Toward Trusted and Secure Communication Among Multiple Internal Modules in CAV

Hong Zhong, Wenwen Cao, Qingyang Zhang, Jing Zhang, *Graduate Student Member, IEEE*, and Jie Cui, *Member, IEEE*

***Abstract*—By equipping various sensors and analyzing sensed data, vehicles can perform automatic driving; these vehicles are known as connected and autonomous vehicles (CAVs). In CAVs, tampered data will result in incorrect driving decisions. Hence, secure data transmission should be ensured to enable correct life-critical decisions. Untrusted resource-constrained modules allow attackers to obtain private data from CAVs, such as the key. Benefitting from trusted computing, the proposed scheme can verify the trusted status of internal modules and achieve secure data transmission by adopting the remote attestation and hash message authentication code. The scheme is proven to be secure in the random oracle model under the computational Diffie–Hellman problem. Furthermore, we perform experiments and evaluate the performance using Intel Software Guard eXtensions, which provide part of the trusted computing function. The experimental results show that the scheme could be efficient and suitable for CAVs.**

***Index Terms*—Connected and autonomous vehicles (CAVs), remote attestation, trusted and secure communication, trusted computing.**

I. Introduction

TECHNOLOGIES, such as artificial intelligence and computer vision, have enabled the development of connected and autonomous vehicles (CAVs) [1]–[3]. CAVs can collect data pertaining to the surrounding environment and realize real-time decisions. The operation of CAVs does not require human involvement. As shown in Fig. 1, CAVs include at least three basic components. The sensor modules collect information around the vehicle, including light detection and ranging, radio detection and ranging, smart cameras, and GPS. The vehicle computing/communication unit (VCU) can analyze and process these data from sensor modules and realize timely decisions. The actuator modules are used to achieve quick responses according to the instructions of the VCU, such as braking or changing lanes, when encountering obstacles ahead [3]. To achieve autonomous driving, some security issues must be resolved. A CAV's decision is based on real-time collected data; these data are vulnerable to many attacks in communication, such as injection, deletion, modification, and replay [1], [4]. Therefore, CAV module communication security must be guaranteed.

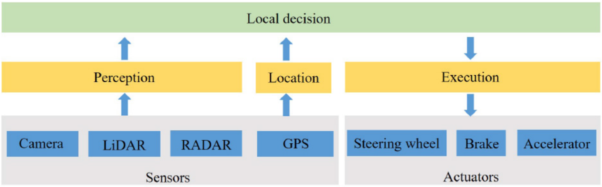


Fig. 1. Generic architecture of CAV.

In order to ensure secure communication in the message transmission process, message authentication is adopted [5]. To realize message authentication, many protocols have been proposed in vehicular *ad hoc* networks (VANETs) [6]–[8]. However, a CAV's real-time decision making is based on the rapid processing of sensor data. Therefore, the message authentication overhead should be minimized to ensure real-time processing. These protocols used for VANETs are not suitable for CAV internal modules. Compared with vehicles and roadside units in VANETs, the computing and storage capabilities of onboard modules are relatively weak. Meanwhile, high-cost authentication and key agreement protocols are suboptimal. Hence, efficient message authentication is necessary for the internal communication of CAVs.

Message authentication can ensure secure data transmission, but it cannot guarantee the reliability of the data generated in the module. For example, an untrusted module may provide unreliable data, thereby causing VCU to make incorrect decisions and trigger a traffic accident [4]. Untrusted modules may leak critical secret data, such as private keys. Trusted computing [9], [10] is a potential solution used to implement trusted modules. This ensures the module always behaves in the expected range and protects critical data. Remote attestation technology [11], [12] in trusted computing can provide trust measurement and reports. However, the current remote attestation incurs a relatively high computational overhead [13], and its performance may be affected by the restricted environment of the CAV. Moreover, if a participant is identified as malicious, the trusted authority must revoke it, which is extremely costly.

Manuscript received December 5, 2020; revised March 8, 2021 and April 27, 2021; accepted May 15, 2021. Date of publication May 21, 2021; date of current version December 7, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61872001, Grant 62011530046, and Grant U1936220; in part by the Special Fund for Key Program of Science and Technology of Anhui Province, China, under Grant 202003A05020043; and in part by the Open Fund of Key Laboratory of Embedded System and Service Computing (Tongji University), Ministry of Education under Grant ESSCKF2018-03. *(Corresponding author: Jie Cui.)*

The authors are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, the Anhui Engineering Laboratory of IoT Security Technologies, and the Institute of Physical Science and Information Technology, Anhui University, Hefei 230039, China (e-mail: cuijie@mail.ustc.edu.cn).

Digital Object Identifier 10.1109/JIOT.2021.3082542

A novel scheme of secure and reliable communication between CAV internal modules is proposed, which could guarantee the trust of modules and realize efficient message authentication. Ensuring the trust of the data sources is a prerequisite to realize message authentication. In the proposed scheme, every module is equipped with additional hardware, trusted platform module (TPM), which is a vital component of trusted computing for collecting status messages and protecting critical data. The modules could achieve mutual authentication based on remote attestation. After authentication, the session key agreement is completed among internal modules. Subsequently, the hash message authentication code (HMAC) can be used to achieve efficient message authentication for real-time data. The main contributions of this scheme are summarized as follows.

1) A novel secure communication framework based on trusted computing is proposed for CAV internal modules. The proposed scheme could achieve fast message authentication for VCU with the help of TPM, which can meet the requirements of CAV for efficient authentication of large amounts of real-time data. The security proof shows that the scheme is provably secure under the random oracle model, and security analysis shows that the proposed scheme can achieve the expected security objectives.
2) The proposed scheme could protect critical data and operations with the support of TPM. The key operations are completed by the TPM, and the nonsensitive operations with higher overhead are realized by the module. The prototype experiment was implemented, including part operations in the Software Guard eXtensions (SGXs) environment. The experimental analysis indicates that the proposed scheme could balance between usability and security.
3) The proposed scheme allows VCU to achieve efficient revocation with the help of TPM when it finds the compromised module. In addition, the proposed scheme could achieve efficient key updates by the backward hash chain.

Related works are described in Section II. Section III provides the preliminaries and system model of the proposed scheme. We briefly describe the scheme in Section IV. Then, the specific security analysis is shown in Section V. In Section VI, performance is evaluated through experiments and analyses. Finally, we provide some concluding remarks in Section VII.

II. Related Works

In this section, we first describe the typical remote attestation protocol and show how its current applications is. Then, this section introduces the typical message authentication protocol and its wide application in many fields.

A. Remote Attestation

The remote attestation protocol is the most widely accepted protocol used to implement platform attestation, and it is implemented through hardware called TPM. It could achieve authentication and privacy protection simultaneously. Camenisch–Lysyanskaya (CL) signature is a typical protocol in remote attestation protocol. The special feature of CL signature [14] is that it allows one to prove the knowledge of the signature in zero knowledge. The first remote attestation protocol was proposed by Brickell *et al.* [15]. This protocol first proposed a new scheme that the attestor achieves the platform authentication for the verifier. Brickell *et al.* deployed TPM to protect critical data and achieve important operations.

However, this protocol proposed by Brickell *et al.* is based on RSA and it is not efficient enough. In 2008, Brickell *et al.* [16] proposed a new scheme to reduce the computational overhead, which is based on the bilinear map, the decisional bilinear Diffie–Hellman assumption and the computational Diffie–Hellman (CDH) problem. Although the computational overhead is relatively low, it is still expensive. Chen *et al.* proposed a new protocol [17] by adopting a property-based attestation (PBA) to accomplish authentication. It could measure the configuration message and collect the integrity state of the attestor through TPM registers. However, the overmuch computation for checking revocation adds extra time. Feng and Qin proposed a new attestation protocol [18] to reduce the length of signature and the overhead of checking revocation, but computational overhead is still expensive.

Due to the characteristics of remote attestation protocol, it has been widely used in many fields. Amelino *et al.* proposed a remote anonymous activation protocol to protect the intellectual property (IP) license [19], and TPM is adopted in the device to protect IP, and the validity period of the IP can be increased or decreased through the hash chain. Yang *et al.* [20] proposed a scheme based on remote attestation to achieve authentication for the vehicles across different trusted domains. After the session key agreement between the participants of different domains is realized, the mutual authentication is realized by remote attestation based on proxy signature. It makes remote attestation suitable for different network scenarios. Yang *et al.* built the trusted cloud computing platform for tenants [21], and remote attestation could protect key operations, such as hash verification and trusted chain measurement. Zhao *et al.* developed the attribute certificate scheme [13] for achieving secure data sharing of smart meters by combining ring signature, and remote attestation could hide the platform configuration and achieve the trusted detection. Chen *et al.* achieved mutual authentication in network-connected unmanned aerial vehicles [22], and it reduced the computational workload of TPM and Host.

B. Message Authentication

In order to achieve secure communication between multiple devices, some protocols based on public key infrastructure (PKI) have been proposed. Wang [23] proposed a scheme based on group signature and secret sharing to achieve authentication for Internet of Things end devices. In order to avoid the attacker obtaining data by unauthorized way, Zeng *et al.* [24] proposed the dual authentication protocol for CAV and roadside units. Liu *et al.* [25] proposed a distributed proxy-based authentication scheme, which uses distributed

TABLE I
PROS AND CONS OF VARIOUS SCHEMES

Schemes	Main technologies adopted	Advantages	Disadvantages
[6]	Pseudonyms-based signature, Hash chain	Privacy perservation, Batch authentication	CRL storage overhead
[7]	Group Signature	Privacy preservation authentication, Batch authentication	Fail to achieve the trust measurement
[8]	DAA, PKI-based signature	TPM enhance privacy, Revocation without CRL	Fail to achieve batch authentication

computing to achieve message authentication for a large number of vehicles. In order to realize message authentication of fast moving vehicles, Cui *et al.* [26] proposed a reliable and efficient content-sharing scheme based on batch authentication. In order to realize collaborative data sharing of multiple vehicles, Cui *et al.* [27] proposed a secure and efficient data-sharing scheme based on edge computing and agent vehicles, and this scheme was based on binary search, which could find false information efficiently. These PKI-based schemes can achieve message authentication to ensure secure transmission. However, there is an obvious flaw that these schemes are not efficient for real-time communications.

