are described in the following Sec. I-A.

A. Related Work

Initial works recognize the problem that ECUs have insufficient resources, such as power and computing capability to perform cryptographic protocols for secure in-vehicle communications [10]–[12]. Hence, some works have been proposed for in-vehicle communications security using low-cost cryptographic techniques [13]–[15]. Herrewege *et al.* [13] proposed the backward-compatible broadcast authentication protocol in CAN. Groza *et al.* [14] provided a security protocol based on symmetric primitives using key splitting and message authentication code (MAC) mixing. They utilized nodes with high-computing power for distributing keys and a mixed MAC to achieve inter-group authentication, not one-to-one authentication. Kurachi *et al.* [15] focused on the problem of detecting spoofing messages in CAN bus, and proposed the lightweight authentication method. In the above mentioned papers, they did not ensure the confidentiality of the data so they are vulnerable to eavesdropping attacks.

Some recent works have achieved data confidentiality in in-vehicle communications [16]–[18]. Woo *et al.* [16] provided a security protocol for secure real-time data processing in CAN through data encryption and authentication technology. The same authors also proposed a security architecture to meet the CAN with flexible data rate (CAN-FD) standards [17]. This work achieved confidentiality, authentication, and integrity with the limited computing power of ECUs using cryptographic techniques with relatively low processing time such as symmetric key cryptography and hash functions. Püllen *et al.* [18] proposed the scheme to establish an authentication key between ECUs subscribing to the same message type using an implicit certificate. However, these works did not provide a solution to the security issues that could occur when the high-performance node, responsible for key management, is corrupted by an attacker.

To increase the security level of the in-vehicle communications, a security protocol using public key-based cryptography was provided for CAN-FD and FlexRay in [19]. However, this protocol is extremely hard to be executed while driving due to the limited performance of ECUs. Therefore, it should be executed in authorized garages or manufacturers, which is not good in terms of usability. In addition, all ECUs sharing the same secret key can have the key exposure problem, caused by ECUs corrupted by attackers.

Cui *et al.* [20] proposed a fast KP-ABE algorithm by reducing encryption and decryption costs to provide access control to applications on personal data in connected and autonomous vehicles (CAV). However, the required computation cost for decryption is still considerably high to ECUs, which generally have low computation capability. Furthermore, Cui *et al.*'s KP-ABE scheme does not consider the privacy protection on the policy of accessing messages and the credentials of ECUs (i.e., the policy and credential privacy) although those type of information can be used to infer additional information such as the behaviors of vehicular operations.

B. Motivation and Contribution

Despite of the intensive efforts for securing vehicle communications, most of the prior works have some issues as follows. *First, the excessive computing power is required.* Complicated operations are introduced to satisfy the aforementioned security requirements [10]–[13], [19]. Generally, ECUs are unable to process the complex operations since they have lower computing capabilities than that of a typical computer. *Second, the proposed protocols are not secure enough.* Some studies tried to construct their protocols by considering the real-time constraint of in-vehicle communications. However, they ended up meeting only low-level basic security requirements using symmetric-based cryptographic schemes [10], [12], [16], [17]. Nonetheless, as the amount of data shared between ECUs increases, there can be more opportunities for attackers to analyze those data to obtain the personal information of vehicles such as the location and the mobility pattern for malicious purposes. Furthermore, each ECU shares data with multiple ECUs without access control, an attacker could corrupt one ECU and access confidential vehicle information [2], [21]. To prevent this, the fine-grained access control to the shared data in the vehicles becomes more and more critical.

As a solution to achieve the fine-grained access control, the attribute-based encryption (ABE) has been mainly adopted among various cryptographic schemes [22]. Compared to other cryptographic scheme for access control, ABE is superior in terms of security, privacy, and key and membership management.[1] Furthermore, the computational complexity of ABE is associated with the number of attributes, not the number of receivers. Hence, ABE can be more suitable for the system with large number of receivers such as the in-vehicle system that shares the data among ECUs and also with external networks. Despite it, ABE has high computation cost, which may lead to long computation latency that cannot be acceptable in the in-vehicle system.

Therefore, for secure in-vehicle communications, it is desirable to develop new techniques that give lower computation burden to ECUs, while providing a higher level of security including access control, which is the main objective of this paper. Specifically, to reduce the computation time of ABE, we utilize computing offloading by taking the concept of edge computing.[2] We designate a high-performance device originally inside the vehicle as the edge device to performs certain cryptographic operations for ECUs (considered as resource-limited devices). Finally, we propose a novel secure in-vehicle communication protocol with fine-grained access control based on edge computing, so-called EC-SVC.

[1]Here, ABE is adopted instead of broadcast encryption (BE) [23] because it is 1) more robust against privacy issues as it does not require to specify the recipient's ID, and 2) more flexible in the key and membership management as it does not need to update/re-install the public key of all users whenever a new user joins.

[2]Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network on downstream data on behalf of cloud services and upstream data on behalf of IoT services [24].

The contribution of this paper is as follows.

- We exploit the concept of edge computing by introducing the SA, which handles cryptographic operations on behalf of low computing power ECUs. In addition, we propose the *enhanced attribute-based encryption with hidden policy and credential (EABEHP)*, which achieves attribute-based access control with data confidentiality and policy privacy even against the SA.
- Using the EABEHP, we propose *EC-SVC* that achieves the fine-grained access control and the end-to-end security against insider attacker, together with the policy and credential privacy adding to the security features, achieved in existing works, such as data confidentiality, authentication, and integrity.
- We formally prove the security of the EABEHP and the EC-SVC. We also implement the proposed protocol using CANoe [25], an in-vehicle network simulator and measure the protocol execution time according to several variables such as the number of attributes and the number of ECUs. Through this, we show the feasibility of the proposed protocol under the real-time requirements in in-vehicle communications.
- By focusing on CAN among the in-vehicle network, we have shown that our work is not limited to in-vehicle but is practical and feasible in a wide range of industrial control systems. In addition, we have shown that the proposed EABEHP is suitable as a solution to achieve access control and enhanced privacy in a system consisting of low computing power devices.

II. System and Security Models

In this section, we describe the in-vehicle network model including the attack model and the security requirements. We then introduce the security model and definitions, which are used to analyze the security of the proposed protocol.

A. System Model

We consider the in-vehicle network, where there exist two kinds of entities, ECU and on board unit (OBU) as shown in Fig. 1. All entities connect to CAN, which is a multi-master network, and communicate through the CAN bus. Each ECU collects the data from sensors, and shares it with actuators, other ECUs, and OBU. The ECU has low computing power, so long processing time is generally required to execute the advanced cryptography that provides a high-level of security. On the other hand, the OBU, which can communicate with nodes inside and outside of the vehicle, has higher computing power compared to the ECU. We utilize the OBU as an edge computing node, which allows performing cryptographic operations, offloaded by ECU. We call this type of OBU as the security agent (SA) [26].[3] In this system model, we also consider security issues that may occur during offloading the cryptographic works from the ECU to the SA, and describe the details of the key management model for attribute-based key exchange in Sec. IV-A.

[3]Note that the role of edge computing node can also be assigned to other/additional node with high computing performance, connected to CAN.

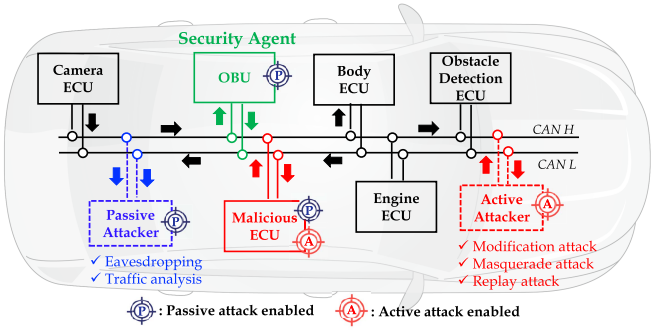

Fig. 1. Overview of the system model.

B. Attack Model and Security Requirements

In the proposed system model, the ECUs can exchange the messages by using public/privacy keys, issued by the administrator of the system. Each ECU has the assigned attributes so that the other ECUs can send the messages to the intended receiver ECUs based on the specified policy. The OBU is considered as an SA, which will perform the operations for offloaded computation of ABE as mentioned in the proposed scheme/protocol. The OBU can authenticate ECUs by the pre-shared secret key to check the legitimacy of ECUs. However, the OBU cannot observe the communication patterns and content of exchanging messages among ECUs in the in-vehicle network.

As shown in Fig. 1, there can be two types of attackers in in-vehicle network, the passive attacker and the active attacker. The passive attackers can be connected to the CAN bus and eavesdrop the messages in CAN. The active attacker can cause severe problems to the vehicle by modifying the data sent by the ECU and the OBU or by sending the wrong key.

Specifically, OBU is considered not only as an edge computing node but also as a gateway between the in-vehicle network and the external network of a vehicle. Thus, the corruption of OBU needs to be addressed due to the accessibility of the OBU to outsider attackers. After corrupting an OBU, the attacker will exploit the corrupted OBU to learn knowledge of the communication patterns and the content of messages against in-vehicle networks. After that, attackers will be able to subsequently launch attacks by sending accurate command messages to the in-vehicle network. Thus, the attacker model of the proposed scheme regards OBU as an honest-but-curious adversary, who will try to expose all the secret keys and the content of messages to maximize the advantages of the corruption. To capture the abilities of passive attacker and active attacker, we propose the following security requirements for in-vehicle networks.

- *Confidentiality*: All messages are broadcasted due to the characteristics of the in-vehicle communications, so an attacker can easily eavesdrop the messages. Hence, the confidentiality should be guaranteed, only the intended recipients can obtain the plaintext of the message.
- *Authentication*: An attacker may retransmit the message, which exchanged between entities in the previous session, to impersonate a legitimate device. Therefore, we need to

achieve mutual authentication to ensure that ECUs and SA can authenticate each other for the legitimacy.

- *Fine-grained Access Control*: Most of the transmitted data in a vehicle are not for all ECUs, but for certain ECUs. In addition, when ECUs or OBU is corrupted by an attacker or some malicious nodes are connected in CAN, the transmitted message in CAN can be exposed without access control [27], [28]. Therefore, the fine-grained access control is required in in-vehicle networks.
- *Policy and Credential Privacy*: A sender encrypts messages using the given policies to indicate the intended receivers according to their attributes for access control of message exchange. The encryption and decryption procedures should not expose identity information of ECUs to avoid potential analysis or attacks to disrupt normal communications [27].

C. Security Definitions of Enhanced Attribute-Based Encryption With Hidden Policy and Credential

To prove the security of the proposed attribute-based encryption scheme in Sec. III, so-called enhanced attribute-based encryption with hidden policy and credential (EABEHP), which achieves partial proxy decryption, hidden credential, and hidden policy against honest-but-curious decryption proxy, this section defines the security properties of EABEHP. Since EABEHP is a reconfigured scheme, established based on PEAPOD [27] with achieving additional security properties, we adopt the security definitions of PEAPOD after modifying them slightly to fit in EABEHP.

Definition 1 (C-IND-CPA-RUCA): EABEHP is said to be ciphertext indistinguishability against chosen plaintext attack and restricted user coalition attack (C-IND-CPA-RUCA) if any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ has only negligible advantage to distinguish the ciphertext of two given messages in the following security game.[4]

1) **Setup Phase:** The challenger $\mathcal{C}$ sets up the EABEHP scheme and provides the attacker with all the public parameters of the system.
2) **Training Phase 1:** $\mathcal{A}$ can only corrupt either the proxy (edge device) or any user except for the target user. That is, (i) the attacker can corrupt the proxy to learn its secrets and act on its behalf or (ii) $\mathcal{A}$ has the following abilities.
   - $\mathcal{A}$ can register a new user or corrupt an honest user, who do not satisfy a policy P^* on the system, thereby $\mathcal{A}$ can learn their secrets and act on their behalves.
   - $\mathcal{A}$ can make requests of **TransformCipherText**, **Extract**, and **ProxyDecrypt1** to the proxy.
   - $\mathcal{A}$ can make requests of **TimeKeyGen**, **TransformUserKey**, and **Shuffle** to honest users.
3) **Challenge Phase:** $\mathcal{A}$ outputs two messages M_0 and M_1 of equal length, and a policy P^* under the following restriction.
   Restriction 1: None of the coalitions of corrupted users collectively satisfies the P^*.
   $\mathcal{C}$ then flips the random coin $b \overset{R}{\in} \{0, 1\}$ and generates C^* by encrypting the message M_b under the policy P^* by the **Encrypt** according to b. $\mathcal{C}$ returns C^* to $\mathcal{A}$.
4) **Training Phase 2:** $\mathcal{A}$ can perform the operations defined in **Training Phase 1**, except that none of the corrupted users can satisfy the policy P^*.
5) **Guessing Phase:** $\mathcal{A}$ outputs a guessing $b' \overset{R}{\in} \{0, 1\}$. $\mathcal{A}$ wins the game if $b' = b$.

