An Efficient Pairing-free Certificateless Signcryption Scheme under the Standard Model for VANET

Xianxiang Liu, Yanwei Zhou, Bo Yang, Tianqing Zhu, Mingwu Zhang

***Abstract*—With the rapid advancement of the Internet of Things (IoT), its applications are becoming increasingly essential in actual application. Specifically, the recent surge in electric vehicles has spurred significant advancements in vehicle ad hoc networks (VANET). Therefore, efficient data transmission is a critical challenge in IoT, especially VANET. The certificateless signcryption (CLS) scheme has high authentication efficiency, which has become increasingly important in IoT. However, most existing schemes rely on bilinear pairing, resulting in high calculation costs or failure to achieve their claimed security. Furthermore, the security of some CLS schemes is proved in the random oracle, which limits the usefulness of the CLS scheme. To further address these issues, we propose an efficient and secure CLS scheme in the standard model for VANET, which enhances its practical applicability in VANET. Our construction employs formal security proofs and cost simulations to ensure the scheme's security and low calculation costs, making it suitable for VANET. Finally, the security and efficiency analyses prove that our scheme offers enhanced security features and superior performance compared to existing schemes.**

***Index Terms*—Certificateless Signcryption; Vehivle Ad Hoc Networks; Satndard Model; Provable Security.**

I. Introduction

With the rapid development of the Internet of Things (IoT) technology, the interconnection and communication among devices enhance our daily lives [1], [2], Vehicle Ad Hoc Networks (VANET) as a crucial component of the IoT, are gaining increasing attention due to their low cost and high efficiency for road transportation, where real-time information about road conditions and vehicle locations can be transmitted through wireless communication between vehicles and roadside units, thereby significantly enhancing travel efficiency [3]. VANET adopts advanced data transmission protocols and algorithms to improve transportation safety and efficiency, where the drivers can receive the latest traffic information, such as accidents, road construction, and real-time traffic flow. In recent years, the rapid development of electric vehicles has further accelerated the iteration of VANET [4], which can take advantage of the VANET, further enhancing the driving experience and enabling them to focus more on driving rather than complex traffic situations, the advancements in smart driving and driver-assistance technologies make our travel more convenient and comfortable, invigorating the potential of VANET, which provides crucial data to vehicles, aiding in the selection of optimal routes and enhancing the intelligence of traffic management. However, as VANET evolves rapidly, researchers have raised concerns regarding the security of transmission data [5], [6] and the attackers could intercept or alter information, leading vehicles to receive false messages, and others have expressed concerns about the integrity and accuracy of messages within VANET [7], the messages received by vehicles from roadside units may be unreliable if the integrity of information cannot be guaranteed, posing a threat to driver safety. Therefore, ensuring data privacy and security in VANET is of utmost importance [8].

To further solve the key-escrow problem of identity-based public key cryptography, the definition of certificateless public key cryptography is proposed [9], which does not require a completely trusted third party to build the system, reduces the maintenance cost of the system, and is more suitable for use in VANET environment.

For the security and efficiency requirements of data transmission of VANET, we propose a novel certificateless signcryption (CLS) scheme to achieve secure and efficient data transmission to address the above issues in VANET, and we prove the security of our construction in the standard model. Therefore, the proposed data transmission protocol's security is formally proven in the standard model, ensuring enhanced practicability in practical applications.

A. Related Works

In recent years, many researchers have proposed the concrete constructions of CLS scheme and designed different application protocols based on the above constructions [10]–[15].

(1) ***CLS scheme under the random oracle model*.** Eltayieb *et al.* [16] and Li *et al.* [17] proposed an online/offline signcryption for IoT to reduce the computation cost and claims that their scheme can guarantee data transportation and ensure data security. However, the scheme used the bilinear operations, which made the corresponding protocol less efficient.

This work is supported by the National Natural Science Foundation of China (62272287, U24B20149), the Key Research and Development Projectin Shaanxi Province (2024GX-YBXM-074), the Research Fund for Key Laboratory of Data Protection and Intelligent Management, Ministry of Education, Sichuan University (SCUSAKFKT202401Y), and the Fundamental Research Funds for the Central Universities (GK202301009). (Corresponding authors: Yanwei Zhou)

Yanwei Zhou is with the School of Computer Science, Shaanxi Normal University, Xi'an 710119, China; and also with the Key Laboratory of Data Protection and Intelligent Management, Ministry of Education, Sichuan University, Chengdu 610065, China.(zyw@snnu.edu.cn)

Xianxiang Liu and Bo Yang are with the School of Computer Science, Shaanxi Normal University, Xi'an 710119, China.

Tianqing Zhu is with the Faculty of Data Science, City University of Macau, Macau 999078, China.

Mingwu Zhang is with the School of Computer, Hubei University of Technology, Wuhan, 430068, China.

Manuscript received April 19, 2021; revised August 16, 2021.

Furthermore, the scheme did not provide formal security proof, which cannot guarantee the unforgeability and confidentiality of messages in data transmission. Han *et al*. [18] proposed a lightweight hybrid CLS scheme, which avoided bilinear operations to increase the efficiency of calculation. Obiri *et al*. [19] proposed a new CLS scheme with the proxy re-encryption function to strongly ensure data security and avoided bilinear operations to reduce calculation costs. Qiao *et al*. [20] proposed a novel anonymous signcryption scheme for VANETs, which achieved computational efficiency in data transmission while protecting vehicle identities from exposure. To achieve multi-message sending, Li *et al*. [21] and Dai *et al*. [22] proposed a certificateless aggregate signcryption (CLAS) scheme for power systems, which achieved multiple message sending and avoided bilinear operations to reduce the communication cost. Furthermore, Zhou *et al*. [23] proposed a novel construction of the CLS scheme, which can achieve data transmission of multi-message and multi-receiver and improved the data transmission efficiency in VANET.

In conclusion, most of the above cryptographic primitives created from certificateless public key cryptography can avoid the bilinear operations to enhance their efficiency. However, the random oracle model relied on strong security assumptions, the security proven in the random oracle model was not as secure as under the standard model.

(2) *CLS scheme under the standard model*. Luo *et al*. [24] proposed a safety-enhanced CLS scheme with a formal proof based on bilinear pairing. However, Yuan *et al*. [25] proved that Luo *et al*. proposed scheme was vulnerable to adversary attacks and failed to meet its claimed security guarantees. Xu *et al*. [26] proposed a CLS scheme for edge computing based on bilinear pairing, which achieved message confidentiality and unforgeability. Zhou *et al*. [27] proposed a CLS scheme in the power system to protect against attacks from adversaries, and Deng *et al*. [28] proposed an efficient CLS scheme by using short signatures in smart healthcare, and claimed that it had confidentiality and unforgeability based on the Computation Attack Algorithm (CAA) Problem through formal proof. Gao *et al*. consider Deng *et al*.'s scheme is vulnerable to adversary [29]. However, the above CLS schemes relied on bilinear operations, which resulted in high computational costs. Long *et al*. [30] also proposed a CLS scheme to achieve secure data transmission in IoT, which avoided using bilinear operations to increase calculation efficiency. Although, the scheme provides a formal proof of its claimed security, the formal proof contains errors. So the scheme can not guarantee the corresponding security. We present the security proof of the scheme in the appendix A, where we identify a challenge in the proof. Cobblah *et al*. [31] proposed a CLS scheme for VANET, which did not use bilinear operations to reduce calculation costs and was claimed to have unforgeability by providing formal proof. In fact, a Type I adversary can replace a user's public key to forge a valid ciphertext, and the unforgeability of the above scheme was broken. Although both schemes avoided bilinear operations, they failed to achieve the security guarantees claimed by their constructions, the detailed reasoning process will be shown in the appendix B.

(3) *The Limitations of the Existing Constructions*. From the above analysis, it can be seen that the current research on CLS scheme under the standard model mainly has the following shortcomings:

(1) **The formal proof of CLS is key challenge in the standard model.** Most existing certificateless schemes in the standard model are based on bilinear pairings, where the signature is an element in a group. The verification process relies on pairings. So the scheme can insert the parameters of The Computational Diffie-Hellman (CDH) Problem in signature and the properties of bilinear pairings to solve the CDH problem [26]–[28]. To achieve a standard model proof for pairing-free CLS, the simulator must simultaneously extract the unknown parameter number b of Discrete Logrithm Assumption (DL) from forged signatures (the signaure has two unkown number) using two linearly independent equations, which making it impossible to resolve two unknowns without the forking lemma [23]. However forking lemma is prohibited in the standard model. Therefore, the formal security proof for a pairing-free CLS scheme under the standard model remains a key challenge. Some researchers have recently proposed pairing-free CLS schemes [30], [31] under the standard model, either the proposed solution fails to meet security requirements or cannot resolve the key challenge. Consequently, its security cannot be reliably proven.

(2) **Security proof is indirect.** In many schemes [24], [26], the security proof of unforgeability and confidentiality are independent. The reason for the above results is that the encryption element and the signature element of the ciphertext cannot be generated simultaneously in the security proof due to the design of scheme. However, in practical vehicular communication scenarios, the encryption and signing operations are inherently intertwined during real-time message transmission. Therefore, separating them may lead certain security concerning.

To sum up, creating a CLS scheme with better performance in security and computational efficiency has become an urgent challenge in practical applications. So, we propose a novel CLS scheme under the standard model. Our scheme employs the CAA problem to overcome the challenge of security proof, which can ensure both the security and efficiency of data transmission in VANET.

B. *Our Motivations*

In VANET, ensuring data confidentiality and unforgeability during vehicle communication is crucial. However, many devices, such as roadside units (RSUs), have limited computing resources in this environment. In related work, we have observed that existing pairing-free CLS schemes either achieve high efficiency but fail to meet security requirements in standard model, or they meet security requirements but rely on bilinear pairings in the standard model, which leads to low efficiency.

However, no CLS scheme has been proposed that does not rely on bilinear pairings and is formally proven secure in the standard model. Such a scheme would significantly improve the efficiency of vehicle communication in VANETs and ensure high security for data transmission due to the standard model proof. Therefore, our motivation is to tackle this key challenge and propose a novel CLS scheme in the standard model.

(1) **The weak security of pairing-free CLS scheme under random oracle models:** Many schemes under the random oracle model [16], [18]–[21], [23] avoid the bilinear pairing operation, and the security proven under the random oracle model is not as secure as under the standard model.

(2) **The huge computational cost of the CLS scheme by bilinear pairing in the standard model:** Most existing CLS schemes in the standard model rely on bilinear pairing operations [26]–[28], which leads to high calculation costs. Hence, the above schemes are unsuitable for the VANET environments that demand high calculation efficiency.

(3) **The security of pairing-free CLS scheme under the standard model is not guaranteed:** Some CLS schemes can avoid bilinear operations in the standard model [30], [31]. However, its security cannot be reliably proven.

(4) **Security proof is indirect:** The proof of confidentiality and unforgeability of most CLS schemes is independent [24], [26], which is not the original design of the CLS scheme. Therefore, direct proof of CLS schemes in the standard model is a meaningful topic.

C. Our Contributions

Designing a CLS scheme that ensures both better security and higher efficiency would greatly enhance the communication efficiency and security in VANET. Therefore, after analyzing the key challenging aspects of existing pairing-free CLS schemes, and based on our motivation, we propose a novel CLS scheme under the standard model. This scheme aims to improve both the security and efficiency of data transmission compared to related schemes. The main contributions are as follows:

(1) **Our proposed scheme avoids bilinear pairing to enhance efficiency:** In VANET, data transmission efficiency is essential for rapid data exchange between vehicles, allowing them to quickly access information on road conditions, locations, and traffic density. Our proposed scheme eliminates bilinear operations, significantly reducing the calculation cost of data transmission.

(2) **The security of our solution satisfies the requirements of the data transmission:** Our novel CLS scheme not only can ensure data confidentiality, but also can ensure message unforgeability under the standard model, providing strong security for data transmission.

(3) **Our proposed scheme guarantees security by rigorous security proof:** Our scheme's security proof not noly proves Indistinguishability under Adaptive Chosen Ciphertext Attack (IND-CCA) based on the static Decision Diffie-Hellman (DDH) assumption, but also utilizes the CAA Problem to prove Existential Unforgeability under Adaptive Chosen Message Attack (EUF-CMA) under the standard model, which brakes the limitation of forking-lemma of random oracle model.

II. Preliminaries

In this section, some theoretical basis employed in our construction will be provided. In addition, the definition of symbols is shown in Table I.

TABLE I
THE DEFINITION OF SYMBOLS USED IN OUR PAPER

Symbol	Definition
κ	the security parameter
KGC	key generation center
id_i	the identity of user
id_S	the identity of sender
id_R	the identity of the receiver
G	additive cyclic group with order q
P	the generator of group G
$H_{i(i=1,2,3,4,5)}$	secure one-way hash functions
$Params$	system public parameter
msk	master secret key of KGC
P_{pub}	master public key of KGC
$pk_i = (X_i, Y_i)$	public key of user with the identity id_i
$sk_i = (x_i, y_i)$	private key of user with the identity id_i

A. Complexity Assumptions

Definition 1 (Decision Diffie-Hellman (DDH) Assumption [18])**.** *Given an additive group G of order q, let $a, b \in \mathbb{Z}_q^*$ be randomly choosen, let $P \in G$. The DDH assumption states that for any polynomial-time adversary, the advantage of distinguishing between the tuples (aP, bP, abP) and (aP, bP, D) is negligible, where D is a random element from G.*

Definition 2 (Computation Attack Algorithm (CAA) Problem [28])**.** *Given a tuple (P, aP), where $a \in \mathbb{Z}_q^*$ and P is a generator of the additive group G of order q. The goal of the CAA problem is to find a value c, such that, $c \neq a$ and outputs a tuple $(c, \frac{1}{a+c}P)$. The advantage of any adversary in solving CAA problem is negligible.*

Definition 3 (Discrete Logarithm (DL) Assumption)**.** *Given tuple (P, bP), where $b \in \mathbb{Z}_q^*$, the target of DL problem is to get b. The advantage of solving the DL problem is negligible.*

III. Certificateless Signcryption Scheme

The formal definition and security model of CLS scheme are reviewed in this subsection.