Message authentication code (MAC) is a solution to achieve fast message authentication. Messages are transmitted over a public channel. Thus, HMAC could provide a method to check the message integrity and it could be used for achieving message authentication in many fields. In [28], HMAC was used to ensure vehicle secure communication, and HMAC is used to reduce pairing operations. Ashritha and Sridhar [29] used HMAC in combination with pseudo-id and timestamp to achieve efficient authentication between RSU and vehicles. Jiang *et al.* [6] replaced the certificate revocation list by computing HMAC to achieve efficient authentication for the vehicle and RSU. Khemissa and Tandjaoui [30] adopted HMAC, random numbers, and exclusive-or operations to check the message integrity after the session key agreement phase. Halabi *et al.* [31] proposed a lightweight protocol for sensor nodes by the hash operation and HMAC, and HMAC was used for achieving lightweight authentication for WSN [32]. Chim *et al.* [33] achieved the authentication process through HMAC and pseudo-id for the smart grid. Mahmood *et al.* [34] proposed a scheme based on Diffie–Hellman key agreement, in which HMAC was used to ensure the message integrity of the smart grid.

Some of the pros and cons for some message authentication and DAA scheme are listed in Table I.

III. PRELIMINARIES AND BACKGROUND

In this section, we introduce the preliminaries of the proposed scheme, and we describe the system model in detail. Then, the security requirements are proposed.

A. Preliminaries

1) Bilinear Maps: Let G_1 and G_T be cyclic groups of prime order q, and g_1 and g_T are generators of G_1 and G_T. Let $e : G_1 * G_1 = G_T$ be an efficient map to satisfy the following property.

1) *Bilinearity:* For all g_1, $g_2 \in G_1$, and a, $b \in Z_q$, there exist $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
2) *Nondegeneracy:* It is not a trivial map. There exists g_1, $g_2 \in G$ that $e(g_1, g_2) \neq 1$.
3) *Computability:* There is an efficient algorithm to compute $e(g_1, g_2)$.

Definition 1: Given the three tuple (x, xP, yP), where P is a generator of G, and x and y are random numbers from Z_q^*, $Q = xP$. It has been proved that calculating xyP from Q is difficult.

Definition 2: Assuming that the algorithm A solves the CDH problem in group G within polynomial time, the probability of success is defined as

$$\mathrm{Succ}_{A,G}^{\mathrm{CDH}} = \Pr\Big[A(P, xP) = x : x \in Z_q^*\Big] \geq \varepsilon.$$

Then, algorithm A solves the CDH hypothesis is negligible in any polynomial time.

2) Property-Based Attestation: Remote attestation is a technique to solve the measurement and verify whether others are credible, and is one of the keys to trusted computing. The remote attestation technology could allow one entity called the *verifier* to verify another entity called the *attestor*. The verifier sends a challenge to the attestor. Then, the attestor receives this challenge and invokes TPM to generate attestation response, and sends the result to the verifier for verification. Then, the verifier checks the response, and this process succeeds if this response matches the desired result.

PBA is a kind of hardware-based remote attestation that achieves attestation through the collected property. These registers *PCR* of TPM could collect the configuration property message during the boot process. The value could extend through the hash chain $PCR_{i+1} = H(PCR_i||\text{input})$, and the whole property message of the boot process could be represented as $cs = (PCR_0, PCR_1, \ldots, PCR_n)$ [15]. CL signature is the basis of the PBA scheme. The main process of CL signature includes system initialization, signature generation, and signature verification. The CL signature scheme is generated as follows [14].

1) The issuer chooses $x, y, z \in Z_q$, and computes $X = g^x$, $Y = g^y$, and $Z = g^z$. Let G_1 and G_T be cyclic groups of prime order q, and g_1 and g_T be generators of G_1 and G_T, and let e:$G_1 * G_1 = G_T$. Set secret key $sk = (x, y, z)$ and public key $pk = (G, G_T, g, g_T, X, Y, Z)$.
2) During the signature phase, the attestor inputs (m, r), along with pk and sk, it chooses $a \in G$, and computes $A = a^z, b = a^y, B = A^y, c = a^{x+mxy}A^{xyr}$, and the output signature as $\sigma = (a, A, b, B, c)$.
3) During the verification phase, the verifier gets $\sigma = (a, A, b, B, c)$, pk and (m, r), the verifier just verifies whether these equations $e(a, Z) = e(g, A)$, $(a, Y) = e(g, B)$, $e(A, Y\ ) = e(g, B)$, $e(X, a) \cdot e(X, b)^m \cdot e(X, B)^r = e(g, c)$ are satisfied or not.

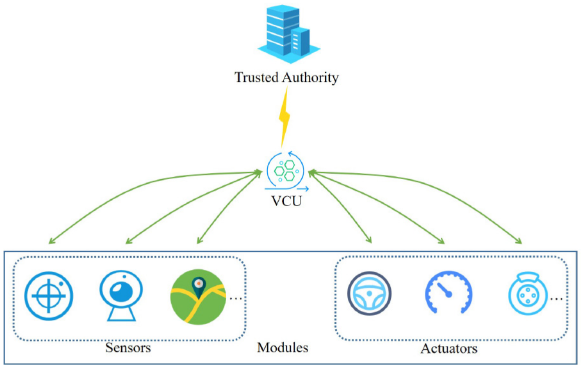


Fig. 2. Framework of the proposed scheme.

CL signature is based on the CDH hard problem, and the CDH problem is shown as follows. Assuming that G is a cyclic group with generator g, choose $x, y \in Z_q$, and compute g^x and g^y. Given g, g^x, and g^y, no efficient adversary in the polynomial time can output g^{xy}. Thus, the CDH problem is considered to be difficult.

3) Keyed-Hash Message Authentication Code: HMAC is a specific type of MAC that could be used for verifying the integrity and authenticity of a message. HMAC contains hash function and secret encryption key. If participants A and B need to establish a secure communication, first A and B need to negotiate a secure session key. A sends a message and HMAC, and the receiver B can use the key and message to calculate a new HMAC and compare it with the received HMAC.

B. System Model

In some schemes, such as Baidu's Apollo, generated data can be processed on the vehicle. In the proposed scheme, all work is also done in CAVs. This mechanism could process data and provide real-time services while avoiding the disclosure of sensitive information. Fig. 2 illustrates the system model of this scheme. The entities include TA, module i, and VCU. The module i represents the sensors and actuators inside the vehicle, such as LiDAR and RADAR. The module t could process raw data and VCU could make real-time decision. Therefore, the module i and VCU are equipped with the TPM to protect critical operations. The module with TPM can achieve attestation, so as to verify the trust of the device. TPM is responsible for collecting configuration property message through registers. The key operations are completed by the TPM, and the nondata-sensitive operations with large overhead are implemented by the module. The TPM$_t$ only communicates to module t. The communication security between TPM$_t$ and module t is guaranteed by a physical method.

TA: TA is a widely reliable and secure entity, and TA has computational and storage capabilities [35]. TA is fully trusted in our assumption and responsible for generating the system parameters. In the proposed scheme, the TA can generate a certificate for every module. Every module could achieve authentication with each other based on the certificate. Furthermore, TA is the only entity that could reveal the identity and the corresponding certificate for every module. When one module is compromised, the TA could track that module and report it to the user.

Modules: Modules represent the sensor modules and vehicle actuators. In CAVs, the sensors can collect real-time road condition information and information around the vehicle. The sensor modules include the smart camera, LiDAR, RADAR, and GPS. The vehicle actuators are able to accept real-time commands and respond positively. Actuators include accelerator, brake, and steering wheel.

VCU: VCU represents the heterogeneous onboard vehicle computing/communication unit. During the certificate request phase, VCU is responsible for transmitting data between the TA and module i. During the phase of generating real-time data, the VCU is responsible for processing the data received by the sensor module and making optimal decisions. It sends a core command to vehicle actuators. If the module i is compromised, the VCU could report to the TA.

C. Assumptions

Like other authentication schemes in the literature, the proposed scheme is based on the following assumptions.

1) The trusted authority is assumed to be secure and fully trusted. Assume that there is a physical binding between the TPM and the module. The TPM is physical security and that the TPM will be damaged if a violent attack occurs. In the proposed scheme, it is assumed that the communication between TPM and module is secure and reliable.
2) Unlike TPM, vehicle internal modules are more susceptible to interference and attack. For example, there are attacks against sensor modules. It is assumed that these modules are in the range of receiving messages and can all receive messages. This scheme does not consider silent modules, in which case VCU can notify the user to implement replacement.
3) The proposed scheme only considers verifying the trust and protecting critical data of the internal modules. The SGX in the experiment could protect critical data and perform critical operations by creating enclaves. The scheme does not consider the other security issues, such as control-flow hijacking attacks and time-of-use-time-of-check attacks.

Since Denial-of-Service (DoS) attacks are almost impossible to resist completely, like other schemes [36], [37], the DoS attacks are beyond our scope.

Also, without loss of generality, we assume that the TA located in the control center has higher computing power, and the VCU is responsible for data processing, and its computing power is better than other modules (such as sensors).

D. Attack Model

The security of data transmission is the focus of the scheme, but there are two kinds of attackers: 1) external attackers and 2) internal attackers. External attackers have more computing power, but they can only get information from messages transmitted on the public channel. The internal attacker may be a malicious module that can access confidential information, so they can also be extremely destructive. Attacks often disrupt the normal process of data transmission and prevent participants from receiving the correct data. Generally, the attack types include adaptive-chosen-message attacks, selective attacks, and common attacks, such as impersonation attacks, modification attacks, and replay attacks.

1) *Impersonation Attacks:* The attacker wants to pretend to be a legitimate participant to deceive the verifier.
2) *Modification Attacks:* The attacker can modify the valid message and send it to the receiver.
3) *Replay Attacks:* The attacker resends the previously obtained legal signature to the receiver to pass verification.

Definition 3: The proposed scheme includes three steps, including setup, signature, and verification. In this section, we define the steps for the scheme under the random oracle model. These settings are defined as follows.

1) *Setup(1^k)*: Given a random system security parameter k, TA will output public system parameter parameters pk_{TA}, and system user key sk_t, pk_t.
2) *Sign(ID_i, sk_i, m):* Given the parameters of the system, the signer's secret key sk_i, the identity ID_i, and certificate of the signer, it will output the corresponding PBA signature.
3) *Verify(ID_v, pk_i, m, σ_i):* Given the parameters of the system, the user's public key, identity, signature σ_i, and message m. If σ_i is a valid signature, then it outputs 1; otherwise, outputs 0.