Definition 2 (P-IND-CPA-RUCA): EABEHP is said to be policy indistinguishability against chosen plaintext attack and restricted user coalition attack (P-IND-CPA-RUCA) if all PPT adversaries only have negligible advantage to distinguish the ciphertext of two given policies in the following security game.

1) **Setup Phase:** Same as that in Definition 1.
2) **Training Phase 1:** Same as that in Definition 1.
3) **Challenge Phase:** $\mathcal{A}$ sends a chosen message M^* and two chosen policies, P_0 and P_1, for the encryption of M^* under the following restriction.
   Restriction 2: All the corrupted users satisfy none of the policies, P_0 and P_1, or they all satisfy both policies. $\mathcal{C}$ selects a random bit $b \in \{0, 1\}$ and encrypts M^* with the given P_b to generate C^* according to b.
4) **Training Phase 2:** Same as that in **Training Phase 1**.
5) **Guessing Phase:** Same as that in Definition 1.

Definition 3 (Credential Privacy): EABEHP is said to support credential privacy if all PPT adversaries only have negligible advantage to distinguish the real credential of the specified user from the credentials of the other users with non-negligible advantages as the following game.

1) **Setup Phase:** Same as that in Definition 1.
2) **Training Phase 1:** Same as that in Definition 1.
3) **Challenge Phase:** $\mathcal{A}$ outputs the credentials of two users, a selected policy P^*, and a selected message M. $\mathcal{C}$ then outputs a ciphertext of M with P^* and SK_{U_b} according to $b \overset{R}{\in} \{0, 1\}$, where the associated attributes of SK_{U_1} and SK_{U_2} either both satisfy P^* or do not satisfy P^*.
4) **Training Phase 2:** Same as that in **Training Phase 1**, except that the corruption of the users possessing SK_{U_1} or SK_{U_2} is not allowed.
5) **Guessing Phase:** Same as that in Definition 1.

D. Security Definitions of Secure In-Vehicle Communications With Access Control

We capture the capabilities of attackers by following definitions in the system model. We first explain the notation used in the security model. The proposed protocol is called Γ, and we regard the communication between two users A and B in communication sessions at t_1 and t_2 as $\Gamma^{t_1}_{A,B}$ and $\Gamma^{t_2}_{B,A}$, respectively. We describe oracles, propose attackers who can

[4]The adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ are used to formally prove the security of the proposed cryptographic scheme or security protocol. The security game of the proposed EABEHP includes policy P, because the EABEHP is a kind of attribute-based encryption, and the encrypter encrypts a message by inputting a public key and a policy that is the condition of the receivers who can see the message.

query these oracles, and define the security of the proposed protocols according to security requirements.

The oracles used to capture the attacker's capabilities are as follows.

- **Execute**($\Gamma_{A,B}^{t_1}$, $\Gamma_{B,A}^{t_2}$): This oracle models all kinds of passive attackers that can eavesdrop all data between $\Gamma_{A,B}^{t_1}$ and $\Gamma_{B,A}^{t_2}$.
- **Send**($\Gamma_{A,B}^{t_1}$, M): This oracle models an active attacker that sends a message M to $\Gamma_{A,B}^{t_1}$.
- **Expose**($\Gamma_{A,B}^{t_1}$): This oracle models the exposure of the session key of A, shared with B, at communication session t_1.
- **Corrupt**($\Gamma_{A,B}^{t_1}$): This oracle models the exposure of the long-term secret key of A, shared with B, at communication session t_1.
- **CorruptAK**(S_i): This oracle models an attribute-based private key SK_i exposure corresponding to attribute set S_i.
- **Test**($\Gamma_{A,B}^{t_1}$): This oracle models the test of session key security. When one queries this oracle and both parties, which are the partners of each other, in the protocol are accepted, it will return a real session key or a random string depending on a random bit. Otherwise, it returns an invalid output.
- **TestPolicy**(P_0, P_1, M) This oracle models to test the privacy of the given policies for encryption. When one queries this oracle with the inputs of two given policies, P_0 and P_1, and a message M, it will output an encryption on M with P_b according to the randomly selected $b \in \{0, 1\}$.
- **TestCert**($\Gamma_{A,B}^{t_1}$, P^*): This oracle models to test the privacy of user's credential. When one queries this oracle with the input of $\Gamma_{A,B}^{t_1}$ and the target policy P^*, it will output either the credential of $\Gamma_{A,B}^{t_1}$ or a randomly selected credential with the restriction that the attributes of both credentials satisfy P^* or do not satisfy P^*.

We also define the security properties of the proposed EC-SVC according to the security requirements discussed in Sec. II-A. Note that Definitions 4 and 5 are adopted from "entity authentication and key distribution" in [29] and "attribute-based authentication key exchange" in [30], respectively, while Definitions 6 and 7 have been newly provided in this work to meet the security requirements.

Definition 4 (Mutual Authentication): We assume that S simulates $\Gamma_{A,B}^{t_1}$ and $\Gamma_{B,A}^{t_2}$, and interacts with $\mathcal{A}$, who can query polynomial number of **Execute** and **Send** oracles. After $\mathcal{A}$ queries these oracles, it sends a message to be accepted by $\Gamma_{A,B}^{t_1}$ or $\Gamma_{B,A}^{t_2}$, where $\Gamma_{A,B}^{t_1}$ or $\Gamma_{B,A}^{t_2}$ has not accepted each other. $\mathcal{A}$ has the following advantage

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{MuAuth}} = \Pr[\mathcal{A} \text{ accepted by } \Gamma_{A,B}^{t_1} \text{ or } \Gamma_{B,A}^{t_2}]. \quad (1)$$

The mutual authentication between A and B is guaranteed if $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{MuAuth}}$ is negligible.

Definition 5 (Attribute-Based Key Exchange): An adversary $\mathcal{A}$ is allowed to make **Execute**, **Send**, and **CorruptAK** oracles in polynomial time in Phase 1. After Phase 1, $\mathcal{A}$ can initiate an instance $\Gamma_{A,B}^{t_1}$ with the selected policy of which cannot be satisfied by the attributes of the secret keys, obtained from **CorruptAK**. Then, the attacker queries **Test** to obtain a session key K or a random string according to a random bit $b \in \{0, 1\}$. The $\mathcal{A}$ can continue asking queries in Phase 2. After finishing Phase 2, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b = b'$, $\mathcal{A}$ wins. The advantage of $\mathcal{A}$ to break this security game is defined as

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{AKE}} = \Pr[\mathrm{Succ}_{\mathcal{A}}^{\mathrm{AKE}}] - 1/2, \quad (2)$$

where $\mathrm{Succ}_{\mathcal{A}}^{\mathrm{AKE}}$ is the event that $\mathcal{A}$ outputs a guess $b' = b$. If $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{AKE}}$ is negligible, the attribute-based key exchange security is achieved.

Definition 6 (Policy Privacy): S simulates $\Gamma_{A,B}^{t_1}$ and $\Gamma_{B,A}^{t_2}$, and interacts with $\mathcal{A}$, who can query polynomial number of **Execute** and **Send** oracles in polynomial time. After this phase, $\mathcal{A}$ queries **TestPolicy** with message M and two valid policies, P_0 and P_1, as the input to obtain a C_0 or C_1 according to a random bit $b \in \{0, 1\}$, where C_0 and C_1 are encryption for M with P_0 and P_1, respectively. $\mathcal{A}$ has the following advantages

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{PP}} = \Pr[\mathrm{Succ}_{\mathcal{A}}^{\mathrm{PP}}] - 1/2, \quad (3)$$

where $\mathrm{Succ}_{\mathcal{A}}^{\mathrm{PP}}$ is the event that $\mathcal{A}$ outputs a guess $b' = b$. If $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{PP}}$ is negligible, the policy privacy is achieved.

Definition 7 (Credential Privacy): S simulates $\Gamma_{A,B}^{t_1}$ and $\Gamma_{B,A}^{t_2}$, and interacts with $\mathcal{A}$, who can query polynomial number of **Execute** and **Send** at polynomial time. After this phase, $\mathcal{A}$ queries **TestCert** with target policy P^* as the input to obtain a credential of $\Gamma_{A,B}^{t_1}$ or a randomly selected credential according to a random bit $b \in \{0, 1\}$, where the attributes of both credentials satisfy P^* or do not satisfy P^*. $\mathcal{A}$ has the following advantages

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{CP}} = \Pr[\mathrm{Succ}_{\mathcal{A}}^{\mathrm{CP}}] - 1/2, \quad (4)$$

where $\mathrm{Succ}_{\mathcal{A}}^{\mathrm{CP}}$ is the event that $\mathcal{A}$ outputs a guess $b' = b$. If $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{CP}}$ is negligible, the credential privacy is achieved.

III. System Preliminaries

This section introduces the required background for the proposed protocol including the cryptographic algorithms.

A. Pseudorandom Function and Permutations

We describe the hash function, e.g., SHA-256, and the symmetric encryption, e.g., AES128, used in this protocol as pseudorandom function [31] and pseudorandom permutation [32], respectively. First, the pseudorandom function defined over (K, X, Y) is an efficient and deterministic function, which returns a pseudorandom output sequence

$$H : K_H \times X \rightarrow Y, \quad (5)$$

where K_H is the key space, $X \subseteq \{0, 1\}^{l_1}$ is the input space, $Y \subseteq \{0, 1\}^{l_2}$, and $l_1 > l_2$. The pseudorandom permutation defined over (K, X) is an efficient and deterministic function which returns a pseudorandom output sequence

$$E : K_E \times X \rightarrow X', \quad (6)$$

where K_E is the key space, $X \subseteq \{0,1\}^l$ is the input space, and $X' \subseteq \{0,1\}^l$. There is an efficient inversion algorithm $D(K_E, X')$ for this pseudorandom permutation.

We additionally describe the shuffle function, which is a kind of pseudorandom permutation to be used in the EABEHP scheme. The shuffle function has the same input and output space, which returns a pseudorandom output sequence

$$SH : K_{SH} \times X \rightarrow X, \tag{7}$$

where K_{SH} is the key space, and $X \subseteq \{0,1\}^{l_N}$ is the input and output space. There is an efficient inversion algorithm $SH^{-1}(K_{SH}, X)$ for this shuffle function.

B. Enhanced Attribute-Based Encryption With Hidden Policy and Credential

1) Intuition: The proposed EABEHP is an enhanced scheme of the privacy-enhanced attribute-based publishing of data (PEAPOD) scheme in [27]. Hence, they have similar algorithms such as **TransformCipherText** and **Extract** algorithms of EABEHP, which are respectively corresponding to Deposit and Retrieval algorithms in PEAPOD. On the other hand, EABEHP has additional algorithms for offloading the decryption operations to the honest-but-curious SA. Specifically, **TimeKeyGen**, **TransformUserKey**, **Shuffle**, and **ProxyDecrypt** are newly introduced for 1) confidentiality against SAs for proxy decryption and 2) hidden attributes and policies against SA. Among them, **ProxyDecrypt** is also required for the SA to perform offloaded decryption operation.

Regarding the hidden attribute and policies against SA, how to achieve them can be briefly explained as follows. First of all, a message sender generates k tuples for the required attributes in a policy, which can be combined as a secret key. It also generates α tuples randomly for the unrequired attributes and the other tuples for irrelevant attributes, which are set to one. In order to hide the encryption policy from the proxy and outsiders, each of those tuples are encrypted using ElGamal encryption. Therefore, the proxy and outsiders will not learn any policy information as the confidentiality of the each tuples is guaranteed. In addition, only the receiver with attributes that satisfy the given policy can decrypt the ciphertext as the secret key used to encrypt the message can be obtained by combining k tuples, which realizes the verification of attribute-policy matching.