A. Formal Definition

The formal definition of the CLS scheme is recalled in the following:

Setup. KGC runs the setup algorithm $\mathsf{Setup}(1^k)$ to generate the system's master secret key msk and the public parameters $Params$ by inputting the secure parameters κ. This process can be described as $(Params, msk) \leftarrow \mathsf{Setup}(1^\kappa)$.

Key Generation. The user with the identity id_i interacts with KGC to generate the partial key. Finally, the user generates the complete public key pk_i and the complete private key sk_i. This process can be described as: $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(Params, msk, id_i)$.

Signcryption. For the messages M_S and two identities tuple (id_S, id_R) of sender and receiver, we run the signcryption algorithm by inputting a tuple $(pk_R, sk_S, Params)$ to generate signcryption ciphertext δ_S on the message M_S. This process can be described as: $\delta_S \leftarrow \mathsf{SignCry}(M_S, pk_R, sk_S, Params)$.

Unsigncryption. Upon receiving the signcryption ciphertext, the receiver verifies its integrity by using the sender's public key pk_S. If the verification passes, the receiver decrypts the message using the private key sk_R and returns the original message M_S; Otherwise, "0" is output. This process is described as $M_S/0 \leftarrow \mathsf{UnSign}(pk_S, sk_R, Params, \delta_S)$.

B. The Formal Definition of Correctness Analysis

The scheme $\prod = (\mathsf{KeyGen}, \mathsf{SignCry}, \mathsf{Unsign})$ satisfies correctness if for every security parameter $\lambda \in \mathbb{N}$ and message $m \in \mathcal{M}$ (where $\mathcal{M}$ denotes the message space), there exists a negligible function $\mathrm{negl}(\lambda)$ such that:

$$\Pr\begin{bmatrix}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda);\\ \delta \leftarrow \mathsf{SignCry}(pk, m);\\ m' \leftarrow \mathsf{Unsign}(sk, pk, \delta) \text{ and } \mathsf{Verify}(pk, \delta) = 1;\\ m' = m\end{bmatrix} \geq 1 - \mathsf{negl}(\lambda)$$

Message Recovery: The message can be recovered as $m' = \mathsf{Unsign}(sk, pk, \delta)$ with overwhelming probability.

Signature Validity: For honestly generated signatures, $\mathsf{Verify}(pk, \delta) = 1$.

C. Security Model

In certificateless cryptographic systems, adversaries are systematically categorized into two distinct types based on their operational capabilities. Type I adversary is malicious users, who operate without knowledge of the system master public key. Their primary attack methodology involves attempts to replace public keys in order to breach the system's security, typically through intercepting vehicle-to-vehicle communications or impersonating legitimate users. Type II adversary is malicious KGC, who possess legitimate key generation and distribution capabilities. Their threat arises from potential abuse of trusted authorities to extract sensitive user information, though they are constrained by their inability to directly replace user public keys. This model captures the systemic corruption risks within trusted authorities. The formal security proofs address real-world threat scenarios, ranging from infrastructure compromise to external attacks.

- Type I adversary: $\mathcal{A}_1$ is a malicious user who cannot obtain the master secret key but can replace the public key. Adversary $\mathcal{A}_1$ is further classified into two subcategories: $\mathcal{A}_1^1$ and $\mathcal{A}_1^2$. The adversary $\mathcal{A}_1^1$ solely attacks on confidentiality, while $\mathcal{A}_1^2$ focuses entirely on the unforgeability of signcryption.
- Type II adversary: $\mathcal{A}_2$ is a malicious KGC that can obtain the master key but cannot replace the public key. Adversary $\mathcal{A}_2$ is further classified into two subcategories: $\mathcal{A}_2^1$ and $\mathcal{A}_2^2$. The adversary $\mathcal{A}_2^1$ solely attacks on confidentiality. Conversely, the adversary $\mathcal{A}_2^2$ focuses entirely on the unforgeability of signcryption.

CLS scheme defines four games to prove confidentiality and unforgeability [23]. **Game 1** and **Game 2** prove the scheme's confidentiality, while **Game 3** and **Game 4** prove the scheme's unforgeability.

Definition 4 (Confidentiality)**.** *For Type I adversary $\mathcal{A}_1^1$ and Type II adversary $\mathcal{A}_2^1$, if the adversary's advantages of winning in **Game 1** and **Game 2** are negligible, the proposed scheme is proven to meet confidentiality.*

Game 1. This game is played between the challenger $\mathcal{C}$ and an adversary $\mathcal{A}_1^1$.

Setup. The challenger $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$ to generate the public parameters $Params$ and the system master secret key msk. The challenger $\mathcal{C}$ sends the public parameters $Params$ to $\mathcal{A}_1^1$. Here, we select id^* as the challenge identity.

Phase 1. During this stage, $\mathcal{A}_1^1$ performs adaptive queries in probabilistic polynomial time (PPT).

(1) ***Public Key Extraction***. When $\mathcal{A}_1^1$ queries the public key for identity id_i, $\mathcal{C}$ runs the key generation algorithm for identity id_i and returns the corresponding public key pk_i to $\mathcal{A}_1^1$.

(2) ***Private Key Extraction***. When $\mathcal{A}_1^1$ queries the private key for id_i. If $id_i = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the key generation algorithm for identity id_i and returns the corresponding private key sk_i to $\mathcal{A}_1^1$.

(3) ***Public Key Replacement***. When $\mathcal{A}_1^1$ submits a new public key pk_i' for identity id_i, $\mathcal{C}$ replaces the existing public key pk_i with pk_i'.

(4) ***Unsigncryption***. Upon receiving a message tuple $(id_S', id_R', \delta_S')$ as the input of the unsigncryption query, if $id_R' = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ executes the unsigncryption algorithm to decrypt the ciphertext and returns the corresponding decryption result.

Challenge. During this phase, $\mathcal{A}_1^1$ sends two identities $\{id_S, id_R\}$ and two length-equal messages $\{M_0, M_1\}$ to $\mathcal{C}$. If $id_R \neq id^*$, the game aborts and returns $\perp$; Otherwise, the challenger $\mathcal{C}$ randomly selects $d \in \{0, 1\}$ and signcrypts message M_d by executing signcryption algorithm to generate the signcryption ciphertext δ^*. Finally, $\mathcal{C}$ sends corresponding signcryption ciphertext δ^* to $\mathcal{A}_1^1$.

Phase 2. $\mathcal{A}_1^1$ continues adaptive queries as in ***Phase 1*** but cannot request the private key sk^* with identity id^* as input.

Guess. The adversary $\mathcal{A}_1^1$ outputs a guess d' for the random value d. $\mathcal{A}_1^1$ wins the game if these conditions are met:

- At no point was the private key sk^* for the challenge identity id^* queried.
- After replacing the public key pk_i, the private key sk_i for identity id_i was not queried.
- $\mathcal{A}_1^1$ cannot replace the corresponding public key of the challenge identity id^*.

The advantage of $\mathcal{A}_1^1$ winning in the above game is $\mathsf{Adv}_{\mathcal{A}_1^1}(\kappa) = \left|\Pr[\mathcal{A}_1^1 \text{ wins}] - \frac{1}{2}\right|$.

Game 2. This game is played between the challenger $\mathcal{C}$ and an adversary $\mathcal{A}_2^1$.

Setup. The challenger $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$ to generate the public parameters $Params$ and the system master secret key msk. The challenger $\mathcal{C}$ sends the public parameters $Params$ and master secret key to $\mathcal{A}_2^1$. Here, we select id^* as the challenge identity.

Phase 1. During this stage, $\mathcal{A}_2^1$ performs adaptive queries in PPT.

(1) ***Public Key Extraction***. When $\mathcal{A}_2^1$ queries the public key for identity id_i, $\mathcal{C}$ runs the key generation algorithm for identity id_i and returns the corresponding public key pk_i to $\mathcal{A}_2^1$.

(2) ***Private Key Extraction***. When $\mathcal{A}_2^1$ queries the private key for id_i. If $id_i = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the key generation algorithm for the identity id_i and returns the corresponding private key sk_i to $\mathcal{A}_2^1$.

(3) ***Unsigncryption***. When $\mathcal{C}$ receives a message tuple $(id'_S, id'_R, \delta'_S)$ as an unsigncryption query, if $id'_R = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ executes the unsigncryption algorithm to decrypt the ciphertext and returns the corresponding decryption result.

Challenge. During this phase, $\mathcal{A}_2^1$ sends two identities $\{id_S, id_R\}$ and two length-equal messages $\{M_0, M_1\}$ to $\mathcal{C}$. If $id_R \neq id^*$, the game aborts and returns $\perp$; Otherwise, the challenger $\mathcal{C}$ randomly selects $d \in \{0,1\}$ and signcrypts message M_d by using signcryption algorithm to generate the signcryption ciphertext δ^*. Finally, $\mathcal{C}$ sends corresponding signcryption ciphertext δ^* to $\mathcal{A}_2^1$.

Phase 2. $\mathcal{A}_2^1$ continues adaptive queries as in ***Phase 1*** but cannot request the private key sk^* with the challenge identity id^*.

Guess. The adversary $\mathcal{A}_2^1$ outputs a guess d' for the random value d. $\mathcal{A}_2^1$ wins the game if the condition is met:

- At no point was the private key sk^* for the challenge identity id^* queried.

The advantage of $\mathcal{A}_2^1$ in the above game is $\mathsf{Adv}_{\mathcal{A}_2^1}(\kappa) = \left|\Pr[\mathcal{A}_2^1\ wins] - \frac{1}{2}\right|$.

Definition 5 (Unforgeability)**.** *For the type I adversary $\mathcal{A}_1^2$ and type II adversary $\mathcal{A}_2^2$, the advantages of adversaries winning in **Game 3** and **Game 4** are negligible, which proves that the scheme meets unforgeability.*

Game 3. The game takes place between the challenger $\mathcal{C}$ and an adversary $\mathcal{A}_1^2$.

Setup. $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$ to generate the public parameters $Params$ and the system master secret key msk. Then, $\mathcal{C}$ sends the public parameters $Params$ to $\mathcal{A}_1^2$. Here, we select id^* as the challenge identity.

Query Phase. During this stage, adversary $\mathcal{A}_1^2$ performs adaptive queries in PPT, including private key extraction, public key extraction, and public key replacement queries similar to **Game 1**.

(1) ***Signcryption***. When receiving a signcryption query for the tuple (id_S, id_R, M_S). If $id_S = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the signcryption algorithm to generate the signcryption ciphertext δ_S. $\mathcal{C}$ then sends corresponding ciphertext δ_S to $\mathcal{A}_1^2$.

Forgery. $\mathcal{A}_1^2$ forges a signcrypiton ciphertext δ^* for the challenge tuple (id_S^*, id_R^*, M^*). If $id_S^* \neq id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{A}_1^2$ wins in the above game, the following conditions are met:

- The forged signcryption ciphertext δ^* is valid.
- At no point during any phase was the private key sk^* for the challenge identity id^* queried.
- After the public key pk_i of identity id_i has been replaced, the private key sk_i of identity id_i cannot be queried by adversary $\mathcal{A}_1^2$.

The advantage of $\mathcal{A}_1^2$ in the above game is $\mathsf{Adv}_{\mathcal{A}_1^2}(\kappa) = \left|\Pr[\mathcal{A}_1^2\ wins]\right|$.

Game 4. The game takes place between the challenger $\mathcal{C}$ and an adversary $\mathcal{A}_2^2$.

Setup. $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$ to generate the public parameters $Params$ and the system master secret key msk. Then, $\mathcal{C}$ sends the public parameters $Params$ and the master secret key msk to $\mathcal{A}_2^2$. Here, we select id^* as the challenge identity.

Query Phase. During this stage, adversary $\mathcal{A}_2^2$ performs adaptive queries in PPT, including private key extraction and public key extraction similar to **Game 2**.

(1) ***Signcryption***. When receiving a Signcryption query for the tuple (id_S, id_R, M_S). If $id_S = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the signcryption algorithm to generate the signcryption ciphertext δ_S. $\mathcal{C}$ then sends corresponding ciphertext δ_S to $\mathcal{A}_2^2$.

Forgery. $\mathcal{A}_2^2$ forges a signcrypiton ciphertext δ^* for the challenge tuple (id_S^*, id_R^*, M^*). If $id_S^* \neq id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{A}_2^2$ wins in the above game, the following conditions are met:

- The forged signcryption δ^* is valid.
- At no point during any phase was the private key sk^* for the challenge identity id^* queried.

The advantage of $\mathcal{A}_2^2$ in the above game is $\mathsf{Adv}_{\mathcal{A}_2^2}(\kappa) = \left|\Pr[\mathcal{A}_2^2\ wins]\right|$.

IV. Secure and Efficient Certificateless Signcryption Scheme under the Standard Model

In this section, we describe the pairing-free CLS scheme in the standard model, building upon the formal definitions, correctness analysis, and security model established in previous sections.

A. Concrete Construction

The specific operations of each algorithm in our construction are described below.

(1) $(Params, msk) \leftarrow \mathsf{Setup}(1^\kappa)$

Given a security parameter κ, KGC uses it to initialize the system. The KGC selects an additive group G of prime order q, and P is a generator of G. It then randomly selects a master secret key $s \in \mathbb{Z}_q^*$ $(msk = s)$ and calculates the master public key $P_{pub} = sP$. KGC selects five one-way hash functions: $H_1, H_2, H_3, H_4, H_5 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. Finally, the KGC publishes the public parameters: $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$.

(2) $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(Params, msk, id_i)$

At this stage, a user with identity id_i randomly selects a secret value $x_i \in \mathbb{Z}_q^*$ and computes the public key as $X_i = x_iP$. The user then sends the tuple (id_i, X_i) to the KGC.

Upon receiving the tuple, KGC randomly selects another secret value $u_i \in \mathbb{Z}_q^*$ and computes $Y_i = u_iP$ and $y_i = u_i + sh_1 \bmod q$, where $h_1 = H_1(id_i, X_i, Y_i, P_{pub})$. The KGC then sends the partial keys tuple (y_i, Y_i) to the user by secure channel.