Definition 4: The authentication scheme is secure if the possibility of an adversary $\mathcal{A}$ breaking it can be ignored in any polynomial time. Under adaptive-chosen-message attacks and selective attacks, the signature algorithm is secure to prevent forgery.

Game: According to the adversary's ability, the security model of the proposed scheme is defined through the game between the adversary $\mathcal{A}$ and challenger B. The game between the adversary $\mathcal{A}$ and challenger B is defined as follows.

Setup: Challenger B runs the setup step to obtain the system parameters and system public key, and then sends them to $\mathcal{A}$.

Query: The adversary $\mathcal{A}$ asks the following questions to challenger B.

1) *Hash Query:* The adversary $\mathcal{A}$ requests a hash function, challenger B returns the corresponding hash value, and stores the hash value.
2) *Attest Query:* Adversary $\mathcal{A}$ requests the signature of selected message m. Then, challenger B returns σ_i to $\mathcal{A}$.

Output: When adversary $\mathcal{A}$ believes that the entire process has been completed, $\mathcal{A}$ will return a valid signature. If so, the signature will be accepted and the signature has not yet been requested. After querying, adversary $\mathcal{A}$ is expected to win the game.

E. Security Objectives

In order to achieve trusted and secure communication between internal modules, the scheme needs to meet the following security objectives.

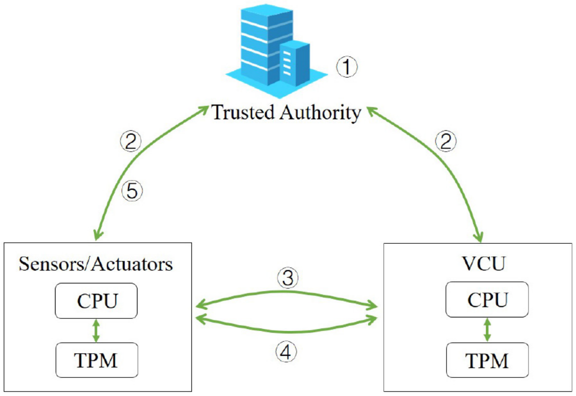


Fig. 3. Interaction model of the proposed scheme.

1) Trust Measurement: Before sending and receiving the data, the VCU and module i must ensure the reliability of each other. The trust of the module is the basis of the internal security communication of CAV.

2) Message Authentication: If the integrity of the message is not guaranteed, an attacker can launch some attacks, such as deleting, modifying, and replaying. Thus, when a module i or the VCU receives one message, it must implement message authentication.

3) Privacy Protection: If the attacker could get the configuration data of one module, it could guess the parameters of this module based on the same type of module. In order to protect the security of the module, it is necessary to realize the privacy protection of the configuration.

4) Resistance to Ordinary Attacks: This scheme should be able to withstand the typical attacks, such as impersonation, modification, and replay to ensure the CAV's internal secure communication.

IV. Proposed Scheme

We give a detailed description of the proposed scheme in this section. This scheme consists of three parts. Fig. 3 demonstrates the interactive model of the entire protocol.

1) TA generates system parameters, and every TPM generates a private key and public key, respectively.
2) Then, module i and the VCU invoke TPM$_i$ and TPM$_v$ to collect the configuration message, respectively. Module i and the VCU send this message to the TA to request the certificate through a secure manner like encryption. Then, the TA issues a certificate for every module. After receiving the certificate, module i and VCU verify and send a blind certificate.
3) Module i and the VCU generate and verify the PBA signature to achieve mutual authentication. After verification, module i and the VCU establish the session key.
4) The module and the VCU could use the generated key to generate the HMAC value for verifying message integrity.
5) The module i is compromised, and TA and TPM$_i$ could revocate the identity of the module. Table II lists the notations in the proposed scheme. To simplify the

TABLE II
NOTATIONS

Notations	Definitions
TA	Trusted authority
module i	Sensor modules or vehicle actuators
VCU	Vehicle Computing/Communication Unit
TPM_t	Trusted Platform Module t
sk_t	The secret key of entity t
pk_t	The public key of entity t
ID_t	The real identity of every entity t
cs_t	Configuration message of TPM_t
ps	The evaluated property for configuration message
σ_t	Certificate for entity t
m, n	The secret random value
δ_t	The signature generated by TPM_t
σ_{PBAt}	PBA signature by module t
k_{vi}	The root key saved in TPM_i and TPM_v
H, h	The Collision-avoid one-way hash operation
m_d	The Real-time message generated by module i
k_d	Session key generated by TPM_i and TPM_v
$\|\|$	Concatenation operation

description in Table II, we introduce the entity t to represent the internal modules, including the VCU and module i.

A. System Initialization

In this phase, the participants are the TA, module i (with TPM$_i$), and VCU (with TPM$_v$). The TA chooses two secure hash functions, $h : \{0, 1\}^* \rightarrow \{0, 1\}^l$, $H : \{0, 1\}^* \rightarrow \{0, 1\}^l$. The TA chooses two groups $G =< g >, G_T =< g_T >$ of prime order q and a bilinear map $e : G_1 * G_1 = G_T$. Then, the TA chooses the random number $x, y, z \in Z_q$ and lets the secret key be $sk_{TA} = (x, y, z)$, so that X, Y, and Z are computed as $X = g^x, Y = g^y, Z = g^z$. The TA publishes the public key $pk_{TA} = (g, g_T, G, G_T, X, Y, Z)$ to all CAVs. TPM$_i$ and TPM$_v$ generate their unique keys (sk_i, pk_i) and (sk_v, pk_v) based on attestation identity key (AIK). sk_i and sk_v could be protected by TPM$_i$ and TPM$_v$, respectively, and TPM$_i$ and TPM$_v$ publish their pk_i and pk_v, respectively.

B. Certificate Generation and Verification

1) Message Collection and Certificate Request: TPM$_i$ collects its secret message configuration specification, denoted by $cs_i = (PCR_{i0}, PCR_{i1}, \ldots, PCR_{in})$ [38] and sends cs_i to module i. Similarly, TPM$_v$ collects the configuration message $cs_v = (PCR_{v0}, PCR_{v1}, \ldots, PCR_{vn})$ and sends it to the VCU. Module i gets cs_i, and then computes $\text{Enc}_{pk_{TA}}(ID_i, cs_i||pk_i)$ and sends it to TA to request the property certificate. The VCU gets cs_v, and computes and sends $\text{Enc}_{pk_{TA}}(ID_v, cs_v||pk_v)$ to TA in the same manner.

2) Verification and Certificate Generation: The TA could get cs_i and cs_v by using sk_{TA} to decrypt $\text{Enc}_{pk_{TA}}(ID_i, cs_i||pk_i)$ and $\text{Enc}_{pk_{TA}}(ID_v, cs_v||pk_v)$. The TA has the evaluated property ps [39], [40], so the TA evaluates the configuration information based on ps. If cs_i or cs_v meet with ps, the TA issues the property certificate for module i and the VCU.

TA generates the signature related to cs_i. First, TA chooses the random number $a_i \in G$, then computes $A_i = a_i^z$, $b_i = a_i^y$, $B_i = A_i^y$, $c_i = a_i^{x+xycs_i} A_i^{xyps}$, and sends the certificate $\sigma_i = (a_i, A_i, b_i, B_i, c_i)$ to module i. Similarly, VCU gets the certificate $\sigma_v = (a_v, A_v, b_v, B_v, c_v)$= $a_v, a_v^z, a_v^y, A_v^y, a_v^{x+xycs_v} A_v^{xyps} (a_v \in G)$. Thus, module i and VCU get the certificate, respectively.

3) Certificate Verification: When the module i receives certificate σ_i from the TA, module i first checks the validity of the certificate through computing $e(a, Z_i) \overset{?}{=} e(g, A_i)$, $e(X, a_i) e(X, b_i)^{cs_i} e(X, B_i)^{ps} \overset{?}{=} e(g, c)$, $e(A, Y_i) \overset{?}{=} e(g, B_i)$. The VCU verifies the certificate σ_v in the same way.

Then, module i and the VCU bind the certificate like [8]. TPM$_i$ first chooses the random number $r_{i_1}, r_{i_2} \in Z_q$, computes $r_{i_1}^{-1}$, and sends $r_{i_1}^{-1}, r_{i_2}$ to module i. Module i computes $a_i' = a^{ri2}, b_i' = b^{ri2}, A_i' = A^{ri2}, B_i' = B^{ri2}, c_i' = c^{ri2 r_{i1}^{-1}}$. $\sigma_i' = (a_i', A_i', b_i', B_i', c_i')$ as the final bind certificate for module i. TPM$_v$ also chooses random number $r_{v_1}, r_{v_2} \in Z_q$, computes $r_{v_1}^{-1}$, and sends $r_{v_1}^{-1}, r_{v_2}$ to the VCU. The VCU computes $a_v' = a^{rv2}, b_v' = b^{rv2}, A_v' = A^{rv2}, B_v' = B^{rv2}, c_v' = c^{rv2 r_{v1}^{-1}}$, and $\sigma_v' = (a_v', A_v', b_v', B_v', c_v')$ as the bind certificate for VCU.

C. Mutual Authentication and Session Key Establish

1) PBA Signature Generation: As shown in Fig. 4, for mutual authentication, First VCU requests the PBA signature of module i. VCU invokes TPM$_v$ to choose two values $N_v, n \in Z_q$, and TPM$_v$ computes g^n and then sends N_v, g^n to module i through VCU. After receiving this request, module i invokes TPM$_i$ to collect the configuration property cs_i.