Therefore, compared to PEAPOD [27], the proposed EABEHP not only supports the decryption offloading that significantly reduces the computational overhead on the user side but also provides the enhancement to privacy protection by achieving policy and credential hiding.

2) Construction: The proposed scheme consists of ten algorithms.

- **Setup**(1^λ): This algorithm chooses the cyclic group $\mathbb{G}$ of prime order p with a generator g. Next, it chooses a large prime number q such that $q|(p-1)$ and random numbers $\{a_i\}_{i\in I}$ for all attributes of the system, where $I = \{1, 2, 3, \ldots, N\}$ is the universal set of attribute indices of the system, and N is the number of system attributes. After that, the algorithm generates the master public key, MPK, of the system as

$$MPK = \{g,\ p,\ q,\ \{PK_{S_i} = g^{a_i}\}_{i\in I}\},$$

where $a_i \in \mathbb{Z}_q^*$. Then, it randomly chooses a master secret key $MSK = K_S$ and generates a transformation secret key $TK = \{TK_{S_i} = s_i\}_{i\in I}$ where $a_i + s_i = K_S$ for all $i \in I$. Finally, it generates a group key K_{group} for all users in the system. The algorithm then outputs MSK, MPK, TK, and K_{group}.
- **KeyGen**(MSK, ID_j, I_j): This algorithm generates user attribute keys, $SK_{U_j} = \{SK_{U_{j,i}} = a_{j,i}\}_{i\in I_j}$, where $I_j \subseteq I$ is the set of attributes indices of user j. Next, it generates a re-encryption key $RK_{U_j} = \sum_{i\in I_j} s_{j,i}$ for all attribute indices of the user j where $a_{j,i} + s_{j,i} = K_S$. This algorithm outputs SK_{U_j} and RK_{U_j} for user j.
- **TimeKeyGen**(K_{group}, t_k): This algorithms takes K_{group} and the time slot t_k as inputs, and generates $\omega_k = H(r_k||K_{\text{group}})$ as the output, where r_k is randomly selected and distinct for different t_k.
- **TransformUserKey**(ω_k, SK_{U_j}): This algorithm takes ω_k and SK_{U_j} as inputs, and outputs $AK_{U_j} = \sum_{i\in I_j} a_{j,i} + \omega_k$ as the transformed user attribute key.
- **Encrypt**(MPK, T, ω_k, M): This algorithm first takes MPK, T, ω_k, and M as inputs, and outputs C as the ciphertext of M. Here, $T = \{t_i\}_{i\in I}$ is a policy set, where $t_i = 1$ if the attribute i is required, $t_i = 0$ if the attribute i is irrelevant, and $t_i = -1$ if the attribute i is unrequired, and $M \in \mathbb{Z}_q$ is the message to be encrypted. Then, it generates the message tuples depending on each $i \in I$ as

$$p_i = \begin{cases} k_i & \text{if } t_i = 1 \\ 1 & \text{if } t_i = 0 \\ \alpha_i & \text{if } t_i = -1 \end{cases},$$

$$\text{such that} \quad \prod_{\substack{t_i\in T \wedge t_i=1, \\ \forall i\in I}} p_i \equiv M \ (mod\ q),$$

for randomly selected $\alpha_i \in \mathbb{Z}_q$. Next, it randomly selects $r_j \in \mathbb{Z}_q$ and encrypts each message tuple, p_i, with $PK_{S_i} = g^{a_i}$ as

$$C = \{A, \langle B_i\rangle_{i\in I}, D\} = \{g^{r_j}, \langle p_i (g^{a_i})^{r_j}\rangle_{i\in I}, (g^{r_j})^{\omega_k}\},$$

where $\langle\cdot\rangle$ means a sequence.
- **Shuffle**(C, ω_k): This algorithm permutes the order of tuples by a pseudorandom permutation as

$$\begin{aligned} sC &= \{A, \langle \widehat{B}_i\rangle_{i\in I}, D\} \\ &= \{g^{r_j}, \langle \widehat{B}_i = p_{i'}(g^{a_{i'}})^{r_j}\rangle_{i\in I\wedge i'=SH_{\omega_k}(i)}, (g^{r_j})^{\omega_k}\}, \end{aligned}$$

where $i' = SH_{\omega_k}(i)$, is a shuffle function, which takes ω_k and $i \in I$ as inputs and outputs $i' \in I$.
- **TransformCipherText**(sC, TK): This algorithm use TK to transform sC as

$$\begin{aligned} sC' &= \{A', \langle B'_i\rangle_{i\in I}, D'\} = \{A, \langle \widehat{B}_i A^{s_i}\rangle_{i\in I}, D\} \\ &= \{g^{r_j}, \langle p_{i'} g^{a_{i'} r_j + s_i r_j}\rangle_{i\in I\wedge i'=SH_{\omega_k}(i)}, (g^{r_j})^{\omega_k}\}. \end{aligned}$$

- **Extract**$(sC', \hat{I}_r)$: This algorithm takes sC' and $\hat{I}_r$ as inputs and outputs the extracted ciphertext sC'_r. Here, $\hat{I}_r$ is a set of elements obtained by inversely permuting each element $i \in I_r$ such as $SH^{-1}_{\omega_k}(i)$. It extracts sC'_r from sC' according to $\hat{I}_r$ as

$$\begin{aligned} sC'_r &= \{A'_r, B'_r, D'_r\} = \{A', \prod_{i\in\hat{I}_r} B'_i, D'\} \\ &= \{A, \prod_{i\in\hat{I}_r} \widehat{B}_i A^{s_i}, D\} \\ &= \{g^{r_j}, \prod_{\substack{i\in\hat{I}_r \wedge \\ i'=SH_{\omega_k}(i)}} p_{i'} g^{a_{i'} r_j + s_i r_j}, (g^{r_j})^{\omega_k}\}. \end{aligned}$$

- **ProxyDecrypt1**$(sC'_r, AK_{U_r}, RK_{U_r}, TK)$: This algorithm takes sC'_r, AK_{U_r}, RK_{U_r}, and TK as inputs, and outputs the partial decrypted ciphertext sC''_r and decryption materials AM_{r_j} as

$$\begin{aligned} sC''_r &= D'_r \cdot B'_r / (A'_r)^{AK_{U_r} + RK_{U_r}} \\ &= \left(\prod_{\substack{i\in\hat{I}_r \wedge \\ i'=SH_{\omega_k}(i)}} p_{i'} \right) (g^{r_j})^{\sum_{i\in\hat{I}_r \wedge i'=SH_{\omega_k}(i)} s_i - s_{i'}}, \\ AM_{r_j} &= \{(A'_r)^{s_i}\}_{i\in I} = \{g^{r_j s_i}\}_{i\in I}, \end{aligned}$$

  where N_r is the number of attributes of user r.
- **ProxyDecrypt2**(sC''_r, AM_{r_j}, I_r): This algorithm takes sC''_r, AM_{r_j}, and I_r as inputs, and outputs M as

$$M = sC''_r \left\{ \prod_{i\in I_r} g^{r_j s_i} \Big/ \prod_{i\in\hat{I}_r} g^{r_j s_i} \right\}.$$

IV. Proposed Secure In-Vehicle Communications With Fine-Grained Access Control Based on Edge Computing (EC-SVC)

This section present, the proposed EC-SVC protocol including the key management and the in-vehicle security protocol. In the key management, we present the method to install the symmetric keys as well as the keys in the EABEHP scheme on each device. In addition, we present the in-vehicle security protocol that satisfies the security requirements presented in Sec. II-A. This protocol involves a sender-ECU, receiver-ECUs, and SA, and includes the authentication process between each device, and the EABEHP scheme. We first describe the key management and then the in-vehicle security protocol.

A. Key Management

Figure 2 shows the key management of EC-SVC. The trust authority (TA) issues the required cryptographic keys for all entities in the system by the following procedures: (1) TA generates MSK, MPK, TK, K_{group}, and $\{K_{\text{SA,ECU}_j}\}_{j\in J}$ by **EABEHP.Setup**. It then generates SK_{U_j} and RK_{U_j} for each ECU_j by **EABEHP.KeyGen**. (2) TA publishes MPK and keeps MSK secretly. It then distributes $\{SK_{U_j}, K_{\text{SA,ECU}_j}\}_{j\in J}$ to each user j, and RK_{U_j}, TK and all $\{K_{\text{SA,ECU}_j}\}_{j\in J}$ to the SA. (3) The TA sends the group key K_{group} securely to each ECU by executing the group key distribution mechanism [33]. Assume that the size of the security keys is sufficient to provide the security of the system for the life-time of a vehicle.

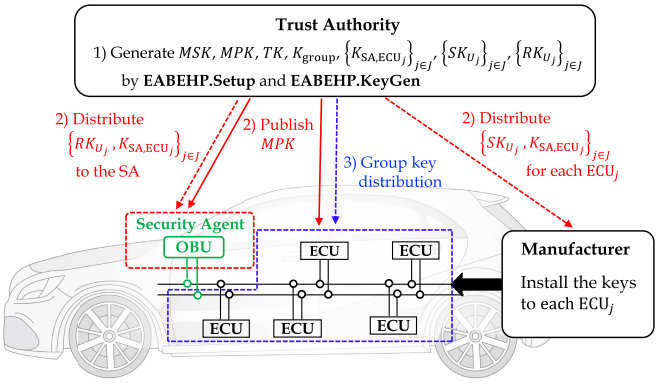

Fig. 2. Long-term secret key management.

B. In-Vehicle Security Protocol

This subsection describes the in-vehicle security protocol, which is the edge computing-based in-vehicle authenticated key exchange protocol with attribute-based access control in Fig. 3. In the proposed system, after such key exchange, the data sharing protocol is performed using the exchanged key. Since the data sharing protocol consists of a simple procedure that uses a hash function and symmetric encryption using an exchanged key for confidentiality and integrity, we omit the description of data sharing protocol. The attribute based key exchange protocol is executed when a vehicle is started and consists of the following twelve steps.

1) The sender-ECU(ECU_S) generates nonce $N_1 \in \{0,1\}^L$ and $\sigma_1 = \text{H}_{K_{\text{SA,ECU}_\text{S}}}(\text{ID}_\text{S}||N_1)$. The ECU_S then send $\{\text{ID}_\text{S}||\sigma_1||N_1\}$ to the SA.
2) The SA verifies σ_1 and generates nonce $N_2 \in \{0,1\}^L$. The SA then generates $\sigma_2 = \text{H}_{K_{\text{SA,ECU}_\text{S}}}(\text{ID}_\text{S}||N_1+1||N_2)$ and sends $\{\sigma_2||N_2\}$ to the ECU_S.
3) The ECU_S verifies σ_2. If passed, the ECU_S executes $\omega_k =$ **EABEHP.TimeKeyGen**(K_{group}, r_k). The ECU_S generates data sharing key $K \in \mathbb{Z}_q$ and encrypts K as $K' = \text{E}_{K_{\text{group}}}(K)$. It then computes $C =$ **EABEHP.Encrypt**(MPK, T, ω_k, K'). After that, the ECU_S computes $sC =$ **EABEHP.Shuffle**(C, ω_k), and generates $CK_\text{S} = \text{H}_{K_{\text{SA,ECU}_\text{S}}}(\text{ID}_\text{S}||N_1+1||N_2+1)$ and $\sigma_3 = \text{H}_{CK_\text{S}}(sC)$. Subsequently, the ECU_S send $\{sC||\sigma_3\}$ to the SA.
4) The SA generates $CK_\text{S} = \text{H}_{K_{\text{SA,ECU}_\text{S}}}(\text{ID}_\text{S}||N_1+1||N_2+1)$ and verifies σ_3. It then computes $sC' =$ **EABEHP.TransformCipherText**(sC, TK). After that, the SA waits for the request message from the $\text{ECU}_{\text{R}j}$.
5) When the SA needs to retrieve the data, sent by $\text{ECU}_{\text{R}j}$, it computes $\omega_k =$ **EABEHP.TimeKeyGen**(K_{group}, r_k) and $A_{\text{R}j} =$ **EABEHP.TransformUserKey**$(\omega_k, SK_{\text{U}_{\text{R}j}})$. The $\text{ECU}_{\text{R}j}$ then generates nonce $N_{3j} \in \{0,1\}^L$

