Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6677137, 16 pages
https://doi.org/10.1155/2021/6677137

Research Article

Certificateless-Based Anonymous Authentication and Aggregate Signature Scheme for Vehicular Ad Hoc Networks

Xin Ye,[1] **Gencheng Xu**,[2] **Xueli Cheng**,[2] **Yuedi Li**,[1] **and Zhiguang Qin**[1]

[1]*School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan Province, 610054, China*
[2]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan Province, 611731, China*

Correspondence should be addressed to Gencheng Xu; xugencheng@std.uestc.edu.cn

Received 11 December 2020; Revised 18 January 2021; Accepted 11 February 2021; Published 16 March 2021

Academic Editor: Zhuojun Duan

Development of Internet of Vehicles (IoV) has aroused extensive attention in recent years. The IoV requires an efficient communication mode when the application scenarios are complicated. To reduce the verifying time and cut the length of signature, certificateless aggregate signature (CL-AS) is used to achieve improved performance in resource-constrained environments like vehicular ad hoc networks (VANETs), which is able to make it effective in environments constrained by bandwidth and storage. However, in the real application scenarios, messages should be kept untamed, unleashed, and authentic. In addition, most of the proposed schemes tend to be easy to attack by signers or malicious entities which can be called coalition attack. In this paper, we present an improved certificateless-based authentication and aggregate signature scheme, which can properly solve the coalition attack. Moreover, the proposed scheme not only uses pseudonyms in communications to prevent vehicles from revealing their identity but also achieves considerable efficiency compared with state-of-the-art work, certificateless signature (CLS), and CL-AS schemes. Furthermore, it demonstrates that when focused on the existential forgery on adaptive chosen message attack and coalition attack, the proposed schemes can be proved secure. Also, we show that our scheme exceeds existing certification schemes in both computing and communication costs.

1. Introduction

With the rapid development of communication technology, various vehicles with powerful smart devices can communicate with each other. Therefore, such a novel application has aroused extensive interest in the society. This kind of application is commonly referred to as vehicle ad hoc networks (VANETs), which can provide guarantee for the distance between vehicles and reduce the probability of vehicle collision accidents, help car drivers navigate in real time, and improve the efficiency of traffic operation by communicating with other vehicles and network systems [1].

Although VANETs have a lot of merits, it has a long way to achieve a wide application. One of the obstacles is that the privacy is violated. Without proper privacy protection, malicious adversaries can collect vehicle information, such as routes or status, to perform attacks. Fortunately, using pseudonyms in communications can avoid this problem. Then, the vehicle can communicate with each other or with roadside unit (RSU) using a pseudonym, and no one can obtain the true identity of the vehicle except for the trusted authority (TA). Even if the messages between the vehicles and the RSUs are collected by hackers, it will not reveal identity privacy. VANETs have other problems such as privacy issues and being vulnerable to attack.

Recently, some novel schemes and algorithms are proposed to solve these problems. Lin et al. [2] proposed a blockchain-based protocol to reduce the verification cost and storage cost for vehicles. Kumar et al. [3] proposed an efficient scheme using path signature to resist Sybil attack. Jiang et al. [4] proposed an anonymous authentication scheme (AAAS) in VANETs, which adopts group signature mechanism to provide more efficient anonymous authentication service for vehicles. Zheng et al. [5] demonstrated a

certificateless group signature anonymous authentication scheme for VANETs, which shortens the length of the signature and improves the efficiency of the signature. Among various schemes, we find that Kamil et al.'s scheme [6] has a significant efficiency. However, we find that the scheme cannot resist coalition attack which is launched by two collusive vehicles. For example, two vehicles can maliciously exchange their locations to generate their signatures which can be verified successfully so that they can hide their real locations which may lead to serious consequences. The detailed description and analysis are shown in Subsection 4.3. We make the RSU both the aggregator and the verifier and add a random list to properly solve the problem. Our main contributions in this paper are as follows:

(i) Prove that Kamil et al.'s schemes are not secure enough to defend against attacks from malicious vehicles and propose a solution to settle the problem

(ii) Propose an improved certificateless-based authentication and aggregate signature scheme in VANETs, and prove that the scheme can perfectly resist the coalition attacks and its correctness

(iii) Use the efficiency analysis and simulation to show the superiority of our scheme in efficiency and practicality

The rest of this paper is organized as follows. In Section 2, we discuss related works of CLS and CL-AS schemes in VANETs. In Section 3, we describe related concepts and models. In Section 4, we analyze Kamil et al.'s scheme and prove that the scheme cannot resist the coalition attack. We propose our proposed scheme in Section 5 in detail. Experiments and results analysis are described in Section 6. We conclude this paper in Section 7.