Upon receiving this tuple, the user verifies the correctness of the equation: $y_iP = Y_i + h_1P_{pub}$. If the equation holds, the user sets the public key as $pk_i = (X_i, Y_i)$ and the private key as $sk_i = (x_i, y_i)$. If the equation does not hold, the user re-executes the key generation algorithm to regenerate the necessary public and private keys.

(3) $\delta_S \leftarrow \mathsf{SignCry}(M_S, pk_R, sk_S, Params)$

The sender with identity id_S selects random values $k, t \in \mathbb{Z}_q^*$ and computes $K = kP$, $T_1 = tP$ and $T_2 = \frac{1}{t}P$. The sender calculates the encryption factor

$$Q_S = k(X_R + Y_R + h_1P_{pub}) + t(Y_R + h_1P_{pub})$$

by using the receiver's public key, where $h_1 = H_1(id_R, X_R, Y_R, P_{pub})$. The message M_S is encrypted using the encryption factor $C_S = M_S \oplus h_2(Q_S)$. The sender also calculates

$$z_S = \frac{t}{h_3x_S + h_4y_S} \text{ and } \sigma_S = \frac{\frac{1}{t} + 1}{h_3x_S + h_5y_S}.$$

where $h_3 = H_3(id_S, id_R, pk_S, pk_R, C_S, K, T_1, T_2)$, $h_4 = H_4(id_S, id_R, pk_S, pk_R, C_S, K, T_1, T_2)$, and $h_5 = H_5(id_S, id_R, pk_S, pk_R, C_S, K, T_1, T_2, z_S)$. Finally, the sender transmits the ciphertext $\delta_S = \{C_S, K, T_1, T_2, z_S, \sigma_S\}$ to the receiver.

(4) $M_S/0 \leftarrow \mathsf{UnSign}(pk_S, sk_R, Params, \delta_S)$

Upon receiving the signcryption ciphertext δ_S, the receiver checks whether the following holds: $z_S(h_3X_S + h_4(Y_S + h_1P_{pub})) + \sigma_S(h_3X_S + h_5(Y_S + h_1P_{pub})) = T_1 + T_2 + P$ to verify the integer of the ciphertext. If checks fail, the receiver outputs "0"; Otherwise, the receiver uses the private key to recover the encryption factor $Q'_S = (x_R + y_R)K + T_1y_R$, and recovers the plaintext M_S by calculating $M_S = C_S \oplus H_2(Q'_S)$.

B. Correctness Analysis.

To ensure the correctness of the scheme, the processes of signature verification and the recovery of the encryption factor are detailed as follows.

$$\begin{aligned}
Q'_S =& (x_R + y_R)K + y_RT_1 \\
=& (x_R + y_R)kP + ty_RP \\
=& k(X_R + Y_R + h_1P_{pub}) + t(Y_R + h_1P_{pub}) = Q_S
\end{aligned}$$

$$\begin{aligned}
&z_S(h_3X_S + h_4(Y_S + h_1P_{pub})) + \sigma_S(h_3X_S + h_5(Y_S + h_1P_{pub})) \\
&= z_S(h_3x_S + h_4y_S)P + \sigma_S(h_3x_S + h_5y_S)P \\
&= \frac{tP}{h_3x_S + h_4y_S}(h_3x_S + h_4y_S) + \frac{\frac{1}{t}P + P}{h_3x_S + h_5y_S}(h_3x_S + h_5y_S) \\
&= tP + \frac{1}{t}P + P \\
&= T_1 + T_2 + P
\end{aligned}$$

From the correctness analysis of our scheme, we observe that the encryption factor can be correctly recovered and the verification process is also valid. Hence, we conclude that our scheme satisfies message recoverability and signature validity. Consequently, the scheme is correct in key generation, signcryption, and unsigncryption.

C. Security Proof.

In this section, we prove the confidentiality and unforgeability of our proposed scheme. The security proof emphasizes that attackers must account for both the encryption process and the corresponding signature. However, it is important to note that in the confidentiality proof of CLS, the unsigncryption algorithm solely decrypts the ciphertext without performing any verification. Likewise, in the unforgeability proof of the CLS scheme, the unsigncryption algorithm only verifies the validity of the signature, regardless of the correctness of decrypting the ciphertext.

1) Confidentiality: Our scheme achieves confidentiality through two theorems. **Theorem 1** proves that the scheme is CCA secure against a Type I adversary $\mathcal{A}_1^1$, while **Theorem 2** proves IND-CCA security against a Type II adversary $\mathcal{A}_2^1$.

Theorem 1. *Under the standard model, if adversary $\mathcal{A}_1^1$ successfully wins the corresponding **Game 1** with a non-negligible advantage ε_1, there exists a challenger $\mathcal{C}$ who can solve the DDH assumption with a non-negligible advantage of $\frac{\varepsilon_1}{e(1+Q_1)}$, where Q_1 is the number of private key extraction query and e is the base of the natural constant.*

Proof. The challenger $\mathcal{C}$ acts as the adversary to the DDH assumption, who is given a challenge tuple (aP, bP, D) based on the DDH assumption, where $D = abP$ or $D = cP$, with c being a random number. The goal of $\mathcal{C}$ is to distinguish two tuples within PPT.

Setup. When the game begins, $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$. That is, $\mathcal{C}$ selects a random value s as master secret key to set master public key $P_{pub} = sP$, and then sends the public parameters $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ to $\mathcal{A}_1^1$. The lists L_{pk} and L_{sk} are used to trace public key extraction and private key extraction queries under the standard model. Here, we select id^* as the challenge identity.

Phase 1. During this phase, $\mathcal{A}_1^1$ executes adaptive queries. The operations for public key extraction, private key extraction, public key replacement, and unsigncryption queries are as follows:

Public Key Extraction. Upon receiving a public key extraction query for the identity id_i. If a tuple (id_i, X_i, Y_i) exists in L_{pk}, $\mathcal{C}$ returns corresponding public key (X_i, Y_i) to $\mathcal{A}_1^1$; Otherwise, the following conditions apply:

(1) If $id_i \neq id^*$, $\mathcal{C}$ runs key generation algorithm to generate public key $pk_i = (X_i, Y_i)$ and private key sk_i. Then $\mathcal{C}$ adds tuples (id_i, X_i, Y_i) and (id_i, x_i, y_i) to L_{pk} and L_{sk} respectively. The challenger $\mathcal{C}$ returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_1^1$.

(2) If $id_i = id^*$, $\mathcal{C}$ sets $X^* = aP$ (implicit setting $x^* = a$), and chooses $u^* \in \mathbb{Z}_q^*$ to calculate $Y^* = u^*P$

and $y^* = u^* + sh_1$, where $h_1 = H_1(id^*, X^*, Y^*, P_{pub})$. Afterwords, $\mathcal{C}$ keeps u^* secret and adds tuples (id^*, X^*, Y^*) and $(id^*, \perp, y^*)$ to L_{pk} and L_{sk} respectively. The challenger $\mathcal{C}$ returns corresponding public key $pk^* = (X^*, Y^*)$ to $\mathcal{A}_1^1$.

Private Key Extraction. Upon receiving a private key extraction query for an identity id_i. If $id_i = id^*$, the game aborts and returns $\perp$; Otherwise, the remaining condition is as following:

If $(id_i, x_i, y_i) \in L_{sk}$, $\mathcal{C}$ returns corresponding private key $sk_i = (x_i, y_i)$; Otherwise, $\mathcal{C}$ executes the public key extraction process to obtain corresponding private key $sk_i = (x_i, y_i)$, and then returns it to $\mathcal{A}_1^1$.

Public Key Replacement. If $\mathcal{A}_1^1$ submits a new public key $pk_i' = (X_i', Y_i')$ for identity id_i. Then $\mathcal{C}$ updates L_{pk} from (id_i, X_i, Y_i) to (id_i, X_i', Y_i').

Unsigncryption. Upon receiving a ciphertext tuple $(id_S', id_R', \delta_S')$ as an unsigncryption query, if $id_R' = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the unsigncryption algorithm to decrypt the ciphertext and returns the corresponding result to $\mathcal{A}_1^1$.

Challenge. The adversary $\mathcal{A}_1^1$ sends two identities (id_S, id_R) along with two length-equal messages (M_0, M_1). since id_S is not the challenge identity, the adversary $\mathcal{A}_1^1$ can gain the public and private keys of identity id_S. As a result, $\mathcal{C}$ can obtain the tuples (id_R, X_R, Y_R), (id_S, X_S, Y_S) and (id_S, x_S, y_S) through queries in PPT.

(1) If $id_R \neq id^*$, the game aborts and returns $\perp$.

(2) Otherwise, $\mathcal{C}$ obtains $y^* = u^* + sh_1$ from L_{sk} with id^*. $\mathcal{C}$ sets $K = bP$ (implicit setting $k = b$) and randomly selects $t \in \mathbb{Z}_q^*$ and calculates $T_1 = tP$ and $T_2 = \frac{1}{t}P$. The challenger $\mathcal{C}$ recovers encryption factor $Q_S' = D + y^*(K + T_1)$. Next, $\mathcal{C}$ encrypts the message $C_S = M_d \oplus H_2(Q_S')$ and then calculates $z_S = t/(h_3 x_S + h_4 y_S)$ and $\sigma_S = (\frac{1}{t} + 1)/(h_3 x_S + h_5 y_S)$, where $h_3 = H_3(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2)$, $h_4 = H_4(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2)$, $h_5 = H_5(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2, z_S)$. Afterward, $\mathcal{C}$ sends the challenge ciphertext $\delta_S = (\sigma_S, C_S, K, T_1, T_2, z_S)$ to $\mathcal{A}_1^1$.

Phase 2. $\mathcal{A}_1^1$ continues adaptive queries as in ***Phase 1*** but cannot request the private key sk^* with identity id^*.

Guess. After performing the above queries for a polynomial number of times, the adversary $\mathcal{A}_1^1$ outputs the guess d' for d. If $d = d'$, $\mathcal{C}$ wins the game.

When $D = abP$, the following equation holds, allowing the adversary $\mathcal{A}_1^1$ to obtain valid information:

$$\begin{aligned} Q_S' &= D + y^*(K + T_1) \\ &= abP + (Y^* + h_1 P_{pub})(k + t) \\ &= kX^* + (Y^* + h_1 P_{pub})(k + t) \\ &= k(X^* + Y^* + h_1 P_{pub}) + t(Y^* + h_1 P_{pub}) = Q_S \end{aligned}$$

Otherwise, D is a random element, so $\mathcal{A}_1^1$ gains a random message.

If $\mathcal{C}$ does not abort during the simulation process and adversary $\mathcal{A}_1^1$ breaks the confidentiality of this scheme with a non-negligible advantage, $\mathcal{C}$ outputs a valid solution to the DDH assumption. Use F_1 to denote that the game will not abort during the query phase, F_2 to denote that the game will not abort during the challenge phase, and F_3 to denote that adversary $\mathcal{A}_1^1$ correctly guesses d. Let Q_1 represent the number of private keys extracted by $\mathcal{A}_1^1$. Therefore, the probability that each phase will not abort is the following:

$$\Pr[F_1] = \left(1 - \frac{1}{(1+Q_1)}\right)^{Q_1}, \ \Pr[F_2] = \frac{1}{(1+Q_1)}, \ \Pr[F_3] = \varepsilon_1.$$

In conclusion, if $\mathcal{C}$ does not halt during the simulation, and the adversary $\mathcal{A}_1^1$ breaks the confidentiality of the scheme with a non-negligible advantage ε_1, $\mathcal{C}$ will output the valid solution to the DDH assumption with a non-negligible advantage of $\Pr[F_1 \wedge F_2 \wedge F_3] = \frac{\varepsilon_1}{e(1+Q_1)}$. □

Theorem 2. *Under the standard model, if the adversary A_2^1 successfully wins the corresponding **Game 2** with a non-negligible advantage ε_2, there exists the challenger $\mathcal{C}$ who can solve the DDH assumption with a non-negligible advantage $\frac{\varepsilon_2}{e(1+Q_2)}$, where Q_2 is the number of private key extract query.*

Proof. The challenger $\mathcal{C}$ acts as the adversary to the DDH assumption, who is given a challenge tuple (aP, bP, D) based on the DDH assumption, where $D = abP$ or $D = cP$, with c being a random number. The goal of $\mathcal{C}$ is to distinguish two tuples within PPT.

Setup. When the game begins, $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^\kappa)$. That is, $\mathcal{C}$ selects a random value s as master secret key to set master public key $P_{pub} = sP$, and then sends the public parameters $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ and master secret key s to $\mathcal{A}_2^1$. The lists L_{pk} and L_{sk} are used to trace public key extraction and private key extraction queries under the standard model. Here, we select id^* as the challenge identity.

Phase 1. During this phase, $\mathcal{A}_2^1$ executes adaptive queries. The operations for public key extraction, private key extraction, public key replacement, and unsigncryption queries are as follows:

Public Key Extraction. Upon receiving a public key extraction query for the identity id_i. If a tuple (id_i, X_i, Y_i) exists in L_{pk}, $\mathcal{C}$ returns corresponding public key (X_i, Y_i) to $\mathcal{A}_2^1$; Otherwise, the following conditions apply:

(1) If $id_i \neq id^*$, $\mathcal{C}$ runs key generation algorithm to generate public key $pk_i = (X_i, Y_i)$ and private key sk_i. Then $\mathcal{C}$ adds tuples (id_i, X_i, Y_i) and (id_i, x_i, y_i) to L_{pk} and L_{sk} respectively. The challenger $\mathcal{C}$ returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_2^1$.

(2) If $id_i = id^*$, $\mathcal{C}$ sets $X^* = aP$ (implicit setting $x^* = a$), and chooses $u^* \in \mathbb{Z}_q^*$ to calculate $Y^* = u^*P$ and $y^* = u^* + sh_1$, where $h_1 = H_1(id^*, X^*, Y^*, P_{pub})$. Afterwords, $\mathcal{C}$ keeps u^* secret and adds tuples (id^*, X^*, Y^*) and $(id^*, \perp, y^*)$ to L_{pk} and L_{sk} respectively. The challenger $\mathcal{C}$ returns corresponding public key $pk^* = (X^*, Y^*)$ to $\mathcal{A}_2^1$.