Step 1: Module i and VCU compute and output the PBA signature. TPM$_i$ computes $\delta_i = \text{Sign}_{sk_i}(N_v, g^n)$ and sends signature δ_i to module i to generate PBA signature further. Module i computes $u_{i_x} = e(X, a_i')$, $u_{i_{xy}} = e(X, b'_i)$, $u_{i_s} = e(g, c_i')$, and $u_{i_{xyz}} = e(X, B_i')$. These parameters $u_{i_x}, u_{i_{xy}}, u_{i_{xyz}}$, and u_{i_s} are a part of the PBA signature, and they could be calculated while offline. Meanwhile, TPM$_i$ chooses random number $w_{i_1}, w_{i_2} \in Z_q$ and sends to module i. Module i computes $T_i = u_{i_s}^{w_{i_2}} (u_{i_x} u_{i_{xy}}^{w_{i_1}} u_{i_{xyz}}^{ps})^{-1}$, and

$$c_{H_i} = H\big(\sigma'_i, u_{i_x}, u_{i_{xy}}, u_{i_{xyz}}, u_{i_s}, T_i, N_v, g^n\big). \tag{1}$$

Module i computes $s_{i1} = w_{i_1} - c_{H_i} * cs_i \text{mod} q$ and $s_{i2} = w_{i_2} - c_{H_i} * r_{i_1} \text{mod} q$. Meanwhile, TPM$_i$ generates two random number $N_i, m \in Z_q$, and TPM$_i$ computes g^m and then sends N_i, g^m to VCU through module i. Finally, TPM$_i$ and module i send PBA signature along with N_i, g^m. Signature $\sigma_{\text{PBA}i}$ is represented as

$$\sigma_{\text{PBA}i} = \big(\sigma'_i, c_{H_i}, \delta_i, s_{i1}, s_{i2}, T_i, N_i\big). \tag{2}$$

Step 2: When the VCU receives N_i, g^m and $\sigma_{\text{PBA}i}$, it generates the PBA signature as follows. VCU first sends N_i, g^m to TPM$_v$ to compute $\delta_j = \text{Sign}_{sk_v}(N_i, g^m)$, and TPM$_v$ sends δ_j to the VCU. Then, the VCU computes $u_{v_x} = e(X, a_v')$, $u_{v_{xy}} = e(X, b'_v)$,$u_{v_s} = e(g, c_v')$, and $u_{v_{xyz}} = e(X, B_v')$. The VCU computes

$$c_{H_v} = H\big(\sigma'_v, u_{v_x}, u_{v_{xy}}, u_{v_{xyz}}, u_{v_s}, T_v, N_i, g^m\big). \tag{3}$$

TPM$_v$ chooses random number $w_{v_1}, w_{v_2} \in Z_q$ and sends it to the VCU. Then, the VCU computes $s_{v1} = w_{v1} - c_{H_v} cs_v \text{mod} q$ and $s_{v2} = w_{v2} - c_{H_v} r_{v_1} \text{mod} q$, and VCU computes $T_v =$

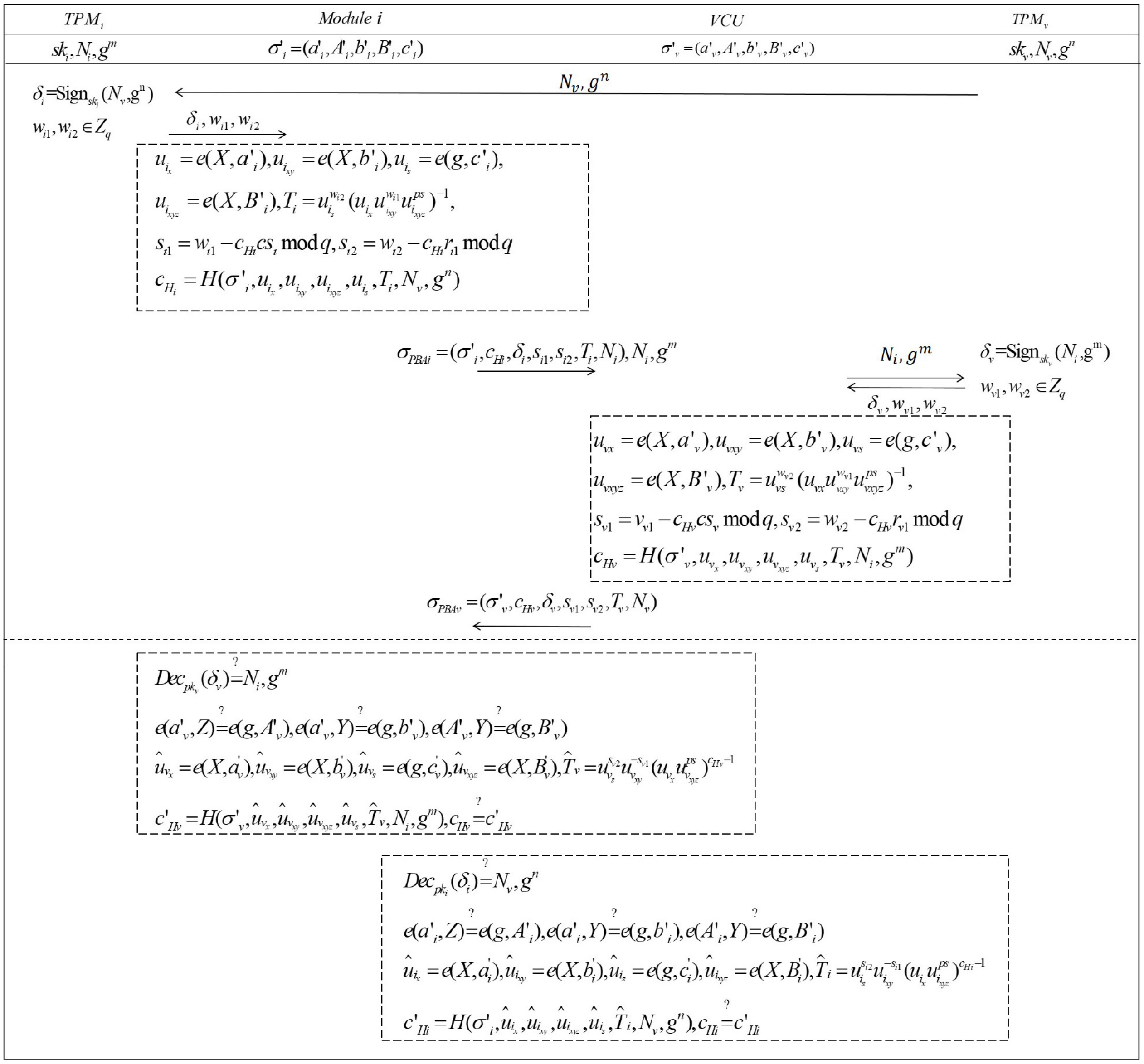

TPM_i
Module i
VCU
TPM_v
sk_i, N_i, g^m
$\sigma'_i = (a'_i, A'_i, b'_i, B'_i, c'_i)$
$\sigma'_v = (a'_v, A'_v, b'_v, B'_v, c'_v)$
sk_v, N_v, g^n
N_v, g^n
$\delta_i = \mathrm{Sign}_{sk_i}(N_v, g^n)$
$w_{i1}, w_{i2} \in Z_q$
δ_i, w_{i1}, w_{i2}
$u_{i_x} = e(X, a'_i), u_{i_{xy}} = e(X, b'_i), u_{i_s} = e(g, c'_i),$
$u_{i_{xyz}} = e(X, B'_i), T_i = u_{i_s}^{w_{i2}}(u_{i_x} u_{i_{xy}}^{w_{i1}} u_{i_{xyz}}^{ps})^{-1},$
$s_{i1} = w_{i1} - c_{Hi} cs_i \bmod q, s_{i2} = w_{i2} - c_{Hi} r_{i1} \bmod q$
$c_{H_i} = H(\sigma'_i, u_{i_x}, u_{i_{xy}}, u_{i_{xyz}}, u_{i_s}, T_i, N_v, g^n)$
$\sigma_{PBAi} = (\sigma'_i, c_{Hi}, \delta_i, s_{i1}, s_{i2}, T_i, N_i), N_i, g^m$
N_i, g^m
$\delta_v = \mathrm{Sign}_{sk_v}(N_i, g^m)$
$w_{v1}, w_{v2} \in Z_q$
δ_v, w_{v1}, w_{v2}
$u_{vx} = e(X, a'_v), u_{vxy} = e(X, b'_v), u_{vs} = e(g, c'_v),$
$u_{vxyz} = e(X, B'_v), T_v = u_{vs}^{w_{v2}}(u_{vx} u_{vxy}^{w_{v1}} u_{vxyz}^{ps})^{-1},$
$s_{v1} = v_{v1} - c_{Hv} cs_v \bmod q, s_{v2} = w_{v2} - c_{Hv} r_{v1} \bmod q$
$c_{Hv} = H(\sigma'_v, u_{v_x}, u_{v_{xy}}, u_{v_{xyz}}, u_{v_s}, T_v, N_i, g^m)$
$\sigma_{PBAv} = (\sigma'_v, c_{Hv}, \delta_v, s_{v1}, s_{v2}, T_v, N_v)$
$Dec_{pk_v}(\delta_v) \overset{?}{=} N_i, g^m$
$e(a'_v, Z) \overset{?}{=} e(g, A'_v), e(a'_v, Y) \overset{?}{=} e(g, b'_v), e(A'_v, Y) \overset{?}{=} e(g, B'_v)$
$\hat{u}_{v_x} = e(X, a'_v), \hat{u}_{v_{xy}} = e(X, b'_v), \hat{u}_{v_s} = e(g, c'_v), \hat{u}_{v_{xyz}} = e(X, B'_v), \hat{T}_v = u_{v_s}^{s_{v2}} u_{v_{xy}}^{-s_{v1}} (u_{v_x} u_{v_{xyz}}^{ps})^{c_{Hv}-1}$
$c'_{Hv} = H(\sigma'_v, \hat{u}_{v_x}, \hat{u}_{v_{xy}}, \hat{u}_{v_{xyz}}, \hat{u}_{v_s}, \hat{T}_v, N_i, g^m), c_{Hv} \overset{?}{=} c'_{Hv}$
$Dec_{pk_i}(\delta_i) \overset{?}{=} N_v, g^n$
$e(a'_i, Z) \overset{?}{=} e(g, A'_i), e(a'_i, Y) \overset{?}{=} e(g, b'_i), e(A'_i, Y) \overset{?}{=} e(g, B'_i)$
$\hat{u}_{i_x} = e(X, a'_i), \hat{u}_{i_{xy}} = e(X, b'_i), \hat{u}_{i_s} = e(g, c'_i), \hat{u}_{i_{xyz}} = e(X, B'_i), \hat{T}_i = u_{i_s}^{s_{i2}} u_{i_{xy}}^{-s_{i1}} (u_{i_x} u_{i_{xyz}}^{ps})^{c_{Hi}-1}$
$c'_{Hi} = H(\sigma'_i, \hat{u}_{i_x}, \hat{u}_{i_{xy}}, \hat{u}_{i_{xyz}}, \hat{u}_{i_s}, \hat{T}_i, N_v, g^n), c_{Hi} \overset{?}{=} c'_{Hi}$

Fig. 4. PBA signature generation and verification phase.