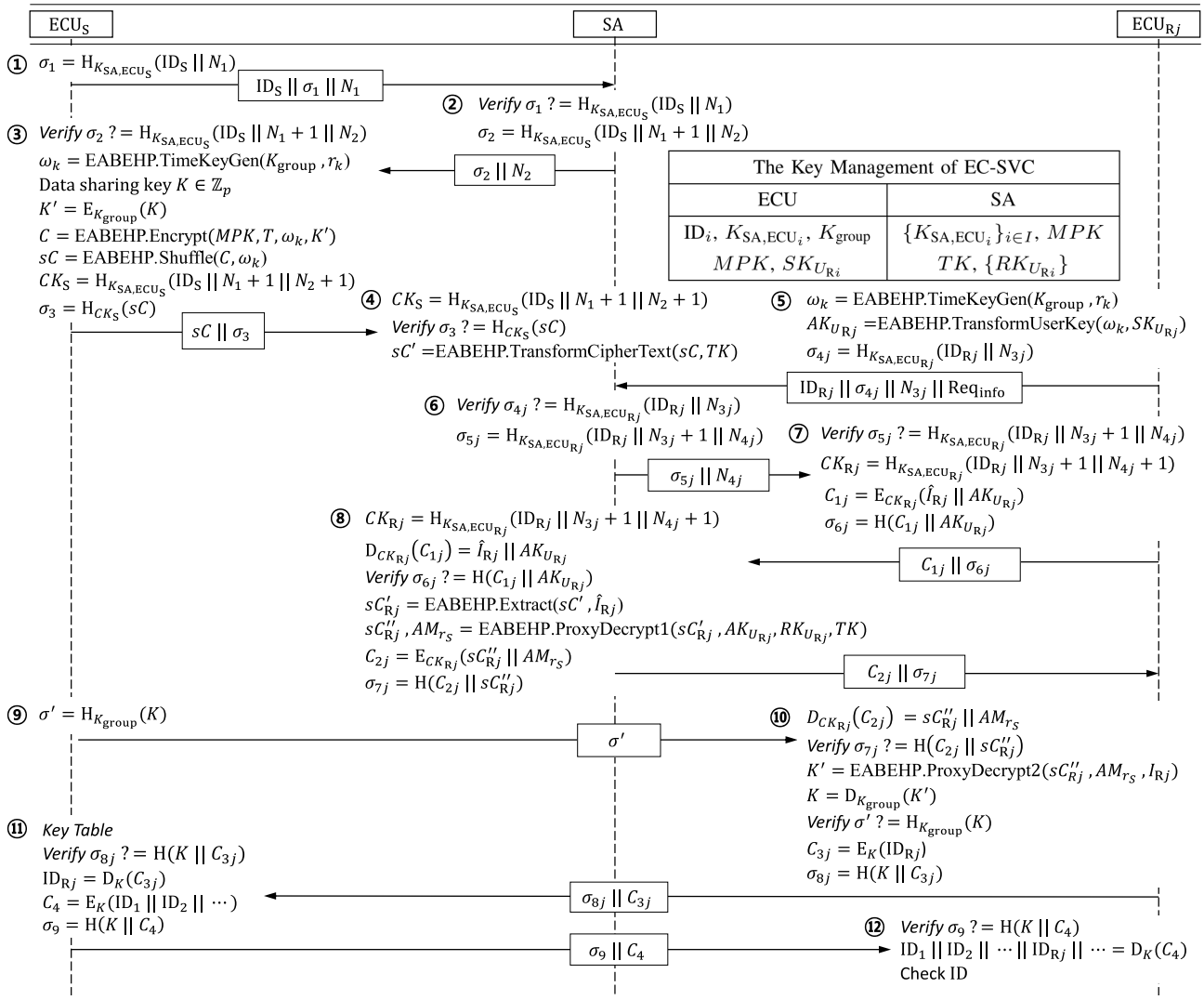

Fig. 3. Edge computing-based in-vehicle authenticated key exchange protocol with attribute-based access control.

and $\sigma_{4j} = \mathrm{H}_{K_{\mathrm{SA,ECU_{Rj}}}}(\mathrm{ID_{Rj}}||N_{3j})$. Subsequently, the $\mathrm{ECU_{Rj}}$ sends $\{ID_{\mathrm{Rj}}||\sigma_{4j}||N_{3j}||\mathrm{Req_{info}}\}$ to the SA, where $\mathrm{Req_{info}}$ is the message of requesting a process to obtain a data sharing key K.

6) The SA verifies σ_{4j} first. If passed, the SA generates nonce $N_{4j} \in \{0,1\}^L$ and $\sigma_{5j} = \mathrm{H}_{K_{\mathrm{SA,ECU_{Rj}}}}(\mathrm{ID_{Rj}}||N_{3j}+1||N_{4j})$. The SA then sends $\{\sigma_{5j}||N_{4j}\}$ to the $\mathrm{ECU_{Rj}}$.
7) The $\mathrm{ECU_{Rj}}$ verifies σ_{5j} and generates $CK_{\mathrm{Rj}} = \mathrm{H}_{K_{\mathrm{SA,ECU_{Rj}}}}(\mathrm{ID_{Rj}}||N_{3j}+1||N_{4j}+1)$. The $\mathrm{ECU_{Rj}}$ encrypts $\{\hat{I}_{\mathrm{Rj}}||AK_{U_{\mathrm{Rj}}}\}$ with the generated CK_{Rj} as $C_{1j} = \mathrm{E}_{CK_{\mathrm{Rj}}}(\hat{I}_{\mathrm{Rj}}||AK_{U_{\mathrm{Rj}}})$ and generates $\sigma_{6j} = \mathrm{H}(C_{1j}||AK_{U_{\mathrm{Rj}}})$. Then, the $\mathrm{ECU_{Rj}}$ sends $\{C_{1j}||\sigma_{6j}\}$ to the SA.
8) After the SA receives $\{C_{1j}||\sigma_{6j}\}$, the SA generates $CK_{\mathrm{Rj}} = \mathrm{H}_{K_{\mathrm{SA,ECU_{Rj}}}}(\mathrm{ID_{Rj}}||N_{3j}+1||N_{4j}+1)$ and then decrypts C_{1j} by CK_{Rj} to obtain the $\{\hat{I}_{\mathrm{Rj}}||AK_{U_{\mathrm{Rj}}}\}$. The SA verifies σ_{6j}. The SA then computes $sC'_{\mathrm{Rj}} = \mathbf{EABEHP.Extract}(sC', \hat{I}_{\mathrm{Rj}})$ and $(sC''_{\mathrm{Rj}}, AM_{r_{\mathrm{S}}}) = \mathbf{EABEHP.ProxyDecrypt1}(sC'_{\mathrm{Rj}}, AK_{U_{\mathrm{Rj}}}, RK_{U_{\mathrm{Rj}}}, TK)$, where r_{S} is a random number generated by $\mathrm{ECU_S}$ during the **EABEHP.Encrypt** process. The SA then encrypts $\{sC''_{\mathrm{Rj}}||AM_{r_{\mathrm{S}}}\}$ with CK_{Rj} to generate C_{2j}, and generates $\sigma_{7j} = \mathrm{H}(C_{2j}||sC''_{\mathrm{Rj}})$. It then sends $\{C_{2j}||\sigma_{7j}\}$ to the $\mathrm{ECU_{Rj}}$.
9) The $\mathrm{ECU_S}$ sends a $\sigma' = \mathrm{H}_{K_{\mathrm{group}}}(K)$ that allows $\mathrm{ECU_R}$ to verify that it has received the correct sharing key K.
10) The $\mathrm{ECU_{Rj}}$ decrypts C_{2j} to obtain $\{sC''_{\mathrm{Rj}}||AM_{r_{\mathrm{S}}}\}$ and verifies $\sigma_{7j} = \mathrm{H}(C_{2j}||sC''_{\mathrm{Rj}})$. The $\mathrm{ECU_{Rj}}$ then computes $K' = \mathbf{EABEHP.ProxyDecrypt2}(sC''_{\mathrm{Rj}}, AM_{r_{\mathrm{S}}}, I_{\mathrm{Rj}})$, and decrypts it with K_{group} to obtain K. Afterwards, the $\mathrm{ECU_{Rj}}$ verifies signature σ'. Finally, the $\mathrm{ECU_{Rj}}$ generates $C_{3j} = \mathrm{E}_K(\mathrm{ID_{Rj}})$ and $\sigma_{8j} = \mathrm{H}(K||C_{3j})$ and sends $\{\sigma_{8j}||C_{3j}\}$ to the $\mathrm{ECU_S}$.
11) The $\mathrm{ECU_S}$ stores the ID of the $\mathrm{ECU_{Rj}}$ that exchanged the data sharing key in the key table. The $\mathrm{ECU_S}$ verifies σ_{8j} and decrypts C_{3j} by K to obtain $\mathrm{ID_{Rj}}$. The $\mathrm{ECU_S}$ then generates $C_4 = \mathrm{E}_K(\mathrm{ID_1}||\mathrm{ID_2}||\cdots)$ for the encryption of all the $\mathrm{ECU_{Rj}}$'s identities and $\sigma_9 = \mathrm{H}(K||C_4)$. Subsequently, the $\mathrm{ECU_S}$ sends $\{\sigma_9||C_4\}$ to the $\mathrm{ECU_{Rj}}$.

12) The $\mathrm{ECU}_{\mathrm{R}j}$ verifies signature σ_9 and decrypts C_4. If the $\mathrm{ECU}_{\mathrm{R}j}$ can find its own identity contained, it can authenticate the $\mathrm{ECU}_{\mathrm{S}}$ successfully, which verified the mutual authentication.

Each step of the proposed protocol can be summarized as follows: 1), 2), and 3) are mutual authentication processes between $\mathrm{ECU}_{\mathrm{S}}$ and SA; 3) also includes the process that $\mathrm{ECU}_{\mathrm{S}}$ encrypts and shuffles attribute-based keys of the transmitted data, and send it to SA; 4) is a process that SA transforms the encrypted attribute-based key; 5), 6), and 7) are mutual authentication processes between SA and $\mathrm{ECU}_{\mathrm{R}j}$; 8) is a process that SA receives the $\mathrm{ECU}_{\mathrm{R}j}$ key and partially decrypts ciphertext of $\mathrm{ECU}_{\mathrm{S}}$'s transmitted data; 9) is the process that $\mathrm{ECU}_{\mathrm{S}}$ generates the signature of the attribute-based key and transmit it to $\mathrm{ECU}_{\mathrm{R}j}$; 10) is the process that $\mathrm{ECU}_{\mathrm{R}j}$ performs the final decryption on partially decrypted data and verifies the attribute-based key with the signature; and 10), 11), 12) are the mutual authentication process between $\mathrm{ECU}_{\mathrm{R}j}$ and $\mathrm{ECU}_{\mathrm{S}}$.

V. Security Analysis

This section proves the security of the cryptographic scheme proposed by Sec. III-B and the proposed protocol based on the security definition and model of Sec. II-D.

A. Security Analysis of Enhanced Attribute-Based Encryption With Hidden Policy and Credential

Theorem 1 (Confidentiality of EABEHP): The proposed EABEHP is with ciphertext indistinguishability against chosen plaintext attack and restricted user coalition attack (C-IND-CPA-RUCA) if the decisional Diffie-Hellman (DDH) assumption holds.[5]

Proof Sketch: The proof of C-IND-CPA-RUCA for the proposed EABEHP consists of two parts. One is the confidentiality of the produced ciphertext, i.e., C, in EABEHP. Moreover, sC' and sC'_r are also considered as the ciphertext of EABEHP since they are transformed from C and the only difference is that the positions of tuples of ciphertext are shuffled. Thus, the C-IND-CPA-RUCA security can be proven according to the same security proof for the proposed PEAPOD in [27] since the structure of the ciphertext in EABEHP is the same as that in PEAPOD.