2. Related Works

To settle the problem of security and some privacy requirements in VANETs, a number of professors and scholars [7–9] proposed a kind of new scheme called Public Key Infrastructure-based (PKI-based) authentication schemes. In their schemes, they either tried to make vehicles compute more to verify the signatures from other vehicles or assume that there exists a trusted certificate authority to issue and maintain certificates of various vehicles. However, the assumption may be unrealistic because a single node cannot afford the oceans of calculation.

Later, a new kind of signature scheme called identity-based signature (IBS) scheme is widely discussed. For example, Liu et al. [10] proposed an IBS scheme which can take the user's identity as the public key, and the private key is generated by public key generation PKG, which can reduce a single node's burden. However, IBS has inherent problems about key escrow which is generated by user's identity.

In Al-Riyami and Paterson's scheme [11], they firstly introduce the certificateless public key cryptography. In recent years, a lot of researches on CLS and CL-AS schemes with bilinear pairing have been carried on by relevant researchers [12–14]. In their schemes, key generation center (KGC) uses its master key and the user's identity information to calculate a part of the private key and send it to the user, whereafter the user combines part of the private key and his/her secret value together to generate the user's real private key which can protect the user's privacy and make the system secure. The above scheme uses the bilinear pairing which costs relatively large computation.

The elliptic curve cryptography is chosen to use in the CLS and CL-AS because of its high efficiency. In Xie et al.'s scheme [15], they proposed rigorous security proof that shows the scheme is able to resist various malicious attacks and ensure privacy protection. In the field of health care, Du et al. [16] proposed a CLAS scheme with high efficiency and low latency which can be more suitable to apply to the field of healthcare. In 2018, Cui et al. [17] demonstrated their novel CLS and CL-AS scheme with ECC, which significantly reduces computing time during sign and verification process. Kamil et al. [6] declared that the scheme proposed by Cui et al. is not secured against the signature forgery attack, and they advanced an improved signature scheme for VANETs. They claimed that their proposed scheme can address all the needs of VANETs about security and privacy. However, we will demonstrate and prove that their scheme cannot resist coalition attacks and our improved scheme can resist the attack and achieves a better performance.

3. Preliminaries

3.1. Elliptic Curve Cryptography. As widely used in the cryptographic, the elliptic curve cryptography is an excellent algorithm which has an extremely high efficiency and a relatively excellent security. It can use much fewer bits to encrypt messages of the same length than the RSA algorithm in the field of public key cryptography. Because of its fewer calculation parameters, shorter bond length, and less time cost, the elliptic curve cryptography can be perfectly applied to application scenarios of VANETs. We will give the following three definitions to describe the elliptic curve cryptography.

Definition 1 (Elliptic curve definition). Our scheme uses an elliptical encryption algorithm with 160 bits. Assume that F_q is a finite field of the module q, where q is a large prime number. The elliptic curve over a finite field F_q can be defined as follows: $E : y^2 = x^3 + ax + b \pmod{p}$, where $a, b, x, y \in F_q$ and $\Delta = 4a^3 + 27b^2 \neq 0 \pmod{p}$.

Definition 2 (Addition of elliptic curves). Assume that $P = (x_1, y_1) \in E$, where P is a point of the elliptic curve E and $-P = (x_1, -y_1) \pmod{p}$ is the negative point of P. Suppose $Q = (x_2, y_2) \in E$, $Q \neq -P$; we can define a line l passes through P and Q, and intersects the elliptic curve at a point $R' = (x_3, y_3)$, The symmetrical point about the x-axis with R' is $R = (x_3, -y_3)$; then we can define $R = P + Q$. In addition, scalar multiplication operation on the elliptic curve can be

described as follows:

$$k \cdot P = P + P + P + \cdots + P \quad \left(k \in Z_q^*\right). \tag{1}$$

Definition 3 (Elliptic curve discrete logarithm problem). Assume that P_1 is a point on the elliptic curve E on the finite field F_q, and select a random number $k \in Z_q^*$. Then, we can calculate $P_2 = k \cdot P_1$. In this case, there is the feasibility of the calculation of P_2 according to Definition 2. According to the elliptic curve discrete logarithm problem (ECDLP), however, it is hardly possible to get k according the above equation.