Private Key Extraction. Upon receiving a private key extraction query for an identity id_i. If $id_i = id^*$, the game aborts and returns $\perp$; Otherwise, the remaining condition is as following:

If $(id_i, x_i, y_i) \in L_{sk}$, $\mathcal{C}$ returns corresponding private key $sk_i = (x_i, y_i)$; Otherwise, $\mathcal{C}$ executes the public key extraction process to obtain corresponding private key $sk_i = (x_i, y_i)$, and then returns it to $\mathcal{A}_2^1$.

Unsigncryption. Upon receiving a ciphertext tuple $(id'_S, id'_R, \delta'_S)$ as an unsigncryption query, if $id'_R = id^*$, the game aborts and returns $\perp$; Otherwise, $\mathcal{C}$ runs the unsigncryption algorithm to decrypt the ciphertext and returns the corresponding result to $\mathcal{A}_2^1$.

Challenge. The adversary $\mathcal{A}_2^1$ sends two identities (id_S, id_R) along with two length-equal messages (M_0, M_1). since id_S is not the challenge identity, the adversary $\mathcal{A}_2^1$ can gain the public and private keys of identity id_S. As a result, $\mathcal{C}$ can obtain the tuples (id_R, X_R, Y_R), (id_S, X_S, Y_S) and (id_S, x_S, y_S) through queries in PPT.

(1) If $id_R \neq id^*$, the game aborts and returns $\perp$.

(2) Otherwise, $\mathcal{C}$ obtains $y^* = u^* + sh_1$ from L_{sk} with id^*. $\mathcal{C}$ sets $K = bP$ (implicit setting $k = b$) and randomly selects $t \in \mathbb{Z}_q^*$ and calculates $T_1 = tP$ and $T_2 = \frac{1}{t}P$. The challenger $\mathcal{C}$ recovers encryption factor $Q'_S = D + y^*(K + T_1)$. Next, $\mathcal{C}$ encrypts the message $C_S = M_d \oplus H_2(Q'_S)$ and then calculates $z_S = \frac{t}{h_3 x_S + h_4 y_S}$ and $\sigma_S = \frac{\frac{1}{t}+1}{h_3 x_S + h_5 y_S}$, where $h_3 = H_3(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2)$, $h_4 = H_4(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2)$, $h_5 = H_5(id_S, id^*, pk_S, pk^*, C_S, K, T_1, T_2, z_S)$. Afterward, $\mathcal{C}$ sends the challenge ciphertext $\delta_S = (\sigma_S, C_S, K, T_1, T_2, z_S)$ to $\mathcal{A}_2^1$.

Phase 2. $\mathcal{A}_2^1$ continues adaptive queries as in ***Phase 1*** but cannot request the private key sk^* with identity id^*.

Guess. After performing the above queries for a polynomial number of times, the adversary $\mathcal{A}_2^1$ outputs the guess d' for d. If $d = d'$, $\mathcal{C}$ wins the game.

When $D = abP$, the following equation holds, allowing the adversary $\mathcal{A}_2^1$ to obtain valid information:

$$\begin{aligned} Q'_S &= D + y^*(K + T_1) \\ &= abP + (Y^* + h_1 P_{pub})(k + t) \\ &= kX^* + (Y^* + h_1 P_{pub})(k + t) \\ &= k(X^* + Y^* + h_1 P_{pub}) + t(Y^* + h_1 P_{pub}) = Q_S \end{aligned}$$

Otherwise, D is a random element, so $\mathcal{A}_2^1$ gains a random message.

In conclusion, if $\mathcal{C}$ does not halt during the simulation, and $\mathcal{A}_2^1$ breaks the confidentiality of the scheme with a non-negligible advantage ε_2, $\mathcal{C}$ outputs the valid solution to the DDH assumption with a non-negligible advantage of $\frac{\varepsilon_2}{e(1+Q_1)}$. □

2) Unforgeability: For our CLS scheme, the unforgeability is gained by the following two theorems. **Theorem 3** proves that our proposal is unforgeable to the Type I adversary $\mathcal{A}_1^2$, and **Theorem 4** proves that our proposal is unforgeable to the Type II adversary $\mathcal{A}_2^2$.

Theorem 3. *Under the standard model, if the adversary $\mathcal{A}_1^2$ successfully wins the corresponding **Game 3** with a non-negligible advantage ε_3, there exists a challenger $\mathcal{C}$ who can solve the CAA problem with a non-negligible advantage $\frac{\varepsilon_3}{e}(1 - \frac{1}{q})$, where e is the base of the natural constant.*

Proof. The challenger $\mathcal{C}$ acts as the adversary to the CAA problem, and a challenge tuple (aP, P) is obtained, where a is a random number. The goal of $\mathcal{C}$ is to output a tuple $(c, \frac{1}{a+c}P)$.

Setup. When the game starts, $\mathcal{C}$ runs the setup algorithm Setup(1^κ) to generate the public parameters $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ and master secret key $msk = s$, where $P_{pub} = sP$, and sends $Params$ to $\mathcal{A}_1^2$. Two lists L_{pk} and L_{sk} are used to trace the adversary $\mathcal{A}_1^2$ for the public key extraction and the private key extraction queries.

Phase 1. In this stage, $\mathcal{A}_1^2$ executes adaptive queries, and private key extraction query and public key replacement query are similar to **Theorem 1**.

Public Key Extraction. When $\mathcal{C}$ receives the public key extract query from $\mathcal{A}_1^2$ with id_i as the input. If list L_{pk} exists tuple (id_i, X_i, Y_i), then $\mathcal{C}$ returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_1^2$; Otherwise, $\mathcal{C}$ does the following operation.

(1) If $id_i \neq id^*$, $\mathcal{C}$ runs key generation algorithm to generate public key pk_i and private key sk_i. Furthermore, $\mathcal{C}$ adds tuples (id_i, x_i, y_i) and (id_i, X_i, Y_i) to the lists L_{sk}, L_{pk} respectively and returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_1^2$.

(2) If $id_i = id^*$, $\mathcal{C}$ sets $X^* = aP$ (implicit setting $x^* = a$), and chooses $u^* \in \mathbb{Z}_q^*$ to compute $Y^* = u^*P$ and $y^* = u^* + sh_1$, where $h_1 = H_1(id^*, X^*, Y^*, P_{pub})$ and $\mathcal{C}$ keeps u^* in secret. Finally, $\mathcal{C}$ adds tuples $(id^*, \perp, y^*)$, (id^*, X^*, Y^*) to the lists L_{sk}, L_{pk} respectively and returns corresponding public key $pk^* = (X^*, Y^*)$ to $\mathcal{A}_1^2$.

Signcryption. When $\mathcal{C}$ receives signcryption query for the tuple (id_S, id_R, M_S). $\mathcal{C}$ queries the public key and private key to obtain the tuple (id_R, X_R, Y_R), (id_S, X_S, Y_S) and (id_R, x_R, y_R) from L_{pk} and L_{sk} respectively.

(1) If $id_S = id^*$, the game aborts and returns $\perp$.

(2) Otherwise, $\mathcal{C}$ runs the signcryption algorithm to generate the signcryption ciphertext δ_S, and sends the signcryption cipheretext δ_S to $\mathcal{A}_1^2$.

Forgery. After all possible queries are made, $\mathcal{A}_1^2$ forges a signcryption tuple $\delta^* = (\sigma^*, C^*, T_1^*, T_2^*, K^*, z^*)$ for the challenge tuple (id_S^*, id_R^*, M^*). If $id_S^* \neq id^*$ or the forged signcryption ciphertext is not valid, then the game aborts and outputs $\perp$; Otherwise, $\mathcal{C}$ computes $\omega = h_3^* z^* T_2^*$ and $c = \frac{h_4^*}{h_3^*}y^*$. If $c \neq a$, then $\mathcal{C}$ can output tuple (c, ω) as the solution of CAA problem.

$$\begin{aligned} \omega = h_3^* z^* T_2^* &= h_3^* \frac{t^*}{h_3^* x^* + h_4^* y^*} \cdot \frac{1}{t^*}P \\ &= h_3^* \frac{1}{h_3^* x^* + h_4^* y^*}P = \frac{1}{x^* + \frac{h_4^*}{h_3^*}y^*}P \\ &= \frac{1}{a + c}P \end{aligned}$$

Where $h_3^* = H_3(id^*, id_R^*, pk^*, pk_R^*, C^*, K^*, T_1^*, T_2^*)$ and $h_4^* = H_4(id^*, id_R^*, pk^*, pk_R^*, C^*, K^*, T_1^*, T_2^*)$.

Let E_1 denote that the game does not abort in the query phase, and E_2 denote that the game does not abort in the forgery phase. Let E_3 represents $c \neq a$. Thus, the probabilities of every phases are

$$\Pr[E_1] = (1 - \frac{1}{1 + Q_3})^{Q_3}, \quad \Pr[E_2] = \varepsilon_3, \quad \Pr[E_3] = 1 - \frac{1}{q}$$

In there, Q_3 is the number of private key extraction query. Therefore, the probability that $\mathcal{A}_1^2$ can win the game and solve the CAA problem with a non-negligible advantage $\Pr[E_1 \wedge E_2 \wedge E_3] = \frac{\varepsilon_3}{e}(1-\frac{1}{q})$. Therefore, if there exists an adversary $\mathcal{A}_1^2$ who can break the unforgeability of out CLS scheme, then we can build a challenger $\mathcal{C}$ who can solve the CAA hard problem. □

Theorem 4. *Under the standard model, if the adversary $\mathcal{A}_2^2$ successfully wins the corresponding **Game 4** with a non-negligible advantage ε_4, then there exists a challenger $\mathcal{C}$ who can solve the CAA problem with a non-negligible advantage $\frac{\varepsilon_4}{e}(1-\frac{1}{q})$.*

Proof. The challenger $\mathcal{C}$ acts as the adversary to the CAA problem, and a challenge tuple (aP, P) is obtained, where a is a random number. The goal of $\mathcal{C}$ is to output a tuple $(c, \frac{1}{a+c}P)$.

Setup. When the game starts, $\mathcal{C}$ runs the setup algorithm $\mathsf{Setup}(1^{\kappa})$ to generate the public parameters $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ and master secret key $msk = s$, where $P_{pub} = sP$, and sends $Params$ and master secret key s to $\mathcal{A}_2^2$. Two lists L_{pk} and L_{sk} are used to trace the adversary $\mathcal{A}_2^2$ for the public key extraction and the private key extraction queries.

Phase 1. In this stage, $\mathcal{A}_2^2$ executes adaptive queries, and private key extraction query are similar to **Theorem** 2.

Public Key Extraction. When $\mathcal{C}$ receives the public key extract query from $\mathcal{A}_2^2$ with id_i as the input. If list L_{pk} exists tuple (id_i, X_i, Y_i), then $\mathcal{C}$ returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_2^2$; Otherwise, $\mathcal{C}$ does the following operation.

(1) If $id_i \neq id^*$, $\mathcal{C}$ runs key generation algorithm to generate public key pk_i and private key sk_i. Furthermore, $\mathcal{C}$ adds tuples (id_i, x_i, y_i) and (id_i, X_i, Y_i) to the lists L_{sk}, L_{pk} respectively and returns corresponding public key $pk_i = (X_i, Y_i)$ to $\mathcal{A}_2^2$.

(2) If $id_i = id^*$, $\mathcal{C}$ sets $X^* = aP$ (implicit setting $x^* = a$), and chooses $u^* \in \mathbb{Z}_q^*$ to compute $Y^* = u^*P$ and $y^* = u^* + sh_1$, where $h_1 = H_1(id^*, X^*, Y^*, P_{pub})$ and $\mathcal{C}$ keeps u^* in secret. Finally, $\mathcal{C}$ adds tuples $(id^*, \perp, y^*)$, (id^*, X^*, Y^*) to the lists L_{sk}, L_{pk} respectively and returns corresponding public key $pk^* = (X^*, Y^*)$ to $\mathcal{A}_2^2$.

Signcryption. When $\mathcal{C}$ receives signcryption query for the tuple (id_S, id_R, M_S). $\mathcal{C}$ queries the public key and private key to obtain the tuple (id_R, X_R, Y_R), (id_S, X_S, Y_S) and (id_R, x_R, y_R) from L_{pk} and L_{sk} respectively.

(1) If $id_S = id^*$, the game aborts and returns $\perp$.

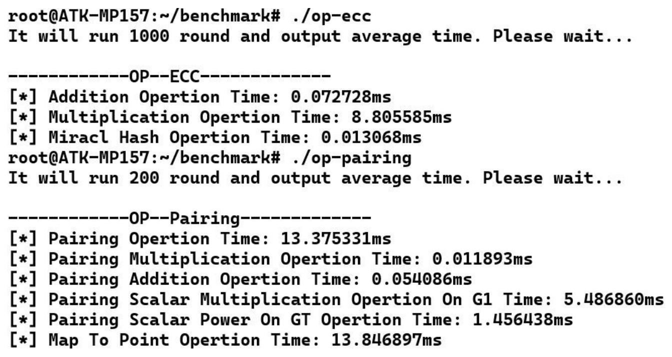
(2) Otherwise, $\mathcal{C}$ runs the signcryption algorithm to generate the signcryption ciphertext δ_S, and sends the signcryption cipheretext δ_S to $\mathcal{A}_2^2$.

Forgery. After all possible queries are made, $\mathcal{A}_2^2$ forges a signcryption tuple $\delta^* = (\sigma^*, C^*, T_1^*, T_2^*, K^*, z^*)$ for the challenge tuple (id_S^*, id_R^*, M^*). If $id_S^* \neq id^*$ or the forged signcryption ciphertext is not valid, then the game aborts and outputs $\perp$; Otherwise, $\mathcal{C}$ computes $\omega = h_3^* z^* T_2^*$ and $c = \frac{h_4^*}{h_3^*}y^*$. If $c \neq a$, then $\mathcal{C}$ can output tuple (c, ω) as the solution of CAA problem.

$$\begin{aligned}
\omega = h_3^* z^* T_2^* &= h_3^* \frac{t^*}{h_3^* x^* + h_4^* y^*} \cdot \frac{1}{t^*} P \\
&= h_3^* \frac{1}{h_3^* x^* + h_4^* y^*} P = \frac{1}{x^* + \frac{h_4^*}{h_3^*} y^*} P \\
&= \frac{1}{a+c} P
\end{aligned}$$

Where $h_3^* = H_3(id^*, id_R^*, pk^*, pk_R^*, C^*, K^*, T_1^*, T_2^*)$ and $h_4^* = H_4(id^*, id_R^*, pk^*, pk_R^*, C^*, K^*, T_1^*, T_2^*)$.