The other is the confidentiality of C, sC', sC'_r, and sC''_r against the proxy for **ProxyDecrypt1**. The second part in the proof of C-IND-CPA-RUCA considers that the confidentiality is guaranteed against the proxy even though AK_U and RK_U are known to the proxy. RK_U is the re-encryption key and it preserves the same structure of that in the PEAPOD scheme. Thus, the exposure of RK_U will not affect the confidentiality of the ciphertext in EABEHP. In addition, the exposure of AK_U will not affect the confidentiality since additional secret key ω_k, unknown to the proxy, is introduced to protect the secrets, $a_{j,i}$, contained in AK_U. Thus, without known ω_k, the proxy cannot eliminate the factor, g^{ω_k}, of blinding the message in the ciphertext to break the confidentiality of EABEHP.

Theorem 2 (Policy Privacy of EABEHP): Policy privacy holds if no one, including the security agent can learn any knowledge of the given policy T for encryption in the proposed EABEHP scheme.

Proof Sketch: In EABEHP, a message M to be encrypted will first be encoded as p_i for $i \in I$ such that $\Pi_{t_i=1 \wedge i \in I} p_i = M$. Here, $T = \{t_i\}_{i \in I}$ is a policy set, where $t_i = 1, 0, -1$ if the attribute i is required, irrelevant, and unrequired, respectively. Here, $p_i = 1$ when $t_i = 0$ and $p_i = \alpha_i$ when $t_i = -1$, for randomly selected $\alpha_i \in \mathbb{Z}_q$. Afterwards, one can then encrypt each p_i as (A, B_i, C) with the public key of its corresponding attribute i. Thus, the only way to learn the given policy I depends on the generated p_i. However, each p_i is encrypted using ElGamal encryption [36], which is the primitive of EABEHP with indistinguishability under chosen plaintext attack (IND-CPA) security. Thus, no one can learn any knowledge from p_i by the ciphertext (A, B_i, C). Hence, no one, including the security agent, can learn the knowledge of policy. Consequently, EABEHP is with policy privacy.

Theorem 3 (Credential Privacy of EABEHP): The proposed EABEHP is with hidden credentials against outsider and decryption proxy if the underlying hash function is a pseudorandom function, and the shuffle function is a pseudorandom permutation.

Proof Sketch: Since the attribute information, i.e., AK_{U_r} or $\hat{I}_r$, of each user will be exposed when during the execution of **ProxyDecrypt1** or **Extract** function, the privacy of user credential is guaranteed if the original attribute information cannot be disclosed. 1) AK_{U_r} is a combination of ω_k and user secret keys for each attribute generated by the **TransformUserKey** algorithm, where $\omega_k = \mathrm{H}(r_k || K_{\mathrm{group}})$, and K_{group} is a pre-distributed key to legitimate users in the system. Therefore, no one has non-negligible probability to distinguish AK_{U_r} by distinguishing a ω_k from random string based on the security of pseudorandom function.

2) $\hat{I}_r$ is the set of inverse permuted attribute indices from the set of original attribute indices by a pseudorandom permutation, shuffle function SH with a given ω_k, which is only known between users. Thus, no one has non-negligible probability to distinguish a permuted index from an original index based on the security of pseudorandom permutation. Thus, the proposed EABEHP is with hidden credentials based on the security of the pseudorandom function and permutations.

B. Security Analysis of EC-SVC Protocol

In this subsection, we present the security analysis of the protocol proposed in Sec. IV-B. Specifically, we prove that mutual authentication, attribute-based key exchange, policy privacy, and credential privacy have been achieved as follows.

Theorem 4 (EC-SVC Security): The proposed EC-SVC protocol is said to be the attribute-based authenticated key exchange protocol with hidden policy and credential

[5]Note that the proposed EABEHP does not achieve collusion resistance. However, in the in-vehicle environment, ECUs generally have microchips or devices that provide the hardware-based security such as TPM and hardware security module (HSM) (e.g., Infineon's AURIX 32-bit microcontroller) [34], [35], which protect and store sensitive passwords or encryption keys. Therefore, attackers cannot extract the secret of each ECU due to the protection of hardware-based security, so the practice of a collusion attack is infeasible. Therefore, the collusion resistance can be provided by using an existing hardware-based security solution with reasonable costs, so it is not considered in the design of the proposed EABEHP.

if H is a pseudorandom function, E_S is a pseudorandom permutation, and EABEHP is a C-IND-CPA-RUCA-secure and P-IND-CPA-RUCA-secure attribute-based encryption scheme.

The advantage $\mathrm{Adv}_{\mathcal{A}}^{\text{EC-SVC}}$ that an attacker $\mathcal{A}$ break the security of EC-SVC protocol are given by

$$\mathrm{Adv}_{\mathcal{A}}^{\text{EC-SVC}} \leq 11\mathrm{Adv_H} + 5\mathrm{Adv}_{E_S} + 2\mathrm{Adv}_{\text{C-IND-CPA}} + 2\mathrm{Adv}_{\text{P-IND-CPA}}, \tag{8}$$

where $\mathrm{Adv_H}$ is an advantage that breaks the security of pseudorandom function, Adv_{E_S} is an advantage that breaks the security of pseudorandom permutation, $\mathrm{Adv}_{\text{C-IND-CPA}}$ is an advantage that breaks the C-IND-CPA-RUCA security of EABEHP, and $\mathrm{Adv}_{\text{P-IND-CPA}}$ is an advantage that breaks the P-IND-CPA-RUCA security of EABEHP.

We proceed with the security game to prove the security of the proposed protocol. The security game proceeds each four requirements mentioned above and claims that the advantages of $\mathcal{A}$ for the proposed protocol can be negligible, depending on the advantages of $\mathcal{A}$ in each game, where $\mathcal{A}$ is an attacker that breaks the security of mutual authentication, attribute-based key exchange, policy privacy, and credential privacy. We denote $\mathrm{Adv}_{\mathcal{A},i}^{\text{EC-SVC}}$ as the advantage of $\mathcal{A}$ in game G_i.

Game G_0: This is a real game, $\mathcal{A}$ has access to EABEHP's master public key MPK, all ECU's identity (ID) $\{ID_i\}_{i=0,1,\ldots}$. In addition, $\mathcal{A}$ has the ability to query all oracles specified in Sec. II-D and knows all the structure of the protocol. Since this paper has shown that EABEHP can be proven IND-CPA secure by simulating EABEHP in Sec. V-A, all the parameters related to EABEHP can be successfully simulated. Therefore, we have $\mathrm{Adv}_{\mathcal{A}}^{\text{EC-SVC}} = \mathrm{Adv}_{\mathcal{A},0}^{\text{EC-SVC}}$.

Game G_1 (Mutual Authentication): In the game G_1, We describe the events of the game as follows. E_1 is an event in which $\mathcal{A}$ impersonates $\mathrm{ECU_S}$ by sending the correct σ_1 to the SA. The E_2 is an event in which $\mathcal{A}$ impersonates the SA by sending the correct σ_2 to the $\mathrm{ECU_S}$. The E_3 is an event in which $\mathcal{A}$ impersonates the ECU_{Rj} by sending the correct σ_{4j} to the SA. The E_4 is an event in which $\mathcal{A}$ impersonates the SA by sending the correct σ_{5j} to ECU_{Rj}. The E_5 is an event in which $\mathcal{A}$ impersonates the ECU_{Rj} by sending the correct $\sigma_{8j}||C_{3j}$ to the $\mathrm{ECU_S}$. The E_6 is an event in which $\mathcal{A}$ impersonates the $\mathrm{ECU_S}$ by sending the correct $\sigma_9||C_4$ to the ECU_{Rj}. We construct a simulator S_1 of the EC-SVC that interacts with $\mathcal{A}$ as the security game defined in Definition 4. In addition, S_1 is provided with the master public key of EABEHP to successfully simulate EC-SVC. If the E_1 happens, S_1 can exploit the ability of $\mathcal{A}$ to break the underlying pseudorandom function security. Hence, we have

$$\begin{aligned}\mathrm{Adv_H} &\geq \{\Pr[S_H, E_1] + \Pr[S_H, \neg E_1]\} - \frac{1}{2} \\ &= \{\Pr[S_H|E_1]\times\Pr[E_1] + \Pr[S_H|\neg E_1]\times(1-\Pr[E_1]\} - \frac{1}{2} \\ &= \{1 \times \mathrm{Adv}_{E_1} + \frac{1}{2}\times(1-\mathrm{Adv}_{E_1})\} - \frac{1}{2} = \frac{\mathrm{Adv}_{E_1}}{2},\end{aligned} \tag{9}$$

where S_H is the event of distinguishing a pseudorandom function from a truly random function successfully, the $\neg E_1$ is the complementary event of the E_1, Adv_{E_1} is the advantage of the E_1, which is the probability that an attacker sends a valid σ_1 to impersonate an $\mathrm{ECU_S}$. Therefore, we have $\mathrm{Adv}_{E_1} \leq 2\mathrm{Adv_H}$. For the probabilities of events E_2, E_3, and E_4, we have $\mathrm{Adv}_{E_2} \leq 2\mathrm{Adv_H}$, $\mathrm{Adv}_{E_3} \leq 2\mathrm{Adv_H}$, and $\mathrm{Adv}_{E_4} \leq 2\mathrm{Adv_H}$. The security analysis regarding E_5 is divided into two cases: First, when E_5 happened, S_1 can also break the security of underlying pseudorandom function or pseudorandom permutation by exploiting the ability of $\mathcal{A}$. Thus, we have $\mathrm{Adv}_{E_5} \leq 2\mathrm{Adv}_{E_S}$, when S_1 simulates the protocol based on the function, which is either a pseudorandom permutation or a random permutation. In addition, when S_1 simulates the protocol based on the function, which is either a pseudorandom function or a random function, we have $\mathrm{Adv}_{E_5} \leq 2\mathrm{Adv_H}$. From the above, we have

$$\mathrm{Adv}_{E_5} \leq \mathrm{Adv}_{E_S} + \mathrm{Adv_H}. \tag{10}$$

Second, when E_5 happened, S_1 can also break the security of underlying pseudorandom permutation or C-IND-CPA-RUCA-secure EABEHP by exploiting the ability of $\mathcal{A}$. Thus, we have

$$\mathrm{Adv}_{E_5} \leq \mathrm{Adv}_{E_S} + \mathrm{Adv}_{\text{C-IND-CPA}}. \tag{11}$$

Through the results of both cases, we have

$$\mathrm{Adv}_{E_5} \leq \mathrm{Adv}_{E_S} + \frac{1}{2}(\mathrm{Adv_H} + \mathrm{Adv}_{\text{C-IND-CPA}}). \tag{12}$$

In the same way, we have $\mathrm{Adv}_{E_6} \leq \mathrm{Adv}_{E_S} + \frac{1}{2}(\mathrm{Adv_H} + \mathrm{Adv}_{\text{C-IND-CPA}})$. Finally, we have

$$\mathrm{Adv}_{\mathcal{A},0}^{\text{EC-SVC}} \leq \mathrm{Adv}_{\mathcal{A},1}^{\text{EC-SVC}} + 9\mathrm{Adv_H} + 2\mathrm{Adv}_{E_S} + \mathrm{Adv}_{\text{C-IND-CPA}}. \tag{13}$$

Game G_2 (Attribute-Based Key Exchange): In game G_2, we construct a simulator S_2 that interacts with $\mathcal{A}$ in the security games defined in Definition 5. S_2 is provided with the master public key of EABEHP to successfully simulate EC-SVC. $\mathcal{A}$ queries the **Test** after interacting with the security game with S_2. S_2 responds to the $\mathcal{A}$ with an attribute-based key K or a random string according to a random bit. If $\mathcal{A}$ can successfully guess the attribute-based key K, S_2 can also break the security of underlying pseudorandom permutation or C-IND-CPA-RUCA-secure EABEHP by exploiting the ability of $\mathcal{A}$. Therefore, we have

$$\begin{aligned}\mathrm{Adv}_{E_S} &\geq \{\Pr[S_{E_S}, E_{AKE}] + \Pr[S_{E_S}, \neg E_{AKE}]\} - \frac{1}{2} \\ &= \{\Pr[S_{E_S}|E_{AKE}] \times \Pr[E_{AKE}] \\ &\quad + \Pr[S_{E_S}|\neg E_{AKE}] \times (1 - \Pr[E_{AKE}]\} - \frac{1}{2} \\ &= \{1 \times \mathrm{Adv}_{E_{AKE}} + \frac{1}{2} \times (1 - \mathrm{Adv}_{E_{AKE}})\} - \frac{1}{2} \\ &= \frac{\mathrm{Adv}_{E_{AKE}}}{2},\end{aligned} \tag{14}$$

where $\mathrm{Adv}_{E_{AKE}}$ is the advantage of the E_{AKE}, which is the probability that an attacker distinguishes the attribute based key K from a random string. Therefore, we have $\mathrm{Adv}_{E_{AKE}} \leq 2\mathrm{Adv}_{E_S}$. Similar to the game G_1, we have $\mathrm{Adv}_{E_{AKE}} \leq 2\mathrm{Adv}_{\text{C-IND-CPA}}$. From the above, we have

$$\mathrm{Adv}_{E_{AKE}} \leq \mathrm{Adv}_{E_S} + \mathrm{Adv}_{\text{C-IND-CPA}}. \tag{15}$$