3.2. Forking Lemma

Definition 4 (Forking lemma [18]). Suppose that A is a probabilistic polynomial time turing machine, and its input includes public data. We use Q and R to symbolize the number of queries that A can ask to the random oracle and the number of queries that A can ask to the signer, respectively. Suppose that over a period of time T, A can generate a legitimate signature $(m, \sigma_1, h, \sigma_2)$ within probability $\varepsilon \geq 10(R + 1)(R + Q)/2^k$. If someone do not know the private key, but successfully forge the signature (σ_1, h, σ_2) with an indistinguishable distribution probability, then we can imagine a machine, which can get the secret information from the machine and obtain and replace the interaction with the signer by simulation. Eventually, it can generate two legitimate signatures $(m, \sigma_1, h, \sigma_2)$ and $(m, \sigma_1, h', \sigma_2{}')$ such that $h \neq h'$ in expected time $T' \geq 120686QT/\varepsilon$.

3.3. Certificateless (Aggregate) Signature Scheme. Generally, a certificateless signature (CLS) scheme and a certificateless aggregate signature (CL-AS) scheme consist of the following seven algorithms.

(1) Setup: the KGC and TA will execute this probabilistic algorithm, which needs a security parameter λ, then generates a elliptic curve E, public keys PK_{TA} and PK_{KGC}, and master secrets key α, β, respectively, then publishes a number of system parameters which is used for ensuring the system in order.

(2) ParitialPrivateKeyGeneration: in this algorithm, firstly, the entity V_i transmits a tuple which includes its real identity and partial pseudo identity to TA. Then TA sends a whole pseudo identity to KGC with calculation. Eventually, KGC transmits the paitial private key to entity V_i in a secure channel.

(3) VehicleKeyGeneration: the entity V_i selects random $\rho_i \in Z_q^*$ as its secret key and calculates its public key PK_{V_i}.

(4) IndividualSign: this algorithm is used by each entity V_i; after generating a message m_i, the entity V_i tries to calculate a set of varieables. Then it sends the signature σ to the verifier.

(5) IndividualVerify: this algorithm is executed by the verifier such as RSU. When receiving input including signature σ, pseudo identity PID_i and current time T_{cur}, the RSU will check the time validity firstly. Then the algorithm will output true if the signature is valid or false otherwise.

(6) AggregateSign: in this algorithm, generally the aggregate signature generator is RSU in our system. For an aggregating set V of n entities $V_1, V_2, \cdots, V_n$, the pseudo identity PID_i of each vehicle V_i as list PID, the corresponding public key PK_{V_i} of V_i, and message signature tuples $((m_1, \sigma_1), (m_2, \sigma_2), \cdots, (m_n, \sigma_n))$ from V_i, respectively. The aggregate signature generator will generate signature σ; then it will transmit the tuple including the signature, the list PID, and time list T to the verifier.

(7) AggregateVerify: in general, this algorithm is executed by another RSU. It takes an aggregating set V of n entities $\{V_1, V_2, \cdots, V_n\}$, the pseudo identity PID_i of each entity V_i. The verifier will check the time validity for each entity firstly. Then it will output true if the signature is valid or false otherwise.

3.4. Security Model. In this section, we will demonstrate the security model of CLS and CL-AS schemes. We consider two different types of adversaries: Type $1\mathcal{A}_1$ and Type $2\mathcal{A}_2$. To be specific, adversary $\mathcal{A}_1$ is able to replace a user's public key or private key but cannot access or even replace the master secret key of KGC. And adversary $\mathcal{A}_2$ is able to access the master secret key of KGC, which can be called an internal attacker. However, it cannot replace or access the public key of a certain user.

Generally, we use two games to model the security of CLS and CL-AS schemes, which is played between an adversary $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$ and a challenger $\mathcal{C}$. $\mathcal{A}$ can access five oracles to get what he needs. The details are as follows:

(1) GenerateUser: given a user's ID PID_i and request for its public key PK_{V_i}, $\mathcal{C}$ returns the public key PK_{V_i} of PID_i.