Therefore, the probability that $\mathcal{A}_2^2$ can win the game and solve the CAA problem with a non-negligible advantage $\frac{\varepsilon_4}{e}(1-\frac{1}{q})$. Therefore, if there exists an adversary $\mathcal{A}_2^2$ who can break the unforgeability of out CLS scheme, then we can build a challenger $\mathcal{C}$ who can solve the CAA hard problem. □



```
root@ATK-MP157:~/benchmark# ./op-ecc
It will run 1000 round and output average time. Please wait...

------------OP--ECC-------------
[*] Addition Opertion Time: 0.072728ms
[*] Multiplication Opertion Time: 8.805585ms
[*] Miracl Hash Opertion Time: 0.013068ms
root@ATK-MP157:~/benchmark# ./op-pairing
It will run 200 round and output average time. Please wait...

------------OP--Pairing-------------
[*] Pairing Opertion Time: 13.375331ms
[*] Pairing Multiplication Opertion Time: 0.011893ms
[*] Pairing Addition Opertion Time: 0.054086ms
[*] Pairing Scalar Multiplication Opertion On G1 Time: 5.486860ms
[*] Pairing Scalar Power On GT Opertion Time: 1.456438ms
[*] Map To Point Opertion Time: 13.846897ms
```

Fig. 1. The time cost for each calculation operation using Raspberry Pi model MP157

TABLE II
OPERATION EXECUTION TIME USING RASPBERRY PI

Symbol	Calculate Operation	Execuate Time
T_{pa}	The addition operation of points on elliptic curves	0.0727
T_{mc}	Multiplication operations on elliptic curves	8.8055
T_{ppa}	Point addition operation for linear pairing	0.0540
T_{bp}	Linear pairing operation	13.3753
T_{me}	Exponentiation operation of a module	1.4564

D. Performance Analysis

To thoroughly illustrate the high efficiency of our proposed scheme, we divide the analysis into two parts. First, we perform a simulation comparative efficiency analysis using raspberry pi, followed by experimental evaluations conducted in a Java simulation environment.

1) ***Simulation Comparative Efficiency Analysis Using Raspberry Pi:*** To enhance the practical applicability of our scheme, we compared its calculation cost with that of other schemes. Using the Raspberry Pi model MP157 with a clock frequency of 800 MHz, we collected experimental data for each operation to simulate real-world vehicle calculation operation cost. Fig. 1 illustrates the time cost for each operation, and for convenience, we have organized this data in Table II.

In this section, we begin by calculating the theoretical calculation cost of each scheme [12], [14], [26], [28], [30], [31], including the signcryption, verification, and unsigncryption

TABLE III
THEORETICAL COMPARISON CALCULATION COST OF THE VARIOUS SCHEMES

Scheme	Signcryption	Authentication	Unsigncryption	Security	Model
Balde *et al.* [12]	$6T_{mc}$+$3T_{pa}$	$5T_{mc}$+$3T_{pa}$	$6T_{mc}$+T_{pa}	yes	random oracle
Chen *et al.* [14]	$11T_{mc}$+$5T_{pa}$	$7T_{mc}$+$2T_{pa}$	$8T_{mc}$+$3T_{pa}$	yes	random oracle
Xu *et al.* [26]	$1T_{mc}$+$2T_{bp}$	$2T_{bp}$+$1T_{me}$+$1T_{mc}$	$2T_{bp}$	yes	standard model
Deng *et al.* [28]	$6T_{mc}$+$3T_{ppa}$+$1T_{bp}$	$1T_{bp}$+$1T_{mc}$+$2T_{pa}$	$1T_{bp}$+$2T_{mc}$	no	standard model
Long *et al.* [30]	$4T_{mc}$+$2T_{ppa}$	$4T_{mc}$+$3T_{pa}$	$3T_{mc}$+T_{ppa}	no	standard model
Cobblah *et al.* [31]	$4T_{mc}$+$1T_{pa}$	$4T_{mc}$+$1T_{pa}$	$4T_{mc}$+$2T_{pa}$	no	standard model
Our Scheme	$4T_{mc}$+$3T_{pa} + 3T_{ppa}$	$5T_{mc}$+$4T_{pa}$	$2T_{mc}$+$1T_{pa}$	yes	standard model

operations, based on data collected from the Raspberry Pi. We present the calculated data clearly in table III. The table presents the theoretical calculation cost at each step.

(1) **Security Analysis.** In Table III, we analyze the security of the schemes proposed by Deng *et al.*, Long *et al.* and Cobblah *et al.* respectively. The security of these schemes raises some concerns, which we address in detail here. Deng *et al.*'s scheme [28], due to the use of short signatures, was criticized by Gao *et al.* for failing to meet security requirements in practical scenarios [29]. Although Long *et al.*'s CLS scheme provides a security proof [30], the formal proof contains errors, which we have demonstrated in Appendix A. We found that it cannot resolve the DL assumption, rendering its security unguaranteed. Cobblah *et al.*'s scheme [31] is vulnerable to attacks from type I adversaries, and the detailed process is also presented in Appendix B.

(2) **Efficiency Analysis.** As shown in Fig. 2, the computational time of our proposed scheme during the signcryption phase is comparable to that of other schemes. While the efficiency in the verification phase is slightly lower, the unsigncryption phase significantly outperforms other schemes. In terms of total computational overhead, our scheme is only marginally less efficient than Long *et al.* proposed scheme [30], while surpassing most other schemes in overall performance. However, the scheme by Long *et al* [30]. is highly vulnerable to adversarial attacks, which compromises its robustness and security in practical applications.

2) Computation Cost Analysis in Simulation: In addition to analyzing the calculation cost of schemes, we also conduct simulations using the Java Pairing-Based Cryptography (JPBC) library. We compare our scheme with others at each stage. As shown in Fig. 3 and Fig. 5, our computational efficiency is lower than other schemes across multiple algorithm executions. The above results prove that our scheme outperforms others in signcryption and unsigncryption phases, giving us a distinct advantage in data transmission and exchange.

As shown in Fig. 4, compared with other schemes, our proposal consistently maintains a lower verification time cost as the number of executions increases. When the number of executions reaches one hundred, our scheme's time cost is slightly over two hundred milliseconds, which is low among the compared schemes.

In Fig. 6, we observe the experimental time for the Long *et al.*'s scheme [30], which is slightly lower than our proposal in single experiments and when we perform only a few trials.

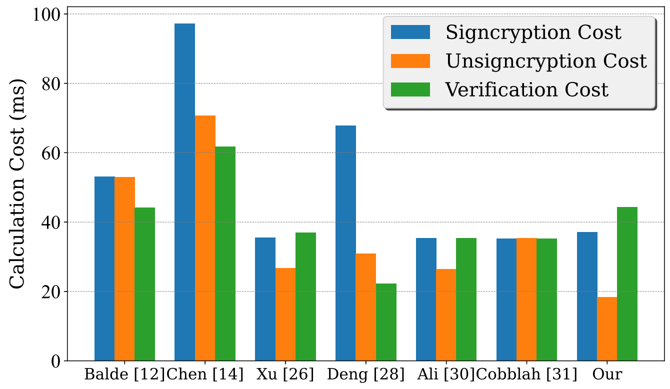


Fig. 2. Theoretical execution time complexity comparison of each scheme

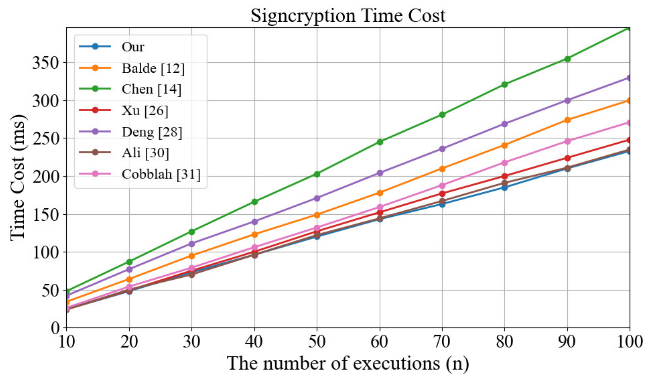


Fig. 3. Performance comparison of signcryption phase execution time

However, as the number of experiments increases, our scheme has a clear calculation cost advantage over Long *et al.*'s scheme [30]. In VANET, where data transfers are frequent, the lower time cost of our scheme across multiple executions becomes particularly beneficial. Therefore, our scheme offers a reduced time cost in repeated operations and provides enhanced security compared to other schemes.

To ensure the authenticity of the experimental results, we uploaded our simulation experiment code to GitHub [1], which allows researchers to view our complete simulation experiments.

[1]https://github.com/Honeyme291/StandardCLSC

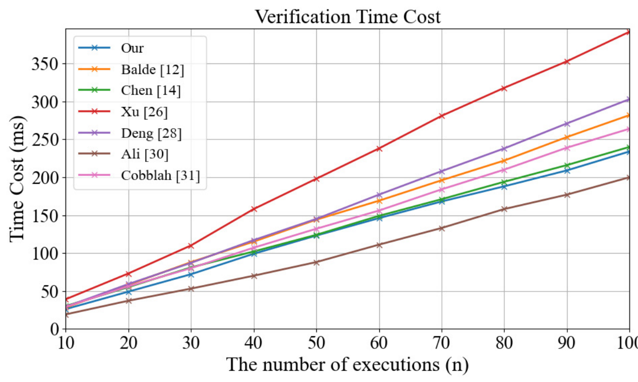


Fig. 4. Performance comparison of verification phase execution time

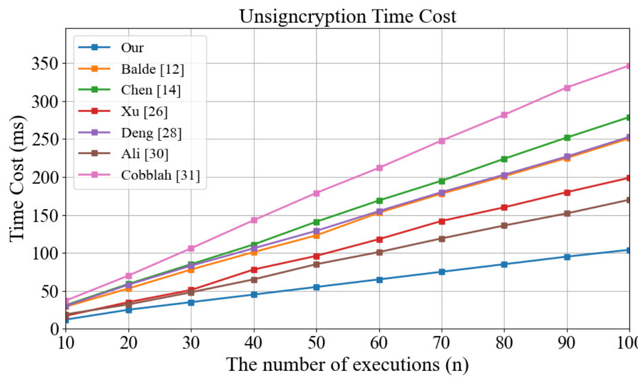


Fig. 5. Performance comparison of unsigncryption phase execution time

V. An Efficient Authentication and Data Transmission Protocol for VANET

In section IV, we presented our scheme and provided a detailed security proof. In this section, to evaluate the practical application, security, and efficiency of our scheme in real-world environments, we apply our proposed protocol to each device in the VANET. We describe our protocol in three main parts: 1) System Framework, 2) Communication and Authentication Protocol, and 3) Enhancing Security: Communication and Authentication Protocol, 4) Experiment Simulation.

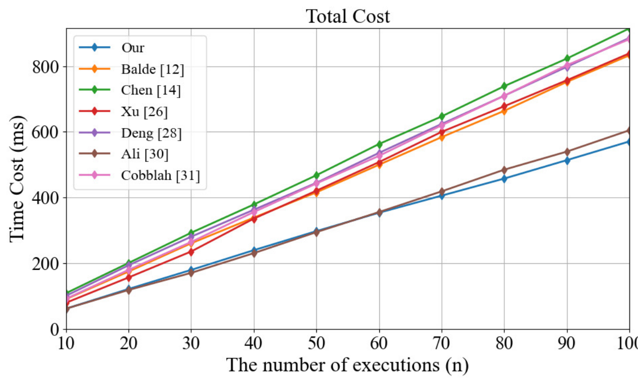


Fig. 6. Performance comparison of the total execution time

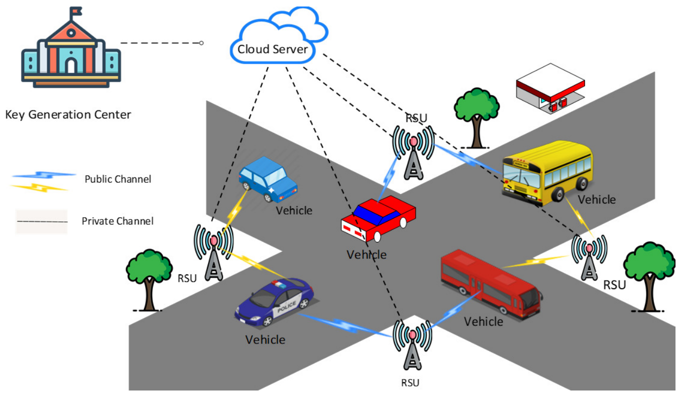


Fig. 7. The system structure of secure VANET

A. System Framework

As shown in Fig. 7, our system architecture consists of four main components: the Key Generation Center, Cloud Server (CS), Roadside Units (RSUs), and Vehicles (OBUs). We introduce the specific functions of each major component in the system below.

(1) *Cloud Server*. CS acts as the core data storage center in VANET, responsible for collecting, storing, and analyzing a large number of data from OBUs and RSUs.

(2) *Key Generation Center*. KGC is a trusted third-party authority, which provides key generation services to RSUs and OBUs.

(3) *Roadside Units*. RSUs are also responsible for verifying messages sent by vehicles and ensuring the integrity and security of messages transmitted by vehicles.

(4) *Vehicles*. OBUs are responsible for transmitting vehicles' real-time data and receiving real-time data from roadside units.

In summary, each component of the system plays a crucial role in ensuring the secure and efficient functioning of VANET. Our proposed efficient and secure data transmission protocol facilitates smooth communication between vehicles in the VANET system, ensuring message security while maintaining timeliness. For the specific process of the transmission authentication protocol, see Section V-B.