Therefore, we have

$$\mathrm{Adv}_{\mathcal{A},1}^{\text{EC-SVC}} \leq \mathrm{Adv}_{\mathcal{A},2}^{\text{EC-SVC}} + \mathrm{Adv}_{\mathrm{E_S}} + \mathrm{Adv}_{\text{C-IND-CPA}}. \quad (16)$$

Game G_3 (Policy Privacy): In game G_3, we construct a simulator S_3 that interacts with $\mathcal{A}$ in the security games defined in Definition 6. S_3 is provided with the master public key of EABEHP to successfully simulate EC-SVC. $\mathcal{A}$ queries the **TestPolicy** after interacting with the security game with S_3. S_3 responds to the $\mathcal{A}$ with an P_0 or P_1 according to a random bit. If $\mathcal{A}$ can successfully guess the correct policy, then $\mathcal{A}$ has the advantage of breaking P-IND-CPA-RUCA security of EABEHP. Therefore, we have

$$\begin{aligned}
&\mathrm{Adv}_{\text{P-IND-CPA}} \\
&\geq \{\Pr[\mathrm{S}_{\text{P-IND-CPA}}, \mathrm{E_{PP}}] + \Pr[\mathrm{S}_{\text{P-IND-CPA}}, \neg\mathrm{E_{PP}}]\} - \frac{1}{2} \\
&= \{\Pr[\mathrm{S}_{\text{P-IND-CPA}}|\mathrm{E_{PP}}] \times \Pr[\mathrm{E_{PP}}] \\
&\quad + \Pr[\mathrm{S}_{\text{P-IND-CPA}}|\neg\mathrm{E_{PP}}] \times (1 - \Pr[\mathrm{E_{PP}}]\} - \frac{1}{2} \\
&= \{1 \times \mathrm{Adv}_{\mathrm{PP}} + \frac{1}{2} \times (1 - \mathrm{Adv}_{\mathrm{PP}})\} - \frac{1}{2} = \frac{\mathrm{Adv}_{\mathrm{PP}}}{2},
\end{aligned} \quad (17)$$

where $\mathrm{E_{PP}}$ is the event that distinguishing the correct policy from random string with additional advantage, and $\mathrm{Adv}_{\mathrm{PP}}$ is the advantage of breaking policy privacy. Thus, we have

$$\mathrm{Adv}_{\mathcal{A},2}^{\text{EC-SVC}} \leq \mathrm{Adv}_{\mathcal{A},3}^{\text{EC-SVC}} + 2\mathrm{Adv}_{\text{P-IND-CPA}}. \quad (18)$$

Game G_4 (Credential Privacy): In game G_4, we construct a simulator S_4 that interacts with $\mathcal{A}$ in the security games defined in Definition 7. S_4 successfully simulates EC-SVC with the supplied EABEHP's master public key. After interacting with the security game with S_4, $\mathcal{A}$ queries the **TestCert**. S_4 responds to $\mathcal{A}$ with a target credential or randomly selected credential according to a random bit. E_9 is an event in which credentials are successfully guessed in the **ProxyDecrypt1** algorithm by adversary $\mathcal{A}$. E_{10} is a case in which the credentials are successfully guessed in the **Extract** algorithm by $\mathcal{A}$. If the E_9 happens, $\mathcal{A}$ has the advantage of breaking pseudorandom function security. Therefore, we have

$$\begin{aligned}
\mathrm{Adv_H} &\geq \{\Pr[\mathrm{S_H}, \mathrm{E}_9] + \Pr[\mathrm{S_H}, \neg\mathrm{E}_9]\} - \frac{1}{2} \\
&= \{\Pr[\mathrm{S_H}|\mathrm{E}_9] \times \Pr[\mathrm{E}_9] + \Pr[\mathrm{S_H}|\neg\mathrm{E}_9] \times (1 - \Pr[\mathrm{E}_9]\} - \frac{1}{2} \\
&= \{1 \times \mathrm{Adv}_{\mathrm{E}_9} + \frac{1}{2} \times (1 - \mathrm{Adv}_{\mathrm{E}_9})\} - \frac{1}{2} = \frac{\mathrm{Adv}_{\mathrm{E}_9}}{2},
\end{aligned} \quad (19)$$

where $\mathrm{Adv}_{\mathrm{E}_9}$ is the advantage of the E_9, which is the probability that an attacker distinguishes the real user private key from a random string. Therefore, we have $\mathrm{Adv}_{\mathrm{E}_9} \leq 2\mathrm{Adv_H}$. In the same way, for E_{10}, we have $\mathrm{Adv}_{\mathrm{E}_{10}} \leq 2\mathrm{Adv}_{\mathrm{E_S}}$. Therefore, $\mathrm{Adv_{CP}} \leq 2\mathrm{Adv_H} + 2\mathrm{Adv}_{\mathrm{E_S}}$, where $\mathrm{Adv_{CP}}$ is the advantage of breaking credential privacy. Thus, we have

$$\mathrm{Adv}_{\mathcal{A},3}^{\text{EC-SVC}} \leq \mathrm{Adv}_{\mathcal{A},4}^{\text{EC-SVC}} + 2\mathrm{Adv_H} + 2\mathrm{Adv}_{\mathrm{E_S}}. \quad (20)$$

There are no additional advantages beyond those analyzed in the game above. Thus, by equations (13), (16), (18), and (20) we can claim that the advantages of $\mathcal{A}$ to the proposed EC-SVC are as given by

$$\begin{aligned}
\mathrm{Adv}_{\mathcal{A}}^{\text{EC-SVC}} \leq\ & 11\mathrm{Adv_H} + 5\mathrm{Adv}_{\mathrm{E_S}} + 2\mathrm{Adv}_{\text{C-IND-CPA}} \\
& + 2\mathrm{Adv}_{\text{P-IND-CPA}}.
\end{aligned} \quad (21)$$

Theorem 5 (Universal Composable Security): The proposed EC-SVC is said to be of universally composable security if H is a pseudorandom function, $\mathrm{E_S}$ is a pseudorandom permutation, and EABEHP is a CPA-secure encryption.

Proof Sketch: Universal composable (UC) [37] security is to prove the security of the proposed protocol if the protocol is modularly composed with multiple protocols and/or under arbitrary concurrent performing protocol instances. Since EC-SVC only contains one protocol, we only need to prove the security under the setting of concurrent protocol execution for UC security. Theorem 4 has shown that the replacement of ideal functions (i.e., random function, random permutation, and ideal CPA encryption function) with real functions (i.e., pseudorandom function, pseudorandom permutation, and proposed CPA-secure encryption) in the same protocol execution is indistinguishable due to the indistinguishability of pseudorandom function, pseudorandom permutation, and the security of CPA-secure encryption. Thus, one can prove the UC security of EC-SVC under the security game, where no attacker can distinguish the two selected protocol instances with a non-negligible advantage, when one instance uses a real function and the other instance uses an ideal function depending on a random bit. Thus, the UC security of the proposed EC-SVC can be guaranteed for the security proof of each security properties given as the above.

Finally, the overall security comparison between security protocols and related works is shown in Table I on the next page. The work satisfies all the security requirements without mounting additional components on ECUs.

TABLE I
COMPARISON ON SECURITY WITH RELATED WORKS

	[10] [14] [17]	[12] [15] [19]	[13] [16] [20]	Our work (EC-SVC)
Message Authentication and Integrity	√	√	√	
	√	√	√	√
	√	√	√	
Data Confidentiality	√	×	×	
	×	×	√	√
	√	√	√	
Resistance to Replay Attacks	√	×	×	
	√	√	√	√
	√	√	×	
Attribute-based Access Control	×	×	×	
	×	×	×	√
	×	×	√	
Hidden policy and hidden credentials	-	-	-	
	-	-	-	√
	-	-	×	

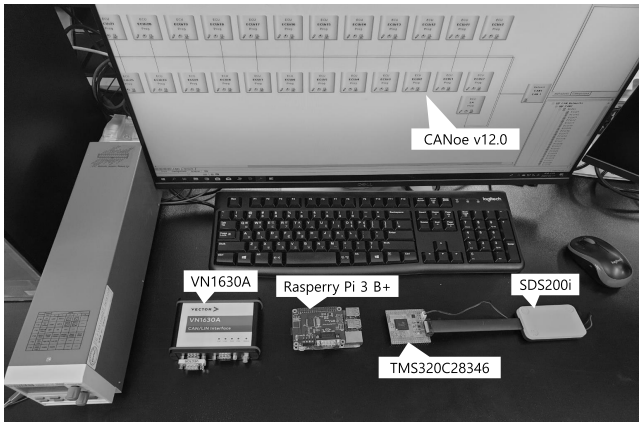

Fig. 4. Performance evaluation environment.

VI. Performance Analysis

In this section, we evaluate the performance in various aspects for demonstrating that the proposed security protocol is practical in in-vehicle scenarios. This work builds up the testbed based on the hardware and software which are the Raspberry Pi, TMS320C28346, and CANoe by Vector Co [25]. Unless otherwise specified, the simulation environment in Fig. 4 and the specifications of the equipment in Table II are used. The testbed adopts CANoe to implement in-vehicle network based on the flexible data rate (CAN-FD) standard [38]. This section first analyzes the execution time of each cryptographic algorithm for each device. We then evaluate the performance of the proposed security protocols in the simulation environment implemented.

A. Cryptographic Algorithm Evaluation

This subsection evaluates the execution time of the different cryptographic algorithms (i.e., SHA-256, AES-128, and EABEHP). For the construction of the system, we take the Raspberry Pi as SA and TMS320C28346 as ECU. We then implement the algorithms of the proposed protocol on the ECUs and SA, separately. In the implementation of the protocol and EABEHP functions, we use the functions of BigInteger, AES, and SHA in the Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE) for the operations of the variables, encryption, and hash functions by the java standard and cryptographic extension APIs. Since EABEHP is based on ElGamal encryption, it is implemented in a manner similar to that of ElGamal encryption [36]. For better accuracy, we measure the execution time 10,000 times repetitively and obtain the average execution time. In the proposed security protocol, 48 bytes input is used to the SHA256 algorithm, and 16 bytes output is obtained by truncated MAC [16], [17]. In addition, 48 bytes out of 64 bytes in the data payload is used as input data to AES128 algorithm. The length of all messages sent by the ECU and the SA is 64 bytes, and the remnant is assumed to be padded. Under this description, the measured execution time of various cryptographic algorithm is shown in Table. III. Table III show the cryptographic algorithm execution time of SHA-256, AES128 Encryption, and AES128 Decryption at the ECU and the SA.