$u_{v_s}^{w_{v2}}(u_{v_x} u_{v_{xy}}^{w_{v1}} u_{v_{xyz}}^{ps})^{-1}$. Finally, the VCU generates and sends the PBA signature as

$$\sigma_{\mathrm{PBA}v} = \left(\sigma'_v, c_{H_v}, \delta_v, s_{v1}, s_{v2}, T_v, N_v\right). \tag{4}$$

2) PBA Signature Verification: When module i receives signature σ_{PBAv}, module i must verify the PBA signature to verify the VCU, which then needs to complete the same verification process.

First, module i uses public key pk_v to verify δ_v by computing $\mathrm{Dec}_{pk_v}(\delta_v) \overset{?}{=} N_i, g^m$. If it is satisfied, then continue, else reject and abort.

Module i could verify $\sigma'_v = (a'_v, A'_v, b'_v, B'_v, c'_v)$ by computing $e(a'_v, Z) \overset{?}{=} e(g, A'_v)$, $e(a'_v, Y) \overset{?}{=} e(g, b'_v)$, and $e(A'_v, Y) \overset{?}{=} e(g, B'_v)$.

Module i verifies the signature by these received data. It could compute $\widehat{u}_{v_x} = e(X, a'_v))$, $\widehat{u}_{v_{xy}} = e(X, b'_v)$, $\widehat{u}_{v_s} = e(g, c'_v)$, $\widehat{u}_{v_{xyz}} = e(X, B'_v)$, and $\widehat{T}_v = u_{vs}^{s_{v2}} u_{v_{xy}}^{-s_{v1}} (u_{v_x} u_{v_{xyz}}^{ps})^{c_{Hv}-1}$. It computes

$$c'_{H_v} = H\left(\sigma'_v, \widehat{u}_{v_x}, \widehat{u}_{vxy}, \widehat{u}_{v_{xyz}}, \widehat{u}_{v_s}, \widehat{T}_v, N_i, g^m\right). \tag{5}$$

If (3) is equal to (5) is satisfied, it turns out that the PBA signature of the VCU is verified by module i.

At the same time, the VCU verifies the PBA signature of module i. First, the VCU verifies δ_i. The VCU uses pk_i to decrypt δ_i and checks $\mathrm{Dec}_{pk_i}(\delta_i) \overset{?}{=} N_v, g^n$. If it passes verification, then continue, else reject and abort.

The VCU verifies $\sigma_i = (a_i, A_i, b_i, B_i, c_i)$ in the same manner. It could verify $e(a'_i, Z) \overset{?}{=} e(g, A'_i)$, $e(a'_i, Y) \overset{?}{=} e(g, b'_i)$, and $e(A'_i, Y) = e(g, B'_i)$.

The VCU verifies the PBA signature from module i in the same manner. It could compute through $\widehat{u}_{i_x} = e(X, a'_i)$, $\widehat{u}_{i_{xy}} = e(X, b'_i)$, $\widehat{u}_{i_s} = e(g, c'_i)$, $\widehat{u}_{i_{xyz}} = e(X, B'_i)$, and $\widehat{T}_i = u_{i_s}^{s_{i2}} u_{i_{xy}}^{-s_{i1}} (u_{i_x} u_{i_{xyz}}^{ps})^{c_{Hi}-1}$. The VCU computes

$$c'_{H_i} = H\left(\sigma'_i, \widehat{u}_{i_x}, \widehat{u}_{i_{xy}}, \widehat{u}_{i_{xyz}}, \widehat{u}_{i_s}, \widehat{T}_i, N_v, g^n\right). \tag{6}$$

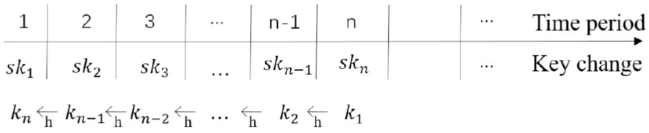

1
2
3
···
n-1
n
···
Time period
sk_1
sk_2
sk_3
...
sk_{n-1}
sk_n
···
Key change
$k_n \xleftarrow{h} k_{n-1} \xleftarrow{h} k_{n-2} \xleftarrow{h} \cdots \xleftarrow{h} k_2 \xleftarrow{h} k_1$

Fig. 5. Backward hash chain to generate the key.

Then, the VCU checks whether (1) is equal to (6), if the equation does not hold, reject and abort.

After verifying the PBA signature, module i and the VCU achieve mutual authentication. Though the computation overhead is relatively high, these phases *A–F* could be triggered periodically as the CAV launches.

3) Session Key Establish: Through the Diffie–Hellman key exchange protocol, TPM$_i$ receives g^n and has m, so it could compute $k_{vi} = (g^n)^m$. TPM$_v$ receives g^m and has n, and it computes $k_{vi} = (g^m)^n$. Thus, the root key is established after the mutual authentication phase. The sets of session keys are different for these modules.

The root key $k_{vi} = g^{mn}$ can be done by TPM$_i$ and TPM$_v$. As shown in Fig. 5, we could adopt the root key k_{vi} to generate a set of keys by using backward hash chain. It could generate d keys by computing $k_{i+1} = h(k_i)$ and $k_1 = h(k_{vi})$.

D. Message Authentication

We adopt the HMAC to verify the message between module i and the VCU, when module i sends the message $m_1, m_2, \ldots, m_d$, and TPM$_i$ can send $k_d, k_{d-1}, \ldots, k_1$ to module i. Then, module i computes HMAC$_{k_{n+1-i}}(m_i) i \in (1, d)$ and sends m_i, HMAC$_{k_{n+1-i}}(m_i)$ to the VCU. Note that the i is the index of the message. TPM$_v$ could generate $\widehat{k}_{n+1-i}$ through the index i. Then, the VCU receives m_i from module i and gets $\widehat{k}_{n+1-i}$ from TPM$_v$. VCU attends to compute HMAC$_{\widehat{k}_{n+1-i}}(m_i)$. The VCU could verify the message by computing

$$\mathrm{HMAC}_{k_{n+1-i}}(m_i) \overset{?}{=} \mathrm{HMAC}_{\widehat{k}_{n+1-i}}(m_i).$$

E. Efficient Revocation

When the VCU finds out module i is sending error message beyond its expectations, the VCU saves the record locally. When the record reaches a threshold, the VCU sends $Enc_{pk_{TA}}(ID_v, m)$ to the TA. TA supports the compromised module check and reporting, which could make the user know the compromised module and take the next action, such as replacing the compromised module or repairing it. m means the report message. Then, the TA verifies the message and finds the certificate of module i. Finally, the TA sends a revocation command to TPM$_i$ to delete the configuration message cs_i and certificate $(a_i, A_i, b_i, B_i, c_i)$. Since every PBA signature must include one unique blind certificate, one module cannot generate the PBA signature without TPM. The TA notifies TPM$_i$ that module i is compromised. The compromised module may ignore the revocation command, so the TA sends the message regularly to TPM$_i$ to ensure that the TPM can receive real-time commands and respond accordingly, which is called the "heartbeat mechanism" [8], [41]. The message is either revocation notice or a timestamp message.

Note that before the key k_i is exhausted, the hash chain could be extended synchronously by the typical algorithm between TPM$_i$ and TPM$_v$.

V. Security Analysis

In this section, we achieve the formal security proof for the PBA signature in the random oracle model. Then, based on the security requirements proposed above, we analyze the security of message authentication in detail.

A. Formal Security Proof

The proposed scheme provides the unforgeability of the PBA signature during the mutual attestation phase under the random oracle model. In this section, we demonstrate that the proposed scheme could resist the adaptive chosen message attack. If adversary $\mathcal{A}$ can forge the PBA signature with a nonnegligible probability, there exists an algorithm $\mathcal{B}$ that could solve a mathematically difficult problem with nonnegligible probability. During the mutual attestation phase, the PBA signature about the VCU is similar to the PBA signature about module i; thus, below we just discuss the PBA signature of module i.

Theorem 1: In the random oracle model, the adversary $\mathcal{A}$ with probabilistic polynomial time executes the game. It wins the game with a probability that cannot be ignored in the corresponding polynomial time. The simulator may solve the CDH problem in the polynomial time.

Within the polynomial time t, $\mathcal{A}$ could execute at most q_h Hash queries and q_I Issue queries. l is the length of hash values.

Proof: If the adversary $\mathcal{A}$ can forge the property attestation signature, then there is a algorithm $\mathcal{B}$ that could utilize $\mathcal{A}$ to solve the CDH problem or the discrete logarithm hard problem. The simulator builds some lists to control the constant with oracle queries. L_{h_i} means the hash oracle $\mathcal{O}$ to generate c_{H_i}. L_{I_i} stores the record of the issued certificate (cs_i, ps, cre_i, s), and $cre_i = (a_i, A_i, b_i, B_i, c_i)$. If cs_i is revoked, it lets $s = 0$, otherwise, let $s = 1$. L_s stores the record of generating the PBA signature during the attestation phase. The record is $(N_v, cre_i, \sigma_i, c)$, and $c = 1$ means that module i or the VCU is controlled by adversary $\mathcal{A}$. ■

Hash Oracle: If there exist $(m, h) \in L_h$, the simulator $\mathcal{S}$ returns h. Otherwise, the simulator $\mathcal{S}$ chooses a random number h, and adds (m, h) into the list and returns h.

Attest Oracle: Suppose that adversary $\mathcal{A}$ gets the configuration message cs_i .The simulator $\mathcal{S}$ acts as the TA, and then simulator $\mathcal{S}$ evaluates the cs_i based on ps and queries the oracle $\mathcal{O}$ in this phase, getting the certificate about cs_i. Then, it adds $(cs_i, ps, cre_i, 0)$ in the record L_I. Let N_v, g^n be the random number controlled by adversary $\mathcal{A}$, and the simulator $\mathcal{S}$ chooses N_i, g^m . The adversary chooses $(N_v, g^m, cs_i, ps, cre_i)$ and requests the prover (simulator $\mathcal{S}$) to attest. $\mathcal{S}$ queries $(cs_i, ps, cre_i, 0/1)$ in the list L_I, and computes the PBA signature on cs_i. There are two cases in our assumption.

Case 1: The TPM is security and the module i is an honest participant and acts as expected. The simulator $\mathcal{S}$ computes the signature according to the above protocol.