B. Enhancing Security: Communication and Authentication Protocol

At this stage, we implement our constructed scheme in the actual scenario of VANET. Fig. 8 vividly illustrates the specific interaction process among each entity in our practical application. Subsequently, we will delineate the detailed process of our protocol.

1) System Initialization: KGC executes the setup algorithm $\mathtt{Setup}(1^k)$ and randomly selects a master secret key $s \in \mathbb{Z}_q^*$ as master secret key and calculate master public key $P_{pub} = sP$. KGC sets five hash functions $H_i = \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $i = 1,2,3,4,5$. KGC also publishes the public parameters $Params = \{G, P, q, P_{pub}, H_1, H_2, H_3, H_4, H_5\}$ and sends to the CS for backup.

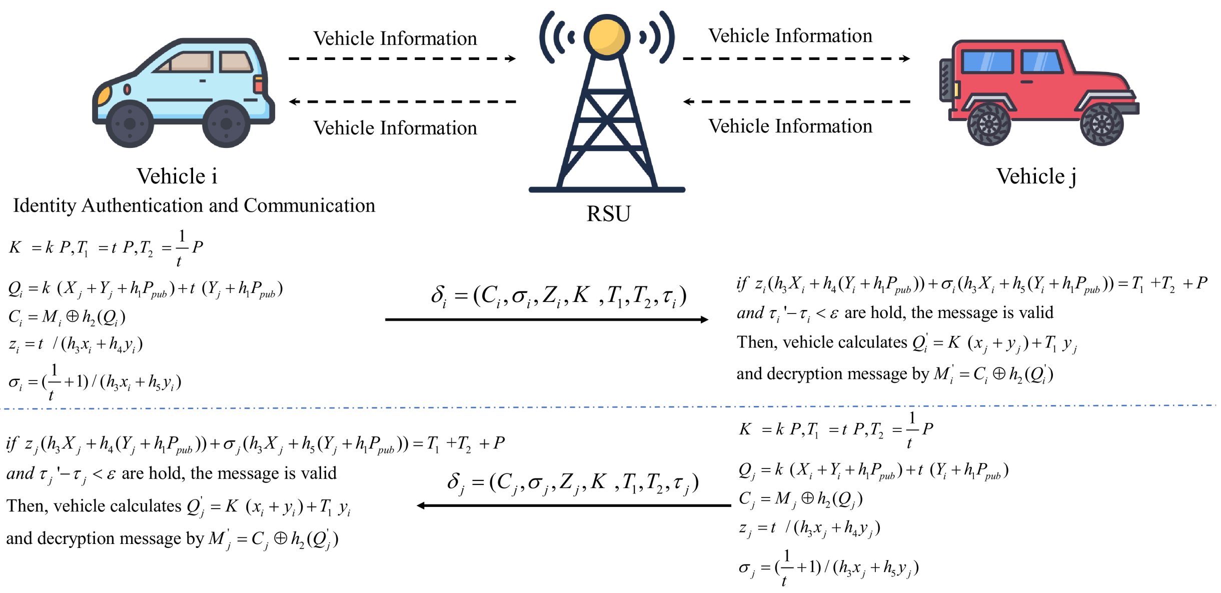

Vehicle Information
Vehicle Information
Vehicle Information
Vehicle Information
Vehicle i
Identity Authentication and Communication
RSU
Vehicle j
$K = kP, T_1 = tP, T_2 = \frac{1}{t}P$
$Q_i = k(X_j + Y_j + h_1 P_{pub}) + t(Y_j + h_1 P_{pub})$
$C_i = M_i \oplus h_2(Q_i)$
$z_i = t/(h_3 x_i + h_4 y_i)$
$\sigma_i = (\frac{1}{t} + 1)/(h_3 x_i + h_5 y_i)$
$\delta_i = (C_i, \sigma_i, Z_i, K, T_1, T_2, \tau_i)$
if $z_i(h_3 X_i + h_4(Y_i + h_1 P_{pub})) + \sigma_i(h_3 X_i + h_5(Y_i + h_1 P_{pub})) = T_1 + T_2 + P$
and $\tau_i' - \tau_i < \varepsilon$ are hold, the message is valid
Then, vehicle calculates $Q_i' = K(x_j + y_j) + T_1 y_j$
and decryption message by $M_i' = C_i \oplus h_2(Q_i')$
if $z_j(h_3 X_j + h_4(Y_j + h_1 P_{pub})) + \sigma_j(h_3 X_j + h_5(Y_j + h_1 P_{pub})) = T_1 + T_2 + P$
and $\tau_j' - \tau_j < \varepsilon$ are hold, the message is valid
Then, vehicle calculates $Q_j' = K(x_i + y_i) + T_1 y_i$
and decryption message by $M_j' = C_j \oplus h_2(Q_j')$
$\delta_j = (C_j, \sigma_j, Z_j, K, T_1, T_2, \tau_j)$
$K = kP, T_1 = tP, T_2 = \frac{1}{t}P$
$Q_j = k(X_i + Y_i + h_1 P_{pub}) + t(Y_i + h_1 P_{pub})$
$C_j = M_j \oplus h_2(Q_j)$
$z_j = t/(h_3 x_j + h_4 y_j)$
$\sigma_j = (\frac{1}{t} + 1)/(h_3 x_j + h_5 y_j)$

Fig. 8. An Efficient and Privacy-Preserving Protocol for Secure Data Transmission

2) Registration: Now, we will show the specific registration operation of our protocol as follows:

(1) The vehicle with identity id_i chooses a random number $x_i \in \mathbb{Z}_q^*$ and calculates $X_i = x_i P$. Then, the vehicle sends its identity tuple (id_i, X_i) to the KGC. Upon receiving the message tuple, KGC selects a random number $u_i \in \mathbb{Z}_q^*$. It then calculates $Y_i = u_i P$ and $y_i = u_i + sh_1$, where $h_1 = H_1(id_i, X_i, Y_i, P_{pub})$. The KGC subsequently sends the partial key tuple (y_i, Y_i) to the vehicle. When the vehicle receives the partial key tuple, it verifies the equation $y_i P = Y_i + h_1 P_{pub}$. If the equation does not hold, it will resend the registration message.

(2) The vehicle with identity id_j chooses a random number $x_j \in \mathbb{Z}_q^*$ and calculates $X_j = x_j P$. Then, the vehicle sends its identity tuple (id_j, X_j) to the KGC. Upon receiving the message tuple, KGC selects a random number $u_j \in \mathbb{Z}_q^*$. It then calculates $Y_j = u_j P$ and $y_j = u_j + sh_1$, where $h_1 = H_1(id_j, X_j, Y_j, P_{pub})$. The KGC subsequently sends the partial key tuple (y_j, Y_j) to the vehicle. When the vehicle receives the partial key tuple, it verifies the equation $y_j P = Y_j + h_1 P_{pub}$. If the equation does not hold, it will resend the registration message.

3) Identity Authentication and Communication: Now, we will show the specific communication operation of our protocol as follows:

(1) The vehicle with identity id_i chooses a random numbers $k, t \in \mathbb{Z}_q^*$, and calculates $K = kP$, $T_1 = tP$ and $T_2 = \frac{1}{t}P$. It then calculates the encryption factor $Q_i = k(X_j + Y_j + h_1 P_{pub}) + t(Y_j + h_1 P_{pub})$. Then, it encrypts the message M_i with $C_i = M_i \oplus H_2(Q_i)$. Finally, it calculates signature $z_i = t/(h_3 x_i + h_4 y_i)$ and $\sigma_i = (\frac{1}{t} + 1)/(h_3 x_i + h_5 y_i)$, where $h_3 = H_3(id_i, id_j, pk_i, pk_j, C_i, K, T_1, T_2)$, $h_4 = H_4(id_i, id_j, pk_i, pk_j, C_i, K, T_1, T_2)$, and $h_5 = H_5(id_i, id_j, pk_i, pk_j, C_i, K, T_1, T_2, z_i)$. Consequently, the vehicle sends the message tuple $\delta_i = (C_i, \sigma_i, K, T_1, T_2, z_i, \tau_i)$ to the another vehicle with identity id_j.

(2) After the vehicle with identity id_j receives the message tuple $\delta_i = (C_i, \sigma_i, K, T_1, T_2, z_i, \tau_i)$, they will verify identity using the equation $z_i(h_3 X_i + h_4(Y_i + h_1 P_{pub})) + \sigma_i(h_3 X_i + h_5(Y_i + h_1 P_{pub})) = T_1 + T_2 + P$ and verify message timeliness by timestamp $\tau_i' - \tau_i \le \varepsilon$, where τ_i' is the message arrival time, and ε is a fixed value. If the equation holds, then the corresponding message tuple is accepted. Then, the vehicle with identity id_j calculates encryption factor $Q_i' = (x_j + y_j)K + T_1 y_j$ and decrypts cipheretxt $M_i = C_i \oplus H_2(Q_i')$. So it can obtain vehicle information that identity is id_i; Otherwise, we discard the message.

The vehicle with identity id_j chooses a random numbers $k, t \in \mathbb{Z}_q^*$, and calculates $K = kP$, $T_1 = tP$ and $T_2 = \frac{1}{t}P$. It then calculates the encryption factor $Q_j = k(X_i + Y_i + h_1 P_{pub}) + t(Y_i + h_1 P_{pub})$. Then, it encrypts the message M_j with $C_j = M_j \oplus H_2(Q_j)$. Finally, it calculates signature $z_j = t/(h_3 x_j + h_4 y_j)$ and $\sigma_j = (\frac{1}{t} + 1)/(h_3 x_j + h_5 y_j)$. Consequently, the vehicle sends the message tuple $\delta_j = (C_j, \sigma_j, K, T_1, T_2, z_j, \tau_j)$ to the another vehicle with identity id_i.

(3) After the vehicle with identity id_i receives the message tuple $\delta_j = (C_j, \sigma_j, K, T_1, T_2, z_j, \tau_j)$, they will verify identity using the equation $z_j(h_3 X_j + h_4(Y_j + h_1 P_{pub})) + \sigma_j(h_3 X_j + h_5(Y_j + h_1 P_{pub})) = T_1 + T_2 + P$ and verify message timeliness by timestamp $\tau_j' - \tau_j \le \varepsilon$. If the equation holds, the vehicle with identity id_i calculates encryption factor $Q_j' = (x_i + y_i)K + T_1 y_i$ and decryption cipheretxt $M_j = C_j \oplus H_2(Q_j')$. So it can obtain vehicle information that

identity is id_j; Otherwise, it discards the message.

C. Ensuring Security in VANET: Requirements and Solutions

In this section, we provide a detailed explanation of the security requirements needed in our environment and how these requirements are ensured within our proposed protocol.

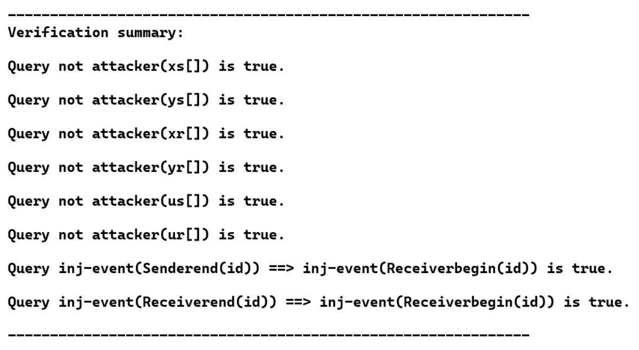
(1) ***Message Integrality***. Data integrity is essential in VANET, as data tampering may lead to poor decisions and endanger driver lives. To ensure data security, our proposed protocol is proved to have message unforgeability in the security proofs in Theorem 3 and 4. This ensures that our messages cannot be tampered with and forged by attackers.

(2) ***Message Privacy***. VNAETs handle substantial amounts of personal and vehicle data, including sensitive information such as location and driving routes. If this data is disclosed, it can lead to privacy violations. Attackers can also use intercepted data for malicious purposes, such as analyzing driving patterns to locate and track vehicles for criminal activities, which would threaten personal safety. In our proposed protocol, we encrypt the data with one-time random numbers and embed difficult problems in our formal proof Theorem 1 and Theorem 2 to ensure the encrypted messages' confidentiality and privacy.

(3) ***Forward Security***. In VANET, keys can be compromised for various reasons, such as insider threats or external attacks. Attackers can then use these keys to recover previously sent messages, potentially exposing sensitive vehicle information and threatening personal data privacy. In our proposed protocol, the session key is generated as a random number, ensuring that even if a vehicle leaks the session key, attackers cannot retrieve valid messages from the signcryption ciphertext, and the privacy and security of vehicle data are protected.

(4) ***Resist replay attacks***. In a replay attack, attackers can intercept and resend valid messages to gain unauthorized control over a vehicle, performing actions like remotely unlocking doors or manipulating vehicle functions. This interference can disrupt communication with navigation and roadside sensing systems, leading to vehicle service interruptions, erroneous operations, and potential traffic accidents, which endanger driver safety. In our proposed protocol, we add a one-time timestamp to a ciphertext message. Here, the timestamp is a unique number. Even if an attacker intercepts the message, they cannot impersonate a legitimate vehicle or RSU to resend it and gain unauthorized vehicle permissions. This can effectively ensure our driving safety in VANET.