TABLE II
Hardware and Software for Performance Evaluation

Model	Note
Raspberry Pi 3 B+ (Single-board Computer)	Clock speed : 1.4GHz or 600MHz
TI TMS320C28346 (Micro Controller Unit (MCU))	Clock speed : 300MHz
SDS200i	JTAG Emulator
VN1630A	CAN-FD Network Interface
Code Composer Studio V9.3	MCU Compiler
CANoe V12.0	In-vehicle Network simulator

TABLE III
Execution Time of Cryptographic Algorithm

	Algorithm execution time (μs)		
Algorithm	**SHA-256**	**AES128(Enc)**	**AES128(Dec)**
ECU	130.8	149.5	198.9
SA (600MHz)	8.4	12.7	13.8
SA (1.4GHz)	3.6	5.4	6.7

TABLE IV
Execution Time of EABEHP Algorithm

		Algorithm execution time (ms)			
Number of system attributes		4	8	12	16
		20	24	28	32
EABEHP Encrypt+Shuffle	ECU	144.7	241.1	338.8	436.9
		529.5	635.5	714.8	817.9
EABEHP TransformCipherText	SA (600MHz)	7	13	20.9	27.8
		34.4	41.8	47.6	54.8
	SA (1.4GHz)	3	6	9	12
		14.5	17.5	21.2	23.6
Number of receiver attributes		4	8	12	16
		20	24	28	32
EABEHP Extract+ProxyDecrypt1	SA (600MHz)	1.92	2.05	2.25	2.46
		2.65	3	3.24	3.64
	SA (1.4GHz)	0.82	0.89	0.96	1.08
		1.12	1.25	1.44	1.56

Table IV shows the EABEHP **Encrypt** and **Shuffle** algorithm execution time at the ECU,[6] and EABEHP **TransformCipherText** execution time and EABEHP **Extract** and **ProxyDecrypt1** execution time at the SA.

B. Security Protocol Evaluation

This subsection measures the execution time of the security protocol, based on the cryptographic algorithm evaluation. Using the CANoe v12.0 by Vector Co, we implement an evaluation environment similar to the real CAN-FD. The execution

[6]Note that, it was not possible to perform EABEHP **Encrypt** and **Shuffle** at the ECU due to the hardware limitation. Hence, we obtain the execution time of them by measuring of Raspberry Pi and then scaling the time by the execution time ratio of other operations (e.g., SHA-256 and AES128 in Table IV). We expect that the ECU, developed in the near future, will be able to support the advanced cryptographic operations.

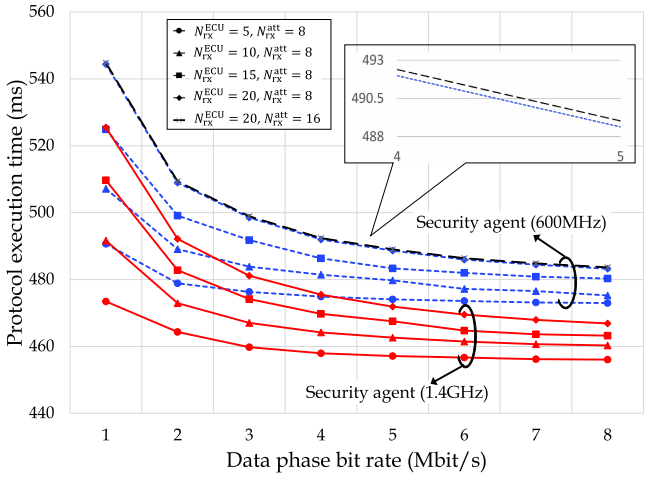

Fig. 5. Execution time of the proposed protocol as a function of the data phase bit rate for different numbers of receiver-ECUs, $N_{\mathrm{rx}}^{\mathrm{ECU}}$, and receiver attributes, $N_{\mathrm{rx}}^{\mathrm{att}}$. Here, the number of system attributes is 16.

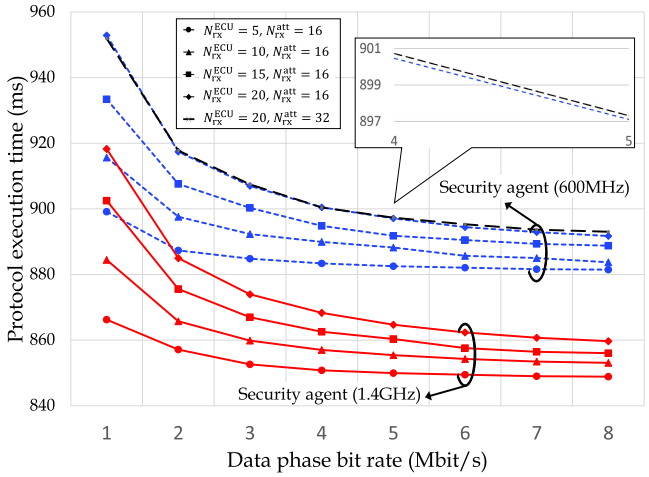

Fig. 6. Execution time of the proposed protocol as a function of the data phase bit rate for different numbers of receiver-ECUs, $N_{\mathrm{rx}}^{\mathrm{ECU}}$, and receiver attributes, $N_{\mathrm{rx}}^{\mathrm{att}}$. Here, the number of system attributes is 32.

time of the proposed protocol is measured by considering the communication delay as well as the execution time of the cryptographic algorithms in Tables III and IV at the CANoe virtual ECU node. Note that this work also achieves several additional features such as attribute-based access control and privacy-preserving for corrupted devices, which are not achieved in existing in-vehicle security works [10]–[19]. Furthermore, this work proposes the novel edge computing-based security protocol that achieves a higher level of security and has reasonable latency in-vehicle systems. For instance, when the total number of ECUs is 16, the computation time in [19] was 2482.1 ms for Cao IBKE [39], which is measured by Aurix TC297. However, the computation time of the proposed protocol is 424 ms, which is much lower than that in [19], even it is measured by TMS320C28346 that has lower computing capability than Aurix TC297.[7] In the following, we present the performance evaluation in various aspects to show that the proposed security protocol is practical.

Figures 5 and 6 show the execution time of the attribute-based key exchange protocols for different data phase bit rates. We perform the evaluation with the fixed arbitration phase bit rate of 0.5Mbit/s and adjust the data phase bit rate from 1Mbit/s to 8Mbit/s. The number of system attributes and receiver attributes is set to 16 and 8, respectively, in Fig. 5, and 32 and 16, respectively, in Fig. 6. The measurement results when the SA clock speed is 1.4GHz and 600MHz are represented by the solid and dotted lines, respectively.

From Figs. 5 and 6, we can see that the protocol execution time decreases as the data phase bit rate increases since the communication delay in CAN becomes smaller. By comparing the results with $N_{\mathrm{rx}}^{\mathrm{att}} = 16$ and $N_{\mathrm{rx}}^{\mathrm{att}} = 8$ in Fig. 5 and those with $N_{\mathrm{rx}}^{\mathrm{att}} = 32$ and $N_{\mathrm{rx}}^{\mathrm{att}} = 16$ in Fig. 6, we can see that the number of the receiver attributes has little impact on the protocol execution time, while the number of system attributes affects significantly on the protocol execution time. However, note that even with 32 system attributes, which is quite a large number to classify ECUs since there are many ECUs with overlapping roles, the execution time of the proposed protocol is less than 1 second.

[7]Note that 424 ms is the computation time, which is different to the protocol execution time includes the communication delay as well.

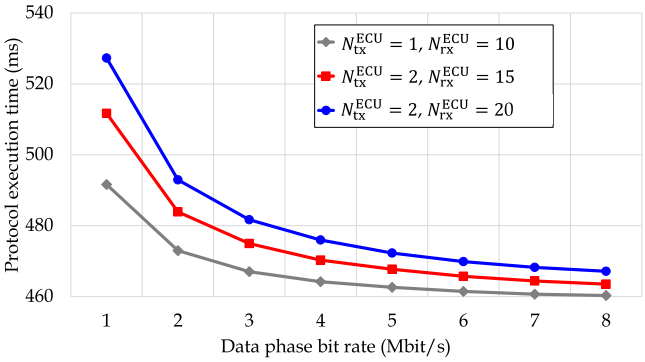

Fig. 7. Execution time of the proposed protocol as a function of the data phase bit rate, for different numbers of sender-ECUs, $N_{\mathrm{tx}}^{\mathrm{ECU}}$. Here, the number of receiver-ECUs, $N_{\mathrm{rx}}^{\mathrm{ECU}}$ allocated to each sender-ECU is 10.

Figure 7 shows the protocol execution time versus the data phase bit rate for different numbers of senders and receivers, denoted by $N_{\mathrm{tx}}^{\mathrm{ECU}}$ and $N_{\mathrm{rx}}^{\mathrm{ECU}}$, respectively. We assume that 10 receivers access the message from one sender. The following number of senders and receivers are used: 1 sender and 10 receivers, 2 senders and 15 receivers (5 receivers received messages from both senders), and 2 senders and 20 receivers (all receivers received messages only from one sender).

We can see that the protocol execution time difference among three cases becomes smaller as the data phase bit rate increases. This is because the number of communications is different in three cases, and the communication delay is inversely proportional to the data phase bit rate. Through the result of Fig. 7, we can see that if the data phase bit rate is high enough, the time taken to execute the proposed attribute-based key exchange process on all ECUs will not be significantly affected by the in-vehicle network size, i.e., the numbers of sender-ECUs and receiver-ECUs. Therefore, the

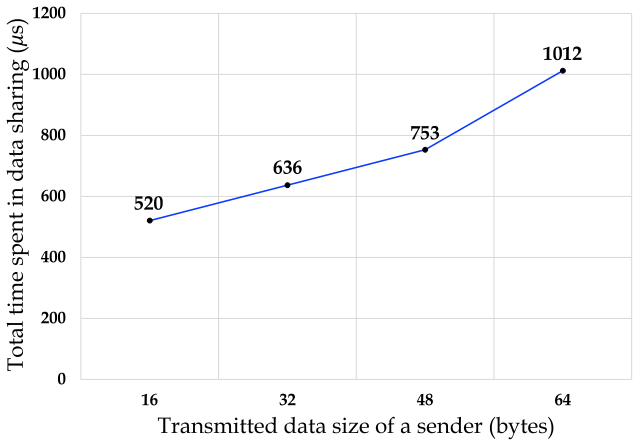

Fig. 8. Total time spent in data sharing based on transmitted data size of a sender.

proposed protocol is expected to be executed in a reasonable time even for different in-vehicle network sizes, which shows the feasibility and the practicality of the proposed protocol.

Finally, we describe the practicality of the proposed secure communication system. The proposed secure communication system with attribute-based access control for CAN consists of two procedures: 1) key exchange and 2) secure data sharing procedures. The execution time in Figs. 5 and 6 is that of the proposed key exchange protocol, which needs to be performed only once before the secure data sharing. Since it only takes less than 1 second, it can be performed and finished during triggering a vehicle without causing any inconvenience to drivers as well as functionalities in a vehicle.

After the key exchange procedure, we can have secure data sharing by encrypting the transmitted messages using the exchanged key. As the payload of CAN-FD bus is 64 bytes and 16 byte truncated MAC needs to be added for the secure data sharing, the maximum transmittable data sizes becomes 48 bytes from 64 bytes. That means additional one transmission is required when the transmitted data size of a sender ECU is larger than 48 bytes. In addition, some computation latency occurs while encrypting and decrypting a message at a sender ECU and a receiver ECU, respectively.

Figure 8 shows the measured total time spent by the data sharing protocol (i.e., the total time for encryption, decryption, and transmission) according to the different sizes of transmitted data at a sender. For this figure, the data phase bit rate was set to 8Mbps and AES128 is adopted for the symmetric encryption. From Fig. 8, we can see that the total spent time in the data sharing protocol is around 0.5 - 1ms, which is quite low latency, even for the case with additional one transmission (i.e., the case of 64 byte data size). Note that this latency is also shorter than the latency requirements of functionaries in a vehicle (e.g., less than 10 ms for control and management, 33 ms for safety data, and 150 ms for infotainment data) as shown in [40].

Furthermore, in the aspect of CAN-FD bus load, the load increases only when one more transmission is required due to larger data size than 48 bytes. Based on our measurement in CANoe, the load increase of the additional one transmission is negligible as it only increases about 0.015% of the bus load. Hence, the proposed protocols can be practically used for securing in-vehicle communications.

VII. Conclusion

This work proposes an edge computing-based in-vehicle security protocol with the attribute-based access control that achieves the privacy on the policy and credentials as well as the end-to-end security against insider attacker. The security of this protocol has been proved based on the security of pseudorandom function, pseudorandom permutation, and C-IND-CPA-RUCA and P-IND-CPA-RUCA EABEHP. Furthermore, the performance analysis of the proposed protocol shows the effect of protocol execution time according to the data phase bit rate, the number of system attributes, the number of receiver attributes, and the number of sender and receiver-ECUs. This shows that a high-security level can be achieved with reasonable latency for in-vehicle communications among ECUs with limited capabilities.