(2) RevealPartialPrivateKey: given a user's pseudo identity PID_i, $\mathcal{C}$ outputs the corresponding partial secret key PPK_i.

(3) RevealSecretKey: given a user's pseudo identity PID_i, $\mathcal{C}$ submits the user's secret key ρ_i.

(4) ReplaceKey: given a user's pseudo identity PID_i and the public key $PK_{V_i}^*$, $\mathcal{C}$ will replace the public key PK_{V_i} with $PK_{V_i}^*$.

(5) Sign: given a message $m_i \in \{0, 1\}^*$, $\mathcal{C}$ uses the algorithm to generate a signature σ_i corresponding to user PID_i on message m_i and submits it to $\mathcal{A}$.

We construct the following two games, Game I and Game II, for our schemes:

(Game I) A Type 1 adversary $\mathscr{A}_1$ and a challenger $\mathscr{C}$ will try to play the game as follows:

Step 1. $\mathscr{C}$ runs the Setup algorithm to generate a master secret key β, a list of system parameters, and the system public key $\mathrm{PK}_{\mathrm{KGC}}$. It then sends the system parameters to $\mathscr{A}_1$ and keeps β secret.

Step 2. $\mathscr{A}_1$ queries the GenerateUser, RevealPartialSecretKey, RevealSecretKey, and Sign oracles in order.

Step 3. $\mathscr{A}_1$ generates the corresponding public key $\mathrm{PK}_{V_i}^*$ and a signature σ_i^* of a user with identity PID_i^*.

$\mathscr{A}_1$ will win the game if the following conditions are met:

(i) It neither uses PID_i^* to access the RevealPartialSecretKey query nor obtains the partial private key

(ii) σ^* is a valid signature of the user with the identity PID_i^* and the corresponding public key $\mathrm{PK}_{V_i}^*$

(iii) It never uses $(\mathrm{PID}_i^*, m_i^*)$ to query the Sign oracle

(Game II) A Type 2 adversary $\mathscr{A}_2$ and a challenger $\mathscr{C}$ will try to play the game as follows:

Step 1. $\mathscr{C}$ runs the Setup algorithm to generate a master secret key β, a list of system parameters, and the system public key PK_{TA}. It then sends the system parameters, β, and $\mathrm{PK}_{\mathrm{TA}}$ to $\mathscr{A}_2$.

Step 2. $\mathscr{A}_2$ queries the GenerateUser, RevealPartialSecretKey, RevealSecretKey, and Sign oracles in order.

Step 3. $\mathscr{A}_2$ generates the corresponding public key $\mathrm{PK}_{V_i}^*$ and a signature σ_i^* of a user with identity PID_i^*.

$\mathscr{A}_2$ will win the game if the following conditions are satisfied:

(i) It never use PID_i^* to access the RevealSecretKey or ReplaceKey query to obtain the partial private key

(ii) σ^* is a valid signature of user with identity PID_i^* and the corresponding public key $\mathrm{PK}_{V_i}^*$

(iii) It never uses $(\mathrm{PID}_i^*, m_i^*)$ to query the Sign oracle

Definition 5. The CLS scheme is provably secure, if neither polynomial time adversary $\mathscr{A}_1$ or $\mathscr{A}_2$ is able to win Game I and Game II, respectively with a non-negligible advantage.

We construct the following two games, Game III and Game IV, for our CL-AS scheme.

(Game III) A Type 1 adversary $\mathscr{A}_1$ and a challenger $\mathscr{C}$ will try to play the game as follows:

Step 1. $\mathscr{C}$ runs the Setup algorithm to generate the master secret key β, system parameter, and the system public key $\mathrm{PK}_{\mathrm{TA}}$. It then sends the system parameter to $\mathscr{A}_1$ and keeps β secret.

Step 2. $\mathscr{A}_1$ queries the GenerateUser, RevealPartialSecretKey, RevealSecretKey, and Sign oracles in order.

Step 3. $\mathscr{A}_1$ outputs an aggregate signature σ^* of n users with identity $\mathrm{PID}^* = \{\mathrm{PID}_1^*, \mathrm{PID}_2^*, \cdots, \mathrm{PID}_n^*\}$ and the corresponding public key $\mathrm{PK}_V^* = \{\mathrm{PK}_{V_1}^*, \mathrm{PK}_{V_2}^*, \cdots, \mathrm{PK}_{V_n}^*\}$ on messages $m^* = \{m_1^*, m_2^*, \cdots, m_n^*\}$.