(5) ***Resist Man-in-the-Middle attacks***. In an open communication channel, vehicle data—including sensitive information like location and routes—is vulnerable to interception, potentially leading to privacy breaches and targeted attacks. In a man-in-the-middle attack, an adversary can intercept and alter communications between vehicles and RSUs. For instance, an attacker can simulate an RSU to send false traffic signals or warnings, leading drivers to make incorrect decisions and increasing the risk of accidents. Therefore, defending against man-in-the-middle attacks is essential for ensuring driver safety. In our proposed protocol, to prevent man-in-the-middle attacks, the novel construction scheme proposed in this paper uses the CLS scheme to recognize identity authentication.



```
--------------------------------------------------------------
Verification summary:

Query not attacker(xs[]) is true.

Query not attacker(ys[]) is true.

Query not attacker(xr[]) is true.

Query not attacker(yr[]) is true.

Query not attacker(us[]) is true.

Query not attacker(ur[]) is true.

Query inj-event(Senderend(id)) ==> inj-event(Receiverbegin(id)) is true.

Query inj-event(Receiverend(id)) ==> inj-event(Receiverbegin(id)) is true.

--------------------------------------------------------------
```

Fig. 9. Security Protocol Simulation via ProVerif

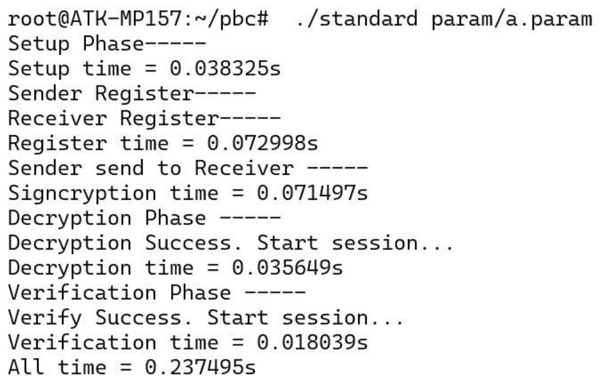
(6) ***Privacy Protection***. The system contains sensitive information in VANET, including vehicle location and destination. If exposed, this data can threaten personal privacy and lead to security issues. In the data transmission of vehicles within our proposed protocol, we use public keys and one-time session keys to signcryption the data. Attackers cannot obtain valid messages from our protocol. Therefore, this ensures that our vehicle data is not leaked to malicious attackers. Only authenticated vehicles can be decrypted.

(7) ***Mutual Authentication***. In the open channels of VANET, without mutual authentication, attackers can spread false information, which can mislead vehicles or cause them to receive outdated or duplicate messages, compromising decision-making and driving safety. Our proposed protocol employs signature-based mutual authentication to verify vehicle identities during data transmission. This prevents attackers from forging the messages we send and spreading false information, thereby protecting the security of VANET.

D. Experiment Simulation

To evaluate the feasibility and effectiveness of the proposed protocol, we conducted two sets of simulations to analyze. The first focused on analyzing the protocol's security using the ProVerif tool, while the second involved implementing the protocol with the Pairing-Based Cryptography (PBC) library to examine its practical feasibility and computational efficiency in real-world scenarios. The simulation results thoroughly validate both the security and practical performance of the protocol.

1) Simulation Protocol Security with Proverif: In our study, we employed the ProVerif tool to conduct a thorough security analysis of the cryptographic protocol under consideration [32]. ProVerif tool is a well-established automated verification tool renowned for its ability to scrutinize cryptographic protocols for confidentiality, authentication, and resilience against various attacks. The verification outcomes, depicted in Fig. 9, reveal that the conditions "inj-event(Senderend(id)) ==> inj-event(Receiverbegin(id))" and "inj-event(Receiverbegin(id)) ==> inj-event(Senderend(id))" hold, proving that the protocol successfully passed all predefined security checks. These results illustrate that no adversary can obtain the user's key or intercept messages, which can ensure the integrity of our data transmission. The comprehen-



```
root@ATK-MP157:~/pbc#  ./standard param/a.param
Setup Phase-----
Setup time = 0.038325s
Sender Register-----
Receiver Register-----
Register time = 0.072998s
Sender send to Receiver -----
Signcryption time = 0.071497s
Decryption Phase -----
Decryption Success. Start session...
Decryption time = 0.035649s
Verification Phase -----
Verify Success. Start session...
Verification time = 0.018039s
All time = 0.237495s
```

Fig. 10. Implementing Simulations with the PBC Library on Raspberry Pi

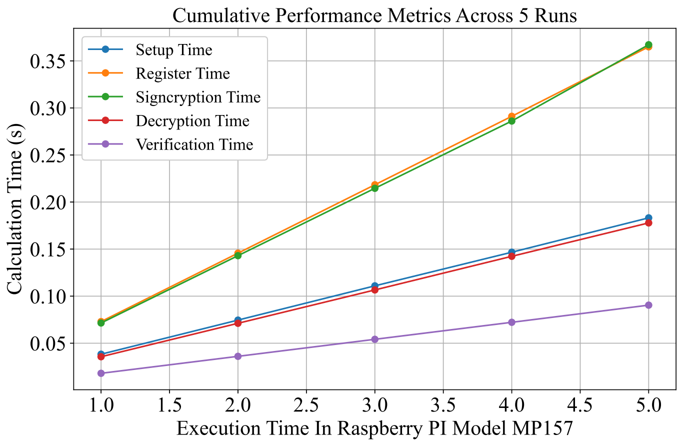


Fig. 11. Evaluating PBC Library Execution Time on Raspberry Pi

sive verification process with the ProVerif tool confirms that the designed cryptographic protocol exhibits a high level of security for authentication and resistance to attacks.

To ensure the authenticity of the experimental results, we uploaded our simulation experiment with the Proverif tool code to GitHub [2], which allows researchers to view our complete simulation experiments.

2) Simulation Computational Efficiency with PBC Library: In this section, we deploy our cryptographic protocol on an experimental platform using PBC [33]. We run the simulation experiment on a Raspberry Pi, and we employ Fig. 10 to show the results. Experimental data demonstrates that our protocol operates reliably on devices with limited calculation resources, achieving an execution time of just 0.23 seconds, which is highly efficient in a real-world environment. The above advantage makes the messages delivered to the driver effective.

In Fig. 11, we repeated each step multiple times, observing a linear relationship in the data. The total execution time for multiple operations does not exceed 0.4 seconds. Therefore, in practical scenarios, the time required to send and receive a message is estimated to be under 0.3 seconds, which is highly acceptable for VANET applications.

To validate the authenticity of our experimental results, We have uploaded our simulation code implemented using the PBC library, along with the dependency library, to GitHub [3].

3) Simulation Protocol Security and Efficiency Conclusion: Overall, the results of the two simulation experiments conducted on our protocol indicate that the proposed protocol offers high efficiency and strong security, improving system performance and ensuring reliable data transmission. Our protocol possesses the two key characteristics of practical application protocols: efficiency and security. Therefore, this solution is highly suitable for VANETs, significantly enhancing communication efficiency and security within these networks.

VI. Conclusion

With the rapid development of vehicular data transmission systems, VANET plays a crucial role in enhancing traffic management. However, the extensive and sensitive data in the above networks introduces significant data transmission risks. In this paper, we propose a novel certificateless signcryption scheme under the standard model to further address the above challenges. This scheme is designed to align closely with real-world environments and is supported by rigorous formal proofs that prove its security.

Additionally, extensive experiments illustrate that our scheme incurs significantly lower calculation costs than comparable schemes. By using the PBC library in a real environment, our scheme is executed in less than 0.3 seconds. Finally, these experiments, coupled with our formal proofs, ensure the security and robustness of our proposed scheme.

VII. Future Work

Although our scheme introduces a novel approach to achieving both security and high-efficiency transmission in CLS protocols, there are still many unresolved issues in the VANET environment. For example, the risk of user private key leakage, which adversaries could exploit, needs to be addressed by tracking the adversary and performing key revocation to prevent the system from being compromised by leaked keys. Additionally, solutions to prevent key leakage and enhance security are still required. Another critical issue in VANETs is ciphertext calculation privacy. A scheme must be proposed that ensures complete privacy from the adversary's perspective, enabling private computation of messages to preserve their confidentiality.

Appendix

In previous work, we highlighted several security issues in CLS schemes. In this section, we briefly discuss the security flaws of these schemes. We focus on analyzing the errors in Long *et al.*'s security proof [30] and the critical steps in Cobblah *et al.*'s scheme [31] that make it vulnerable to adversarial attacks. The detailed process has been uploaded to GitHub for researchers to review, in order to add value to our research.

[2]https://github.com/Honeyme291/standard-Proveif

[3]https://github.com/Honeyme291/standard-PBC

A. Long et al's scheme [30]

System Initialization Phase Setup (1^k): KGC chooses a k-bit prime q to define the tuple $\{F_q, E/F_q, G_q, P\}$, where P generates group G_q. KGC selects a master secret $x \in_R Z_q^*$, computes the system public key $P_{\text{pub}} = xP$, and sets msk $= x$. Five hash functions are chosen: $H_0, H_1, H_2 : \{0,1\}^* \times G \times G \rightarrow Z_q^*$, $h_1 : \{0,1\}^* \rightarrow Z_q^*$, and $h_2 : G_q \rightarrow Z_q^*$. The system parameter Ω is published as $\{F_q, E/F_q, G_q, P, P_{\text{pub}}, H_0, H_1, H_2, h_1, h_2\}$.

KeyGen $(\Omega, O, U, \text{CSS})$: Let $O = \{O_1, \ldots, O_d\}$ represent the data owners. For each participant ID_i (including CSS and user U) as following.

(1) Set-Secret-Value: Participant ID_i selects a secret value $x_i \in_R Z_q^*$ and generates a public key $P_i = x_i P$.

(2) Extract-Partial-Private-Key: ID_i sends (ID_i, P_i) to KGC. KGC picks $r_i \in_R Z_q^*$, computes $R_i = r_i P$ and $e_i = r_i + xH_0(ID_i, R_i, P_i) \mod q$, then sends e_i to ID_i. ID_i verifies the partial key by checking $e_i P = R_i + H_0(ID_i, R_i, P_i)P_{\text{pub}}$.

(3) Set-Private-Key: The full private key of participant ID_i is constructed as $SK_i = (x_i, e_i)$.

(4) Set-Public-Key: The full public key of participant ID_i is constructed as $PK_i = (R_i, P_i)$.

Enc $(\Omega, F, W, \{ID_i\}, ID, \{SK_{O_i}\}, \{PK_{O_i}\}, PK_u)$: Data owners encrypt the EMR set $F = \{f_1, \ldots, f_n\}$ into ciphertexts $C = \{c_1, \ldots, c_n\}$. For multi-signature on ciphertext c_t: Each owner O_i selects $y_{i,t} \in_R Z_q^*$, computes $Y_{i,t} = y_{i,t}P$, broadcasts $Y_{i,t}$ to co-owners, and aggregates $Y_t = \sum_{i=1}^{d} Y_{i,t}$. O_i calculates shared parameters $P_0 = \sum_{i=1}^{d} P_i$ and $Q_0 = \sum_{i=1}^{d} Q_i$. O_i computes $h_t = H_1(c_t, id_t, Y_t, P_0)$ and $h'_t = H_1(c_t, id_t, Y_t, Q_0)$, then generates a signature component $V_{i,t} = y_{i,t} + h_t x_i + h'_t e_i$. Designated clerk O_z aggregates all $V_{i,t}$ into $V_t = \sum_{i=1}^{d} V_{i,t}$ and outputs the signature $\text{sig}_t = (Y_t, V_t)$, validated through $V_t P \overset{?}{=} Y_t + h_t P_0 + h'_t Q_0$.

TrapGen (Ω, SK_u, W'): User U computes aggregated keys $P_0 = \sum_{i=1}^{d} P_i$ and $R_0 = \sum_{i=1}^{d} R_i$. U generates a keyword index for set $W' = \{w'_1, \ldots, w'_l\}$ by setting $T_{W',m+1} = \eta P$ with $\eta \in_R Z_q^*$.

Test $(\Omega, T_{W'}, I, C)$: CSS computes $M_i = \lambda_i Q_C$ using its secret key. CSS verifies the equation:

$$I_{i,1} + \sum_{j=0}^{m} V_{i,j}(T_{W'_j} - x_C T_{W'_{m+1}}) \overset{?}{=} M_i$$

If valid, CSS returns matching ciphertexts $C' = \{c_{k_1}, \ldots, c_{k_s}\}$ and identities $ID' = \{id_{k_1}, \ldots, id_{k_s}\}$; otherwise, outputs $\perp$.

*1) **Security Proof Analysis of Long et al.'s scheme:*** We will now provide a complete security proof by inserting DL Assumption to show that the scheme cannot be proven secure. To keep the proof concise, we will use a Type I adversary as an example.

Initialization. The challenger $\mathcal{C}$ runs the Initialization algorithm to generate the public parameters Ω and the system master secret key x. The challenger $\mathcal{C}$ sends the public parameters Ω to $\mathcal{A}_1$. Here, we select ID^* as the challenge identity.

Public Key Extraction. Upon receiving a public key extraction query for the identity ID_i.

(1) If $ID_i \neq ID^*$, $\mathcal{C}$ runs registration algorithm to generate corresponding key tuple (R_i, P_i) to $\mathcal{A}_1$.

(2) If $ID_i = ID^*$, $\mathcal{C}$ sets $P^* = bP$ (implicit setting $x_i = b$), and chooses $r^* \in \mathbb{Z}_q^*$ to calculate $R^* = r^*P$ and $e^* = r^* + bH_0(ID^*, R^*, P^*)$. The challenger $\mathcal{C}$ returns corresponding tuple (R^*, P^*) to $\mathcal{A}_1$.

Public Key Replacement. When $\mathcal{A}_1$ submits a new public key tuple (R'_i, P'_i) for identity ID_i, $\mathcal{C}$ replaces the existing public key tuple (R_i, P_i) with (R'_i, P'_i).

Forgery Assume that the correct keyword ciphertext and its identity are $C = \{c_1, c_2, \cdots, c_n\}$ and $sig = \{sig_1, \cdots, sig_n\}$, where $sig_t = \{Y_t, V_t\}$, where

$$\phi_1 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{k_\tau}, P_0),$$

$$\phi_2 = \sum_{\tau=1}^{s} H_1(c_{k_\tau}, id_{k_\tau}, Y_{k_\tau}, Q_0).$$

$$\sigma = \sum_{\tau=1}^{s} y_{k_\tau} + \phi_1 \sum_{\tau=1}^{s} x_i + \phi_2^* \sum_{\tau=1}^{s} e_i$$

Therefore, we consider b concluded by $\sum_{\tau=1}^{s} x_i$ to be the solution to the DL hard problem. However, the equation contains two unknowns, leading to an infinite number of possible solutions. This indicates that the hard problem cannot be solved by the challenger. Long *et al*. have claimed that the security is vulnerable.