1) The adversary challenges the simulator $\mathcal{S}$ with random number N_v, g^n, and the simulator $\mathcal{S}$ receives this message and invokes TPM to generate signature. TPM computes $\sigma_i = sk_i(N_v, g^n)$. The simulator $\mathcal{S}$ gets the signature, and sends (N_v, g^n, σ_i) to module i.
2) The simulator $\mathcal{S}$ chooses random number $r', c', s_1, s_2 \in Z_q$, and computes $a = a^{r'}, b = b^{r'}, A = A^{r'}$, and $B = B^{r'}$.
3) The simulator $\mathcal{S}$ computes $u_x = e(X, a')$, $u_{xy} = e(X, b')$, $u_s = e(g, c')$, and $u_{xyz} = e(X, B')$.
4) The simulator $\mathcal{S}$ chooses random $c_H \in \{0,1\}^{L_H}$, and queries whether c_H exists in the L_H or not. If exist, then continue.
5) The simulator $\mathcal{S}$ computes $T = u_s^{s_2} u_{xy}^{-s_1} (u_x u_{xyz}^{ps})^{c_H - 1}$.
6) Set $w = a', A', b', B', c', u_x, u_{xy}, u_{xyz}, u_s, N_v$. The simulator $\mathcal{S}$ queries (c_{Hi}, w). If it exists in the list, then go to the first step, otherwise, add (c_{Hi}, w) to the list L_H.
7) The simulator $\mathcal{S}$ outputs the PBA signature as $\sigma_{\text{PBAi}} = (\sigma_i, c_{Hi}, \delta_i, s_1, s_2, N_v)$.
8) The simulator $\mathcal{S}$ adds $(N_v, g^n, cre_i, \sigma_{\text{PBA}_i}, 0)$ into list L_s.

Case 2: The TPM is physically secure, but the prover (module i) is controlled by adversary. $\mathcal{S}$ simulates the attestation process in this case. If the attestation is completed, it means that the $\mathcal{S}$ generates the correct signature, otherwise, adversary $\mathcal{A}$ forges the property attestation signature.

The protocol outputs the PBA signature represented as $\sigma_{\text{PBAi}} = (\sigma_i, c_{Hi}, \delta_i, s_1, s_2, N_v)$ in the attestation phase, and PBA signature is indistinguishable between the simulation and real case. If adversary $\mathcal{A}$ forges the PBA signature successfully with a nonnegligible probability ε, then we can construct another algorithm $\mathcal{B}$ by using adversary $\mathcal{A}$ to solve the discrete logarithm hard problem as

$$\Pr{}_{A-\text{DLP}} \geq \frac{\varepsilon}{2} \tag{7}$$

or solve the CDH problem as

$$\Pr{}_{A-\text{CDH}} \geq \frac{\varepsilon}{2}. \tag{8}$$

If the adversary could forge the PBA signature on (cs_i, ps), the simulator chooses random $z \in Z_q$, and computes $Z = g^z$. Algorithm $\mathcal{B}$ constructs the system parameter $pk = (q, G, G_T, g, g_T, e, X, Y, Z), sk = (x, y, z)$ like the CDH problem. Suppose that adversary $\mathcal{A}$ queries the attest oracle $\mathcal{O}$ q times. $(cs_j, ps_j)(j = 1, 2, \ldots, q)$ are not queried before, thus $(cs_j, ps_j) \neq (cs, ps)$. There are two cases.

1) For any $(cs_j, ps_j), (j \in (1, p))$, there exists some j for $cs_j, ps_j, j \in (1, p)$ that satisfy
$$\Pr\left[cs + ps * z = cs_j + ps_j * z\right] \geq \frac{\varepsilon}{2}. \tag{9}$$
That is to say, algorithm $\mathcal{B}$ can get z by $z = (cs_j - cs)/(ps - ps_j)$. That means, algorithm $\mathcal{B}$ could solve the discrete logarithm problem.
2) For any $(cs_j, ps_j), (j \in (1, p))$, there exists some j for $cs_j, ps_j, j \in (1, p)$ that satisfy
$$\Pr\left[cs + ps * z \neq cs_j + ps_j * z\right] \geq \frac{\varepsilon}{2}. \tag{10}$$

The simulator $\mathcal{S}$ acts as the verifier to get the PBA signature from adversary $\mathcal{A}$. Since the signature does not belong to the attestation list L_S, the simulator $\mathcal{S}$ has to invoke the attest oracle to generate new c_{Hi}, and the simulator $\mathcal{S}$ could choose the same value $(\sigma_i, C_i, T, a', A', b', B', c')$ with different (c_{Hi}, s_1, s_2) and (c'_{Hi}, s'_1, s'_2). Thus, the signature can be defined as follows: $(\delta_i, T, a', A', b', B', c', c_{Hi1}, s_1, s_2)$ $(\delta_i, T, a', A', b', B', c', c'_{Hi1}, s'_1, s'_2)$.

Let $a' = g^{\alpha}$, $b' = g^{\beta}$, $c' = g^{\gamma}, \alpha, \beta, \gamma \in Z_q^*$, and $c_{Hi1} \neq c'{}_{Hi1}, s_i \neq s'{}_i$ in the two signature. Let $\Delta c_{Hi} = c_{Hi1} - c'_{Hi1}$, $\Delta s_1 = s_1 - s'_1$, and $\Delta s_2 = s_2 - s'_2$. Since the signature is generated by adversary $\mathcal{A}$, the simulator acts as the verifier could get $T_1 = u_s^{s_2} u_{xy}^{-s_1} (u_x u_{xyz}^{ps})^{c_{Hi1} - 1}$, $T_2 = u_s^{s'_2} u_{xy}^{-s'_1} (u_x u_{xyz}^{ps})^{c'_{Hi1} - 1}$, and two signature both pass verification. The verifier attempts to get the cs_i through the following method.

Let $T_1 = T_2$, which means $u_s^{\Delta s_2} = u_{xy}^{-\Delta s_1} (u_x u_{xyz}^{ps})^{\Delta c_{Hi}}$. Since $c_{Hi} \neq 0$, we can let $st_1 = \Delta s_1 / \Delta c_{Hi}, st_2 = \Delta s_2 / \Delta c_{Hi}$. Then, we can get $u_s^{st_2} = u_x u_{xy}^{st_1} u_{xyz}^{ps}$, and then the simulator (act as verifier) could get $cs_i = st_1$ from the equation $u_x u_{xy}^{st_1} u_{xyz}^{ps} = u_s^r$. Let $m = st_1 + ps * zmodq$, and set the (st_1, ps) queries the attest oracle. These parameters must satisfy the equations $e(a', Y) = e(g, b')$, $u_s^{st_2} = u_x u_{xy}^{st_1} u_{xyz}^{ps}$. Combined with the above equations, we can get

$$e\left(g, c'\right)^{st_2} = e\left(X, a'\right) e\left(X, b'\right)^{st_1} e\left(X, B'\right)^{ps}. \tag{11}$$

That means: 1) these equations are true $e(a', Y) = e(g, b')$, $e(g^{\alpha}, g^{y}) = e(g, g^{\beta})$, $g_T^{\alpha y} = g_T^{\beta}$, based on $\alpha y = \beta \bmod p$ and 2) by simple calculation, the equation is satisfied

$$e\left(X, B'\right) = e\left(X, A'^{y}\right) = e\left(X, a'^{yz}\right) = e\left(X . b'\right)^{z}. \tag{12}$$

By combining (1) and (2), we can know $\gamma st_2 = (x\alpha + x\beta m)$, $\gamma st_2 = (x\alpha + x\alpha y m)$.

Then, we can review the whole phase and analyze the possibility that adversary $\mathcal{A}$ solves the CDH problem. The algorithm $\mathcal{B}$ outputs $a' = g^{\alpha}$, $b' = g^{\beta}$, $c' = g^{\gamma}$. Let $a = a'$, $b = b'$, $c = c'^{st2}$, which shows $a = g^{\alpha}$, $b = g^{\beta} = g^{\alpha y} = a^{y}$, $c = c'^{st_2} = g^{\gamma st_2} = g^{\alpha(x + xym)} = a^{x + xym}$. The adversary $\mathcal{A}$ wins the game when the following events happen simultaneously.

1) *Event E_1:* The collision happened on two hash functions H and h. The maximum probability is $\Pr[E_1] = [(q_h^2 + q_H^2)/(2^{l+1})]$.
2) *Event E_2:* The collision happened on issue queries. The maximum probability is $\Pr[E_2] = (q_I^2 / 2q)$.

If these two event happened, $\mathcal{A}$ will win the game and the advantage of adversary is $Adv(\mathcal{A})$. We obtain

$$\text{Adv}(\mathcal{A}) = \Pr[E_1] + \Pr[E_2] = \frac{q_h^2 + q_H^2}{2^{l+1}} + \frac{q_I^2}{2q}. \tag{13}$$

As a result, if adversary $\mathcal{A}$ could calculates xyP in a polynomial time with the advantage of $[(q_h^2 + q_H^2)/(2^{l+1})] + [(q_I^2)/2q]$, it would solve the CDH problem, which satisfies Theorem 1. However, it is difficult to solve the CDH problem within a short time. Therefore, the proposed scheme could against the adaptive chosen message attack under the random oracle model.

B. Security Analysis

1) Trust Measurement: After getting the corresponding certificate from the TA, module i and VCU send the request to get PBA signature. The VCU first verifies the signature $\sigma_{\text{PBA}i}$ including δ_i, c_{Hi}, and σ_i'. In the same manner, module i validates the VCU by verifying whether δ_v, c_{Hv}, and σ_v' in $\sigma_{\text{PBA}v}$ are efficient. The TPM$_i$ cannot be compromised; thus, even if the module i is compromised, the attacker cannot impersonate as module i.

2) Message Authentication: As mentioned earlier, the key is stored in the TPM and the key can be updated. The difficulty of guessing the key value is based on the CDH problem. Therefore, the recipient can use HMAC to verify the integrity of the received message.

3) Privacy Protection: The configuration message cs_t is different for every module, but only TPM$_t$ gets cs_t at the beginning. Then, cs_t satisfies the property of TPM, and cs_t is securely sent to the module. In the entire authentication process, cs_t is only sent to the TA after encryption. That is to say, the module is attacked before the system setup, and the TPM cannot work successfully and will not generate cs_t. During the transmission after encryption, the security of cs_t is ensured by the encryption algorithm. At the last phase of attestation, cs_t is included in σ_{PBA}. If one attacker could get cs_t successfully, it must solve the discrete logarithm hard problem.

4) Resistance to Ordinary Attacks: This scheme could resist the following common attacks.