Finally, this work has demonstrated the feasibility of the proposed protocol for secure in-vehicle communications with fine-grained access control through comprehensive experiments. Although the proposed protocol is mainly developed for CAN-FD, the lightweight design in computation and communications makes the integration of the proposed protocol with the other in-vehicle communication protocol practical. Especially, the feasibility demonstration can be still valid for the in-vehicle communication protocols with higher data rate than CAN-FD such as FlexRay, MOST, and Ethernet [41]–[43]. Note that the proposed security protocol can also be used in an in-vehicle communication protocol with lower data rate and smaller maximum payload, such as CAN, but the protocol execution time will be longer and the security level can be lessened for data integrity protection. However, when a longer triggering time caused by longer protocol execution time is not a concern and the data integrity protection by 32-bit MAC is sufficient, the proposed protocol can be adopted to CAN without any other issues.

References

[1] W. Stephan, R. Solveig, and M. Markus, "Aspects of secure vehicle software flashing," in *Embedded Security in Cars*. Berlin, Germany: Springer, 2006, pp. 17–26.

[2] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. USENIX Secur. Symp.*, vol. 4, Aug. 2011, pp. 447–462.

[3] A. Greenberg, "Hackers remotely kill a jeep on the highway-with me in it," *Wired*, vol. 7, no. 2, p. 21, 2015.

[4] J. Golson, "Car hackers demonstrate wireless attack on Tesla model S," *Verge*, vol. 19, Sep. 2016. [Online]. Available: https://www.theverge.com/2016/9/19/12985120/tesla-model-s-hack-vulnerability-keen-labs

[5] A. Greenberg, "Hackers can steal a Tesla model S in seconds by cloning it's key fob," *Wired Mag.*, vol. 10, Sep. 2018. [Online]. Available: https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob/

[6] T. Hoppe, S. Kiltz, and J. Dittmann, "Security threats to automotive can networks-practical examples and selected short-term countermeasures," in *Proc. Int. Conf. Comput. Saf., Rel., Secur.*, Sep. 2008, pp. 235–248.

[7] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *The Ethics of Information Technologies*. Evanston, IL, USA: Routledge, 2020, pp. 119–134.

[8] J. Petit and S. E. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 546–556, Apr. 2015.

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

[10] H. Oguma, A. Yoshioka, M. Nishikawa, R. Shigetomi, A. Otsuka, and H. Imai, "New attestation based security architecture for in-vehicle communication," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2008, pp. 1–6.

[11] P. Kleberger, T. Olovsson, and E. Jonsson, "Security aspects of the in-vehicle network in the connected car," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2011, pp. 528–533.

[12] H. Schweppe, Y. Roudier, B. Weyl, L. Apvrille, and D. Scheuermann, "Car2X communication: Securing the last meter—A cost-effective approach for ensuring trust in Car2X applications using in-vehicle symmetric cryptography," in *Proc. IEEE Veh. Technol. Conf. (VTC Fall)*, Sep. 2011, pp. 1–5.

[13] A. V. Herrewege, D. Singelee, and I. Verbauwhede, "CANAuth—A simple, backward compatible broadcast authentication protocol for CAN bus," in *Proc. ECRYPT Workshop Lightweight Cryptogr.*, Nov. 2011, p. 20.

[14] B. Groza, S. Murvay, A. V. Herrewege, and I. Verbauwhede, "LiBrA-CAN: A lightweight broadcast authentication protocol for controller area networks," in *Proc. Int. Conf. Cryptol. Netw. Secur.*, Dec. 2012, pp. 185–200.

[15] R. Kurachi, Y. Matsubara, H. Takada, N. Adachi, Y. Miyashita, and S. Horihata, "CaCAN—Centralized authentication system in CAN (controller area network)," in *Proc. Int. Conf. Embedded Secur. Cars*, Nov. 2014, pp. 1–10.

[16] 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.

[17] 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.

[18] D. Pullen, N. A. Anagnostopoulos, T. Arul, and S. Katzenbeisser, "Using implicit certification to efficiently establish authenticated group keys for in-vehicle networks," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2019, pp. 1–8.

[19] B. Groza and P.-S. Murvay, "Identity-based key exchange on in-vehicle networks: CAN-FD & FlexRay," *Sensors*, vol. 19, no. 22, p. 4919, Nov. 2019.

[20] J. Cui, X. Chen, J. Zhang, Q. Zhang, and H. Zhong, "Toward achieving fine-grained access control of data in connected and autonomous vehicles," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7925–7937, May 2021.

[21] R. McMillan, "With hacking, music can take control of your car," *Computerworld*, vol. 14, Mar. 2011. [Online]. Available: https://www.computerworld.com/article/2748225/with-hacking–music-can-take-control-of-your-car.html

[22] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the IoT," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 725–730.

[23] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in *Proc. Annu. Int. Cryptol. Conf.* Santa Barbara, CA, USA: Springer, 2005, pp. 258–275.

[24] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[25] *Vector*. Accessed: May 6, 2020. [Online]. Available: https://www.vector.com/in/en/

[26] R.-H. Hsu, J. Lee, T. Q. S. Quek, and J.-C. Chen, "Reconfigurable security: Edge-computing-based framework for IoT," *IEEE Netw.*, vol. 32, no. 5, pp. 92–99, Sep. 2018.

[27] A. Kapadia, P. P. Tsang, and S. W. Smith, "Attribute-based publishing with hidden credentials and hidden policies," in *Proc. NDSS*, vol. 7, Feb. 2007, pp. 179–192.

[28] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, May 2007, pp. 321–334.

[29] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Proc. Annu. Int. Cryptol. Conf.*, Aug. 1993, pp. 232–249.

[30] M. C. Gorantla, C. Boyd, and J. M. G. Nieto, "Attribute-based authenticated key exchange," in *Proc. Australas. Conf. Inf. Secur. Privacy*, Jul. 2010, pp. 300–317.

[31] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *Proc. Annu. Int. Cryptol. Conf.*, Aug. 1996, pp. 1–15.

[32] M. Luby and C. Rackoff, "How to construct pseudorandom permutations from pseudorandom functions," *SIAM J. Comput.*, vol. 17, no. 2, pp. 373–386, Apr. 1988.

[33] W. Trappe, J. Song, R. Poovendran, and K. J. R. Liu, "Key distribution for secure multimedia multicasts via data embedding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2001, pp. 1449–1452.

[34] Trusted Computing Group. *Trusted Platform Module (TPM) 2.0 Library Specification*. Accessed: May 9, 2020. [Online]. Available: https://trustedcomputinggroup.org/

[35] A. Ludovic *et al.*, "Secure automotive on-board electronics network architecture," in *Proc. FISITA World Automot. Congr.*, Budapest, Hungary, vol. 8, 2010, pp. 1–9.

[36] Y. Tsiounis and M. Yung, "On the security of ElGamal based encryption," in *Proc. Int. Workshop Public Key Cryptogr.* Yokohama, Japan: Springer, 1998, pp. 117–134.

[37] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *Proc. 42nd IEEE Symp. Found. Comput. Sci.*, 2001, pp. 136–145.

[38] F. Hartwich *et al.*, "CAN with flexible data-rate," in *Proc. Int. CAN Conf.*, 2012, pp. 1–9.

[39] X. Cao, W. Kou, and X. Du, "A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges," *Inf. Sci.*, vol. 180, no. 15, pp. 2895–2903, 2010.

[40] Z. Sheng, D. Tian, V. C. M. Leung, and G. Bansal, "Delay analysis and time-critical protocol design for in-vehicle power line communication systems," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 3–16, Jan. 2018.

[41] R. Makowitz, "FlexRay—A communication network for automotive control systems," in *Proc. IEEE Int. Workshop Factory Commun. Syst.*, Jun. 2006, pp. 207–212.

[42] C. Most, "Most media oriented system transport—Multimedia and control networking technology," *MOST Specification Rev.*, vol. 2, no. 3, 2004.

[43] P. Hank, S. Müller, O. Vermesan, and J. Van Den Keybus, "Automotive ethernet: In-vehicle networking and smart mobility," in *Proc. Des., Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2013, pp. 1735–1739.

Donghyun Yu (Graduate Student Member, IEEE) received the B.E. degree from the School of Undergraduate Studies, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, South Korea, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Information and Communication Engineering. His research interests include network and communication security, cryptographic protocols, applied cryptography, and vehicle security.

Ruei-Hau Hsu (Member, IEEE) received the B.S. and M.S. degrees in computer science from Tunghai University, Taiwan, in 2002 and 2004, respectively, and the Ph.D. degree in computer science and engineering from National Sun Yat-sen University, Kaohsiung, Taiwan, in 2012.

He was a Post-Doctoral Research Fellow with the Department of Computer Science, National Chiao Tung University, from 2012 to 2014; and the iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design, from 2014 to 2017. He was a Scientist with the Data Storage Institute (DSI) and the Institute for Infocomm Research (I2R), Agency for Science, Technology and Research (A*STAR), Singapore. He is currently an Assistant Professor with the Department of Computer Science and Engineering, National Sun Yat-sen University. In 2012, he was a member of the Phi Tau Phi Scholastic Honor Society. He received the Best Doctoral Dissertation Award twice from the Institute of Information and Computing Machinery and the Best Doctoral Dissertation Award from the Chinese Cryptology and Information Security in 2012. From August 2007 to December 2007, he was with the International Collaboration for Advancing Security Technology (iCAST) Program as a Visiting Scholar with Carnegie Mellon University (CMU), USA. From June 2010 to September 2010 and from March 2011 to February 2012, he received scholarships granted by the Deutscher Akademischer Austausch Dienst (DAAD), Germany; and the National Science Council (NSC), Taiwan, as a Visiting Scholar with the Center for Advanced Security Research Darmstadt (CASED), Technische Universitat Darmstadt. He also served as the Publication Co-Chair for the 16th IEEE Sponsored Asia-Pacific Network Operations and Management Symposium (APNOMS) in 2014, the Technical Program Committees of APNOMS in 2016, the International Conference of Internet of Things, Embedded Systems and Communications (IINTEC) in 2017, and the 2018 IEEE Conference on Dependable and Secure Computing (DSC) in 2018; and the Finance Chair for the ACM Symposium on Information, Computer and Communications Security in 2020 (ASIACCS 2020), and the Technical Program Committees of the Ninth ACM ASIA Public-Key Cryptography Workshop (APKC 2022) held in conjunction with ACM AsiaCCS 2022.

Jemin Lee (Member, IEEE) received the B.S. (Hons.), M.S., and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2004, 2007, and 2010, respectively.

She was a Post-Doctoral Fellow with the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, from 2010 to 2013; a Temasek Research Fellow with the iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design (SUTD), Singapore, from 2014 to 2016; and an Associate Professor with the Department of Information and Communication Engineering, Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, South Korea, from 2016 to 2021. She is currently an Associate Professor with the Department of Electrical and Computer Engineering, Sungkyunkwan University (SKKU), Suwon, South Korea. Her current research interests include wireless communications, wireless security, intelligent networking, and blockchain networks. She received the Haedong Young Engineering Researcher Award in 2020, the IEEE ComSoc Young Author Best Paper Award in 2020, the IEEE ComSoc AP Outstanding Paper Award in 2017, the IEEE ComSoc AP Outstanding Young Researcher Award in 2014, the Temasek Research Fellowship in 2013, and the Chun-Gang Outstanding Research Award in 2011. She also serves as the Chair for the IEEE Communication Society (ComSoc) Radio Communications Technical Committee (RCC). She has been serving as the Symposium/Track Chair for a number of international conferences, including 2020 IEEE GLOBECOM and 2020 IEEE WCNC. She was an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the IEEE COMMUNICATIONS LETTERS from 2014 to 2019. She is also an Editor of the IEEE WIRELESS COMMUNICATIONS LETTERS.

Sungjin Lee received the B.E. degree in electrical engineering from Korea University in 2005 and the M.S. and Ph.D. degrees in computer science and engineering from Seoul National University in 2007 and 2013, respectively. He is currently an Associate Professor with DGIST, South Korea. Before joining DGIST, he was a Post-Doctoral Associate with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. His current research interests include storage systems, operating systems, and system software.