B. Cobblah et al.'s scheme [31]

Initialization The CVA chooses a Hyperelliptic Curve (HEC) $\mathcal{HCC}$. The divisor of the $\mathcal{HCC}$ is denoted as $\mathcal{D}$. The four one-way and collision-resistant hash functions are chosen, denoted as $(\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4)$. The CVA selects a master secret key $\mathcal{M}_{sk}$ from $\mathbb{Z}_q^*$, where $\mathbb{Z}_q^* = \{1, 2, \ldots, q-1\}$. Then, the CVA computes the master public key as $\mathcal{M}_{pb} = \mathcal{M}_{sk} \cdot \mathcal{D}$. A set of publicly identifying parameters is defined: $\rho = (\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4, \alpha, \mathcal{HCC}, \mathcal{M}_{pb}, \mathcal{D})$.

Secret Value Determination The vehicle device selects a secret value a. It then computes $B = a \cdot \mathcal{D}$ and designates a as the secret value for the device. Finally, the vehicle device returns a and B to the CVA via a secure channel.

Pseudo-Partial-Private-Key Determination The vehicle device selects a value b from the set $\mathbb{Z}_q^*$. It computes $C = b \cdot \mathcal{D}$ and calculates the pseudo-partial-private key as $\mathcal{PP}_k = b + \mathcal{M}_{sk} \cdot \mathcal{H}_1(B, C, \mathcal{U}_{ID}) + \mathcal{H}_1(\mathcal{M}_{sk}, C)$. The vehicle device returns $\mathcal{PP}_k$ and C to the CVA through a secure channel.

Private Key Determination Upon receiving $\mathcal{PP}_k$ and C, the CVA verifies if the equation $\mathcal{PP}_k \cdot \mathcal{D} = C + \mathcal{M}_{pb} \cdot \mathcal{H}_1(B, C, \mathcal{U}_{ID}) + \mathcal{H}_1(a \cdot \mathcal{M}_{pb}, \mathcal{U}_{ID}) \cdot \mathcal{D}$ holds. If the equation holds, the CVA extracts the partial private key as $\mathcal{PP}_{vk} = \mathcal{PP}_k - \mathcal{H}_1(a, \mathcal{M}_{pb}, \mathcal{U}_{ID})$ and computes the private key as $\mathcal{PV}_k = \mathcal{PP}_{vk} + a$.

Public Key Determination The CVA computes the public key as $\mathcal{PB}_k = B + C$. The CVA then returns $\mathcal{PB}_k$ to the vehicle device via an open network.

Content-Signcryption The vehicle device PV selects a value η from the random set $\mathbb{Z}_q^*$. It computes $u = \eta \cdot \mathcal{D}$

and $\gamma_k = \mathcal{H}_1(B, C, \mathcal{U}_{ID})$. The vehicle device PV computes $\lambda = \eta \cdot (\mathcal{PB}_k + \gamma_k \cdot \mathcal{M}_{pb})$ for the consumer vehicle. The vehicle device PV calculates $\sigma = \mathcal{H}_3(\lambda, u, \mathcal{U}_{ID_{cv}})$ and forms $\delta = (u, \sigma, \mathcal{M}, \tau)$. The vehicle device PV calculates $\mu = \mathcal{H}_4(\delta, u, \mathcal{U}_{ID_{pv}})$ and determines $v = \eta - \mu \cdot \mathcal{PV}_{kpv} \mod q$. The signcrypted content is $\zeta = (\delta, \mu, v)$, which is distributed to the consumer vehicle.

Content-Unsigncryption Upon receiving the signcrypted content, the consumer vehicle CV computes $\gamma_{pv} = \mathcal{H}_1(B, C, \mathcal{U}_{ID})$ and $u' = v \cdot \mathcal{D} + \mu \cdot (\mathcal{PB}_{kpv} + \gamma_{pv} \cdot \mathcal{M}_{pb})$. The consumer vehicle CV calculates $\mu' = \mathcal{H}_4(\delta, u', \mathcal{U}_{ID_{pv}})$. If $\mu = \mu'$, the consumer vehicle CV computes $\lambda' = \mathcal{PV}_k \cdot u'$ and calculates $\sigma' = \mathcal{H}_3(\lambda', u, \mathcal{U}_{ID})$. Finally, the consumer vehicle CV derives the plaintext message $\mathcal{M} = \mathcal{DC}_{rpt}(\sigma', u, \delta)$.

Now, we will demonstrate that this CLS scheme is vulnerable to Type I adversary attacks as follows:

When a legitimate vehicle PV sends a message $\zeta = (\delta, \mu, v)$, the adversary $\mathcal{A}_1$ intercepts it. The adversary $\mathcal{A}_1$ then performs the following operations:

(1) ***Public Key Replacement***: Next, $\mathcal{A}_1$ replace PV's public key $\mathcal{PB}_{kpv}$ with $\mathcal{PB}^*_{kpv} = \mathcal{PB}_{kpv} - \frac{1}{\mu}K$.

(2) ***Forgery***: $\mathcal{A}_1$ selects a random number $k \in \mathbb{Z}_q^*$. Afterwords $\mathcal{A}_1$ calculates $K = k \cdot D$ and $v^* = v + k$. $\mathcal{A}_1$ computes $\mu = \mathcal{H}_4(\delta, u, \mathcal{U}_{ID_{pv}})$. Finally, $\mathcal{A}_1$ sends the forged message $\zeta^* = (\delta, \mu, v^*)$ to the consumer vehicle CV.

(3) ***Verify***: Upon receiving the signcrypted content, the consumer vehicle CV calculates $u^* = v^* \cdot \mathcal{D} + \mu \cdot (\mathcal{PB}^*_{kpv} + \gamma_{pv} \cdot \mathcal{M}_{pb})$.

$$\begin{aligned}
u^* &= v^* \cdot \mathcal{D} + \mu \cdot (\mathcal{PB}^*_{kpv} + \mathcal{H}_1(B, C, \mathcal{U}_{ID}) \cdot \mathcal{M}_{pb}) \\
&= \eta D + kD + \mu \cdot (\mathcal{PB}_{kpv} - \frac{1}{\mu}K + \gamma_{pv} \cdot \mathcal{M}_{pb}) \\
&\quad - \mu \cdot (\mathcal{PB}_{kpv} + \gamma_{pv} \cdot \mathcal{M}_{pb}) \\
&= \eta D + K - \mu \frac{1}{\mu} K \\
&= \eta D = u
\end{aligned}$$

Therefore, we can conclude $u^* = u$, the message forged by the adversary $\mathcal{A}_1$ can pass the verification process of the algorithm.

REFERENCES

[1] F John Dian, Reza Vahidnia, and Alireza Rahmati. Wearables and the internet of things (iot), applications, opportunities, and challenges: A survey. *IEEE access*, 8:69200–69211, 2020.

[2] Karam Sallam, Mona Mohamed, and Ali Wagdy Mohamed. Internet of things (iot) in supply chain management: challenges, opportunities, and best practices. *Sustainable Machine Intelligence Journal*, 2:3–1, 2023.

[3] Muhammet Ali Karabulut, A. F. M. Shahen Shah, Haci Ilhan, Al-Sakib Khan Pathan, and Mohammed Atiquzzaman. Inspecting vanet with various critical aspects – a systematic review. *Ad Hoc Networks*, 150:103281, 2023.

[4] Jiguo Li and Yichen Zhang. Cryptanalysis and improvement of batch verification certificateless signature scheme for vanets. *Wireless Personal Communications*, 111:1255–1269, 2020.

[5] Nirupama Ravi, C. Mani Krishna, and Israel Koren. Mix-zones as an effective privacy enhancing technique in mobile and vehicular ad-hoc networks. *ACM Comput. Surv.*, 56(12), October 2024.

[6] Hongyuan Cheng, Zhiyuan Tan, Xianchao Zhang, and Yining Liu. Reliable and fair trustworthiness evaluation protocol for platoon service recommendation system.

[7] Zahraa Sh Alzaidi, Ali A Yassin, and Zaid Ameen Abduljabbar. Main primitive and cryptography tools for authentication in vanet environment: Literature review. *Basrah Researches Sciences*, 50(1):29–29, 2024.

[8] Mahmoud A Shawky, Syed Tariq Shah, Mohammed Abdrabou, Muhammad Usman, Qammer H Abbasi, David Flynn, Muhammad Ali Imran, Shuja Ansari, and Ahmad Taha. How secure are our roads? an in-depth review of authentication in vehicular communications. *Vehicular Communications*, page 100784, 2024.

[9] Yang Lu and Jiguo Li. Provably secure certificate-based signcryption scheme without pairings. *KSII Transactions on Internet and Information Systems (TIIS)*, 8(7):2554–2571, 2014.

[10] Sattam S Al-Riyami and Kenneth G Paterson. Certificateless public key cryptography. In *International conference on the theory and application of cryptology and information security*, pages 452–473. Springer, 2003.

[11] Jiguo Li, Xinyi Huang, Yi Mu, and Wei Wu. Cryptanalysis and improvement of an efficient certificateless signature scheme. *Journal of Communications and Networks*, 10(1):10–17, 2008.

[12] Lansana Balde, Jiashu Zhang, Ahmed Elkhalil, and Otabek Khudayberdiev. A practical and secure certificateless signcryption for privacy-preserving scheme in healthcare management system. In *2021 16th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pages 572–578, 2021.

[13] Rachana Y Patil, Arijit Karati, and Yogesh H Patil. A signcryption with identity-based authentication for secure ehr sharing in iomt utilizing ecc. *International Journal of Information Technology*, pages 1–16, 2024.

[14] Xin Chen, Debiao He, Muhammad Khurram Khan, Min Luo, and Cong Peng. A secure certificateless signcryption scheme without pairing for internet of medical things. *IEEE Internet of Things Journal*, 10(10):9136–9147, 2022.

[15] Yangfan Liang, Yining Liu, Xianchao Zhang, and Gao Liu. Physically secure and privacy-preserving charging authentication framework with data aggregation in vehicle-to-grid networks. *IEEE Transactions on Intelligent Transportation Systems*, 25(11):18831–18846, 2024.

[16] Nabeil Eltayieb, Rashad Elhabob, Yongjian Liao, Fagen Li, and Shijie Zhou. A heterogeneous signcryption scheme with cryptographic reverse firewalls for iot and its application. *Journal of Information Security and Applications*, 83:103763, 2024.

[17] Jiguo Li, Jingjing Zhao, and Yichen Zhang. Certificateless online/offline signcryption scheme. *Security and Communication Networks*, 8(11):1979–1990, 2015.

[18] Yu Han, Yanwei Zhou, Bo Yang, Zhe Xia, and Mingwu Zhang. An efficient and secure lightweight certificateless hybrid signcryption scheme. *IEEE Internet of Things Journal*, 2023.

[19] Isaac Amankona Obiri, Abigail Akosua Addobea, Eric Affum, Jacob Ankamah, and Albert Kofi Kwansah Ansah. A certificateless signcryption with proxy-encryption for securing agricultural data in the cloud. *Journal of Computer Security*, (Preprint):1–39, 2024.

[20] Zirui Qiao, Kui Ma, Yanwei Zhou, Qiliang Yang, Zhe Xia, Bo Yang, and Mingwu Zhang. An anonymous and efficient certificate-based identity authentication protocol for vanet. *IEEE Internet of Things Journal*, 2023.

[21] Xue Li, Renqing Zhu, Dajun Du, Cheng Jiang, and Zhe Zhou. Ecc-based certificateless aggregate signcryption scheme in cyber-physical power systems. *IEEE Systems Journal*, 18(2):893–904, 2024.

[22] Cong Dai and Zhongwei Xu. Pairing-free certificateless aggregate signcryption scheme for vehicular sensor networks. *IEEE Internet of Things Journal*, 10(6):5063–5072, 2023.

[23] Yanwei Zhou, Ran Xu, Zirui Qiao, Bo Yang, Zhe Xia, and Mingwu Zhang. An anonymous and efficient multi-message and multi-receiver certificateless signcryption scheme for vanet. *IEEE Internet of Things Journal*, 2023.

[24] Ming Luo and Yuwei Wan. An enhanced certificateless signcryption in the standard model. *Wireless Personal Communications*, 98:2693–2709, 2018.

[25] Yumin Yuan. Security analysis of an enhanced certificateless signcryption in the standard model. *Wireless Personal Communications*, 112:387–394, 2020.

[26] Guangxia Xu, Jingnan Dong, Chuang Ma, Jun Liu, and Uchani Gutierrez Omar Cliff. A certificateless signcryption mechanism based on blockchain for edge computing. *IEEE Internet of Things Journal*, 10(14):11960–11974, 2023.

[27] Caixue ZHOU. Certificateless signcryption scheme without random oracles. *Chinese Journal of Electronics*, 27(5):1002–1008, 2018.

[28] Lunzhi Deng, Bo Wang, Yan Gao, Zhiwei Chen, and Siwei Li. Certificateless anonymous signcryption scheme with provable security in the standard model suitable for healthcare wireless sensor networks. *IEEE Internet of Things Journal*, 2023.

[29] Yan Gao, Lunzhi Deng, Shuai Feng, Huan Liu, Binhan Li, and Na Wang. Revocable certificate-based broadcast signcryption scheme for edge-enabled iiot. *Information Sciences*, 690:121540, 2025.

[30] Weifeng Long, Jiwen Zeng, Yaying Wu, Yan Gao, and Hui Zhang. A certificateless verifiable bilinear pair-free conjunctive keyword search encryption scheme for iomt. *Electronics*, 13(8):1449, 2024.

[31] Christian Nii Aflah Cobblah, Qi Xia, Jianbin Gao, Hu Xia, Goodlet Akwasi Kusi, and Isaac Amankona Obiri. A secure and lightweight ndn-based vehicular network using edge computing and certificateless signcryption. *IEEE Internet of Things Journal*, 11(16):27043–27057, 2024.

[32] Bruno Blanchet, Ben Smyth, Vincent Cheval, and Marc Sylvestre. Proverif 2.00: automatic cryptographic protocol verifier, user manual and tutorial. *Version from*, 16:05–16, 2018.

[33] Wenying Zheng, Bing Chen, and Debiao He. An adaptive access control scheme based on trust degrees for edge computing. *Computer Standards & Interfaces*, 82:103640, 2022.