Impersonation Attacks: An attacker who wants to impersonate a legitimate module needs to successfully deceive the verifier, that is, to produce its unique PBA signature. The certificate and the PBA signature are associated with a unique cs_t. Even if the adversary gets the certificate, it does not acquire the secret value cs_t, nor does it succeed in generating the correct PBA signature. So it cannot successfully pass the verifier's authentication.

Modification Attacks: In this scheme, every message m is sent with HMAC$_k(m)$. If the attacker tries to modify the message, the attacker must modify the HMAC value at the same time. However, the key k_i is different every time; thus, the message authentication with HMAC can resist the modification attack.

Replay Attacks: The VCU and module i check the received message m, HMAC$_k(m)$ every time. In most instances, the message is different each time, but m_k and m_{i+1} are sometimes the same. At this time, the receiver compares HMAC$_k(m)$. Since k and $k+1$ are different, the two values of HMAC are also different even though m and $m+1$ are the same. If the computed value and received value are equal, the message is dropped.

Some security comparison has been shown in Table III, only the scheme could achieve more target when compared with the related scheme [6]–[8].

TABLE III
SECURITY COMPARISON

	[6]	[7]	[8]	The proposed scheme
Impersonation attacks	√	√	√	√
Modificaiton attacks	√	√	√	√
Replay attacks	√	√	×	√
Efficient revocation	√	√	×	√
Trust measurement	×	×	√	√

VI. PERFORMANCE EVALUATION

In this section, the computation and communication overhead of different schemes are compared in detail. Among them, the computation cost of different schemes has been compared. The large integer operation is based on GMP library version 6.1.2, and the pairing calculation is based on PBC library version 0.5.14. The operating system is Ubuntu 16.04, and CPU is Intel Core i7-6700 4 GHz, and the memory is 16 GB. As mentioned earlier, we use SGX as the implementation of TPM [20]. More precisely, the graphene-SGX shown in [42] and [43] is used to build a trusted environment.

Table IV shows the computation overhead comparison of different schemes, where the basic operation execution time is shown in Table V. As mentioned earlier, G_1 and G_t are 128 bytes, and the message and timestamp lengths are 20 and 4 bytes, respectively. The experiment uses an A-type curve, and it could achieve an 80-bit security level.

In similar scenarios, there are some schemes for message authentication in VANETs. We compare the computational overhead of achieving the same function in related schemes [6]–[8]. The functions of the scheme include message authentication, batch authentication, and key update. Jiang *et al.*'s scheme [6] adopted bilinear pairing cryptooperations, and the vehicle sends the message PID$_{i,j}$, M_i, tt_i, $Y_{i,j}$ that requires $3\times128+20\times2+4 = 428$ bytes. For n messages, it could cost $428n$ bytes. Zhang *et al.*'s scheme [7] sent a message M_i, ID_i, T_i, σ_i that will cost 84 bytes, and it will cost $84n$ bytes to send n messages. Desmoulins *et al.*'s scheme [8] sent the signature message that will cost $2 \times 128 = 256$ bytes, and it will cost $256n$ bytes to send n messages. The proposed scheme sends the PBA signature and HMAC, which will cost $128 \times 2 + 20 \times 3 = 316$ bytes to send the signature σ_i', c_{hi}, s_{i1}, s_{i2}. It costs $20\times2n = 40n$ bytes to send n messages as m_i, HMAC$_{k_{n+1-i}}(m_i)$.

To achieve batch authentication, for n messages, Jiang *et al.*'s scheme [6] will cost $2T_{bp.m}+3T_{bp}+nT_{e.a}+nT_h = 0.0276n + 16.051$ ms. Zhang *et al.*'s scheme [7] will cost $(n+2)T_{e.m}+nT_{e.sm}+nT_{e.a}+(2n)T_h = 0.4578n+0.884$ ms to achieve batch authentication. Desmoulins *et al.*'s scheme [8] will cost $nT_{e.m}+nT_{e.sm}+nT_{e.a} = 0.4144n$ ms. In the proposed scheme, it will cost $3T_{bp} + nT_h = 0.013n + 8.422$ ms to achieve message authentication for n messages. In order to achieve key update, Jiang *et al.*'s scheme [6] will cost $T_{e.m} = 1.7090$ ms. Zhang *et al.*'s scheme [7] will cost $T_{e.m} = 0.4420$ ms to achieve the key update. The proposed scheme will run a hash operation T_h, which will cost 0.0138 ms.

The results of the experiment show the associated computational overhead. Fig. 6 shows the computational overhead of the certificate generation and mutual authentication. The certificate generation phase costs 4.945 ms, and the certificate verification costs 6.736 ms. The PBA signature includes δ_i generated in the SGX environment, and it costs 3.314 ms.

TABLE IV
COMPARISON OF THE AUTHENTICATION SCHEMES

	Jiang et al. [6]	Zhang et al. [7]	Desmoulins et al. [8]	The proposed scheme
Message length (bytes)	428n	84n	256n	316+40n
Authentication (time:ms)	0.0276n+16.051	0.4578n + 0.884	0.4144n	0.013n+8.422
Key update (time:ms)	1.7090	0.4420	—	0.0138

TABLE V
EXECUTION TIME OF BASIC CRYPTOGRAPHIC OPERATIONS

Symbol	Cryptographic operation	Time (ms)
T_{bp}	Bilinear pairing operation	4.2110
$T_{bp.m}$	Scale multiplication operation	1.7090
$T_{bp.sm}$	Small scale multiplication operation	0.0535
$T_{bp.a}$	Point addition operation	0.0071
T_{mtp}	The map to point	4.4060
$T_{e.m}$	Scale multiplication operation	0.4420
$T_{e.sm}$	Small scale multiplication operation	0.0276
$T_{e.a}$	Point addition operation	0.0140
T_h	Secure hash operation	0.0138

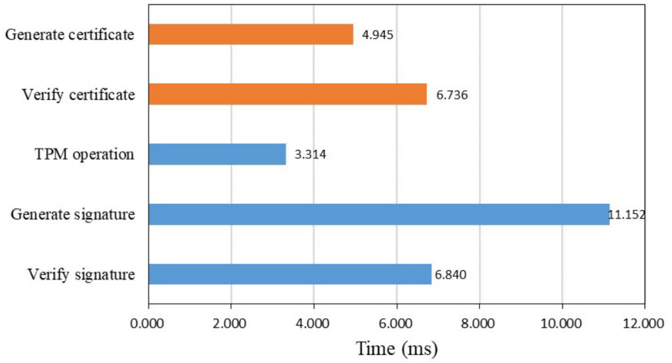


Fig. 6. Computational overhead of certificate generation and mutual authentication.

Note that although the computational overhead increases by 1 ms compared with operation in the normal environment, the SGX guarantees higher security. It costs 11.152 ms for module i to generate the PBA signature. It costs 6.840 ms to verify the PBA signature and generate the session key.

The generation and verification of signatures are necessary in the remote attestation. It would take milliseconds, but it is essential to ensure trust of the modules. All the processes of remote attestation can be implemented periodically, such as when the vehicle starts, this overhead is acceptable.

The related experiment shows the computational overhead of generating and verifying the HMAC with different message lengths (1 KB to 2 MB). The experiment is based on the HMAC-SHA256 algorithm. As shown in Fig. 7, the message length is 1 KB, generating HMAC and verifying the message cost of 30 and 49 μs, respectively. The message length is 2 MB, generating HMAC and verifying the message cost of 2050 and 2119 μs, respectively. Note that the key is generated by TPM in the proposed scheme. It could protect key generation in the SGX environment. It takes 90 μs to generate a key in the SGX environment.

Fig. 8 shows the computational overhead of massive message authentication. In the massive authentication process, the proposed scheme has a lower computational cost. Through the above comparison, the proposed scheme is superior to other schemes for massive message authentication. Since data are being generated all the time, achieving fast message authentication is a crucial requirement for secure CAV communication. The result shows that the proposed scheme can meet the security and efficiency of CAVs.

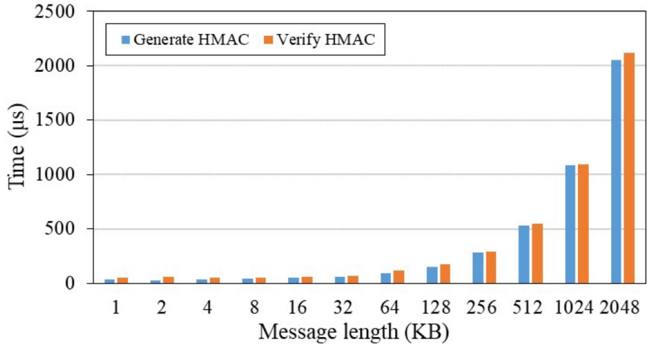


Fig. 7. Computational overhead of HMAC generation and verification.

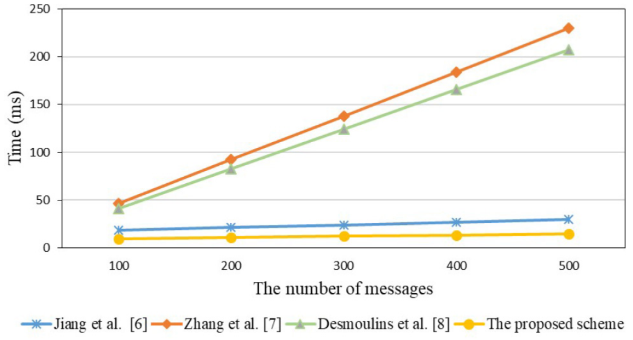


Fig. 8. Overhead of massive message authentication.

VII. CONCLUSION

CAVs can realize automatic driving without human intervention, and its decision is based on real-time data provided by internal modules. Therefore, the secure transmission of real-time data between internal modules is particularly important. Data are generated by these modules, and untrusted modules may generate misleading data, which are fatal to CAVs. Hence, the module trust is a prerequisite for data trust. This article presents a reliable and secure communication scheme based on remote attestation technology. Security analysis indicates that the scheme can satisfy the security requirements, such as message authentication, integrity, and revocation. Based on the CDH problem, the scheme is proved to be forgery resistant in the random oracle model.

The experimental environment is based on graphene-SGX, and the experimental results demonstrate that the scheme balances usability and security. Therefore, this scheme can realize the secure communication of CAV internal modules and meet the practical requirements.

Acknowledgment

The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this article.

References

[1] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proc. IEEE*, vol. 107, no. 8, pp. 1697–1716, Aug. 2019.

[2] Q. Zhang *et al.*, "OpenVDAP: An open vehicular data analytics platform for CAVs," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2018, pp. 1310–1320.

[3] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—Part I: Distributed system architecture and development process," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7131–7140, Dec. 2014.

[4] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, Nov. 2017.

[5] J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1621–1632, May 2019.

[6] S. Jiang, X. Zhu, and L. Wang, "An efficient anonymous batch authentication scheme based on HMAC for VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2193–2204, Aug. 2016.

[7] J. Zhang, J. Cui, H. Zhong, Z. Chen, and L. Liu, "PA-CRT: Chinese remainder theorem based conditional privacy-preserving authentication scheme in vehicular ad-hoc networks," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 2, pp. 722–735, Mar./Apr. 2021,

[8] N. Desmoulins, A. Diop, Y. Rafflé, J. Traoré, and J. Gratesac, "Practical anonymous attestation-based pseudonym schemes for vehicular networks," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, 2019, pp. 1–8.

[9] M. Xu *et al.*, "Dominance as a new trusted computing primitive for the Internet of Things," in *Proc. IEEE Symp. Security Privacy (SP)*, 2019, pp. 1415–1430.

[10] J. Yuan and X. Li, "A reliable and lightweight trust computing mechanism for IoT edge devices based on multi-source feedback information fusion," *IEEE Access*, vol. 6, pp. 23626–23638, 2018.

[11] W. Yan, A. Fu, Y. Mu, X. Zhe, S. Yu, and B. Kuang, "EAPA: Efficient attestation resilient to physical attacks for IoT devices," in *Proc. 2nd Int. ACM Workshop Security Privacy Internet Things*. 2019, pp. 2–7.

[12] I. A. Sumra, H. B. Hasbullah, and J.-L. Ab Manan, "Using TPM to ensure security, trust and privacy (STP) in VANET," in *Proc. 5th Nat. Symp. Inf. Technol. Towards New Smart World (NSITNSW)*, 2015, pp. 1–6.

[13] J. Zhao, J. Liu, Z. Qin, and K. Ren, "Privacy protection scheme based on remote anonymous attestation for trusted smart meters," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3313–3320, Jul. 2018.

[14] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Proc. Annu. Int. Cryptol. Conf.*, 2004, pp. 56–72.

[15] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. 11th ACM Conf. Comput. Commun. Security*, 2004, pp. 132–145.

[16] E. Brickell, L. Chen, and J. Li, "A new direct anonymous attestation scheme from bilinear maps," in *Proc. Int. Conf. Trusted Comput.*, 2008, pp. 166–178.

[17] L. Chen, R. Landfermann, H. Löhr, M. Rohe, A.-R. Sadeghi, and C. Stüble, "A protocol for property-based attestation," in *Proc. 1st Workshop Scalable Trusted Comput.*, 2006, pp. 7–16.

[18] D. Feng and Y. Qin, "A property-based attestation protocol for TCM," *Sci. China Inf. Sci.*, vol. 53, no. 3, pp. 454–464, 2010.

[19] D. Amelino, M. Barbareschi, and A. Cilardo, "An IP core remote anonymous activation protocol," *IEEE Trans. Emerg. Topics Comput.*, vol. 6, no. 2, pp. 258–268, Apr.–Jun. 2018.

[20] L. Yang, J. Ma, W. Lou, and Q. Jiang, "A delegation based cross trusted domain direct anonymous attestation scheme," *Comput. Netw.*, vol. 81, pp. 245–257, Apr. 2015.

[21] F. Yang, L. Pan, M. Xiong, and S. Tang, "Establishment of security levels in trusted cloud computing platforms," in *Proc. IEEE Int. Conf. Green Comput. Commun. Internet Thin. Cyber Phys. Social Comput.*, 2013, pp. 2119–2122.

[22] L. Chen, S. Qian, M. Lim, and S. Wang, "An enhanced direct anonymous attestation scheme with mutual authentication for network-connected UAV communication systems," *China Commun.*, vol. 15, no. 5, pp. 61–76, 2018.

[23] Z. Wang, "A privacy-preserving and accountable authentication protocol for IoT end-devices with weaker identity," *Future Gener. Comput. Syst.*, vol. 82, pp. 342–348, May 2018.

[24] X. Zeng, G. Xu, X. Zheng, Y. Xiang, and W. Zhou, "E-AUA: An efficient anonymous user authentication protocol for mobile IoT," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1506–1519, Apr. 2019.

[25] Y. Liu, L. Wang, and H. Chen, "Message authentication using proxy vehicles in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 8, pp. 3697–3710, Aug. 2015.

[26] J. Cui, J. Chen, H. Zhong, J. Zhang, and L. Liu, "Reliable and efficient content sharing for 5G-enabled vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 24, 2020, doi: 10.1109/TITS.2020.3023797.

[27] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, and L. Liu, "Edge computing in VANETs-an efficient and privacy-preserving cooperative downloading scheme," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1191–1204, Jun. 2020.

[28] C. Hu, T. W. Chim, S.-M. Yiu, L. C. Hui, and V. O. Li, "Efficient HMAC-based secure communication for VANETs," *Comput. Netw.*, vol. 56, no. 9, pp. 2292–2303, 2012.

[29] M. Ashritha and C. S. Sridhar, "RSU based efficient vehicle authentication mechanism for VANETs," in *Proc. IEEE 9th Int. Conf. Intell. Syst. Control (ISCO)*, 2015, pp. 1–5.

[30] H. Khemissa and D. Tandjaoui, "A novel lightweight authentication scheme for heterogeneous wireless sensor networks in the context of Internet of Things," in *Proc. Wireless Telecommun. Symp. (WTS)*, 2016, pp. 1–6.

[31] D. Halabi, S. Hamdan, and S. Almajali, "Enhance the security in smart home applications based on IoT-CoAP protocol," in *Proc. 6th Int. Conf. Digit. Inf. Netw. Wireless Commun. (DINWC)*, 2018, pp. 81–85.

[32] T. Lin, P. Wu, F. Gao, and L. Wang, "A secure query protocol for multi-layer wireless sensor networks based on Internet of Things," *Revue Intell. Artificielle*, vol. 33, no. 2, pp. 145–149, 2019.

[33] T. W. Chim, S.-M. Yiu, L. C. Hui, and V. O. Li, "PASS: Privacy-preserving authentication scheme for smart grid network," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, 2011, pp. 196–201.

[34] K. Mahmood, S. A. Chaudhry, H. Naqvi, T. Shon, and H. F. Ahmad, "A lightweight message authentication scheme for smart grid communications in power sector," *Comput. Electr. Eng.*, vol. 52, pp. 114–124, May 2016.

[35] J. Kang, R. Yu, X. Huang, and Y. Zhang, "Privacy-preserved pseudonym scheme for fog computing supported Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2627–2637, Aug. 2018.

[36] B. Kuang, A. Fu, S. Yu, G. Yang, M. Su, and Y. Zhang, "ESDRA: An efficient and secure distributed remote attestation scheme for IoT swarms," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8372–8383, Oct. 2019.

[37] E. Dushku, M. M. Rabbani, M. Conti, L. V. Mancini, and S. Ranise, "SARA: Secure asynchronous remote attestation for IoT systems," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3123–3136, 2020.

[38] A.-R. Sadeghi and C. Stüble, "Property-based attestation for computing platforms: Caring about properties, not mechanisms," in *Proc. Workshop New Security Paradigms*, 2004, pp. 67–77.

[39] N. Asokan *et al.*, "SEDA: Scalable embedded device attestation," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Security*, 2015, pp. 964–975.

[40] N. Koutroumpouchos *et al.*, "Secure edge computing with lightweight control-flow property-based attestation," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, 2019, pp. 84–92.

[41] L. Buttyán, T. Holczer, A. Weimerskirch, and W. Whyte, "SLOW: A practical pseudonym changing scheme for location privacy in VANETs," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, 2009, pp. 1–8.

[42] V. Costan and S. Devadas, "Intel SGX explained," IACR Cryptol. ePrint Archive, Lyon, France, Rep. 2016/086, 2016.

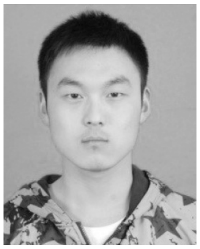
[43] C.-C. Tsai, D. E. Porter, and M. Vij, "Graphene-SGX: A practical library OS for unmodified applications on SGX," in *Proc. USENIX Annu. Tech. Conf.)*, 2017, pp. 645–658.

Hong Zhong was born in Anhui, China, in 1965. She received the Ph.D. degree in computer science from the University of Science and Technology of China, Hefei, China, in 2005.

She is currently a Professor and a Ph.D. supervisor with the School of Computer Science and Technology, Anhui University, Hefei. She has over 120 scientific publications in reputable journals (e.g., IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE TRANSACTIONS ON BIG DATA, and IEEE INTERNET OF THINGS JOURNAL), academic books, and international conferences. Her research interests include applied cryptography, IoT security, vehicular *ad hoc* network, cloud computing security, and software-defined networking.

Wenwen Cao is currently pursuing the Graduation degree with the School of Computer Science and Technology, Anhui University, Hefei, China.

His research focuses on the security of connected and autonomous vehicles.

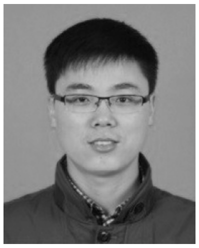


Qingyang Zhang received the B.Eng. degree in computer science and technology from Anhui University, Hefei, China, in 2014, where he is currently pursuing the Ph.D. degree.

His research interest includes edge computing, computer systems, and security.

Jing Zhang (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Anhui University, Hefei, China.

She has over ten scientific publications in reputable journals (e.g., IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *Information Sciences*, *Science China Information Sciences*, and *Vehicular Communications*), and international conferences. Her research interests include vehicular *ad hoc* network, IoT security, and applied cryptography.



Jie Cui (Member, IEEE) was born in Henan, China, in 1980. He received the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2012.

He is currently a Professor and a Ph.D. supervisor with the School of Computer Science and Technology, Anhui University, Hefei. He has over 100 scientific publications in reputable journals (e.g., IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and IEEE INTERNET OF THINGS JOURNAL), academic books, and international conferences. His current research interests include applied cryptography, IoT security, vehicular *ad hoc* network, cloud computing security, and software-defined networking.