$\mathscr{A}_1$ wins the game if the following conditions are satisfied:

(i) At least one of the identities has not been submitted to the RevealPartialSecretKey query to obtain the partial secret key

(ii) σ^* is a valid signature on n messages $M^* = \{m_1^*, m_2^*, \cdots, m_n^*\}$ of n users with identities $\mathrm{PID}^* = \{\mathrm{PID}_1^*, \mathrm{PID}_2^*, \cdots, \mathrm{PID}_n^*\}$ and the corresponding public key $\mathrm{PK}_V^* = \{\mathrm{PK}_{V_1}^*, \mathrm{PK}_{V_2}^*, \cdots, \mathrm{PK}_{V_n}^*\}$.

(iii) It never uses $(\mathrm{PID}_i^*, m_i^*)$ to query the Sign oracle

(Game IV) A Type 2 adversary $\mathscr{A}_2$ and a challenger $\mathscr{C}$ will try to play the game as follows:

Step 1. $\mathscr{C}$ runs the Setup algorithm to generate the master secret key β, system parameter, and the system public key $\mathrm{PK}_{\mathrm{TA}}$. It then sends the system parameter, β, $\mathrm{PK}_{\mathrm{TA}}$ to $\mathscr{A}_2$.

Step 2. $\mathscr{A}_2$ queries the GenerateUser, RevealPartialSecretKey, RevealSecretKey, and Sign oracles in order.

Step 3. $\mathscr{A}_2$ outputs an aggregate signature σ^* of n users with identity $\mathrm{ID}^* = \{\mathrm{ID}_1^*, \mathrm{ID}_2^*, \cdots, \mathrm{ID}_n^*\}$ and the corresponding public key $\mathrm{PK}_V^* = \{\mathrm{PK}_{V_1}^*, \mathrm{PK}_{V_2}^*, \cdots, \mathrm{PK}_{V_n}^*\}$ on messages $M^* = \{m_1^*, m_2^*, \cdots, m_n^*\}$.

$\mathscr{A}_2$ will win the game if the following conditions are satisfied:

(i) It has not used all of the identities to access the RevealPartialSecretKey query to obtain the partial private key.

(ii) σ^* is a legitimate signature on n messages $m^* = \{m_1^*, m_2^*, \cdots, m_n^*\}$ of n users with identities $\mathrm{PID}^* = \{\mathrm{PID}_1^*, \mathrm{PID}_2^*, \cdots, \mathrm{PID}_n^*\}$ and the corresponding public key $\mathrm{PK}_V^* = \{\mathrm{PK}_{V_1}^*, \mathrm{PK}_{V_2}^*, \cdots, \mathrm{PK}_{V_n}^*\}$.

(iii) It never uses (ID_i^*, m_i^*) to query the Sign oracle

Definition 6. The CL-AS scheme is provably secure, if neither polynomial time adversary $\mathcal{A}_1$ or $\mathcal{A}_2$ is able to win Game III and Game IV, respectively, with a nonnegligible advantage.

4. Overview of Kamil et al.'s CLS and CL-AS Scheme

In the scheme proposed by Kamil et al. [6], there mainly exist four entities including TA, regional transport management authority (RTMA), which is a trusted party responsible for partial secret key generation, RSU, and vehicle. The scheme is reviewed as follows:

4.1. Overview of Kamil et al.'s CLS Scheme

(1) Setup: the TA selects a security parameter k, two secure primes p and q, an elliptic curve E which can be defined by the equation $y^2 = x^3 + ax + b \bmod p$, where $a, b \in F_q$, a generator P with order q of additive group G consisting of all the points on E, and five hash functions, h_0, h_1, h_2, h_3, and h_4. Then, it picks $x \in \mathbb{Z}_q^*$ as its master secret key and calculates its public key P_{pub}. Also, TA defines a time-function $f(t_i)$, where t_i is the current time. TA publishes the param $= \{p, q, a, b, P, P_{\text{pub}}, h_0, h_1, h_2, h_3, h_4, f(t_i)\}$.

(2) UserRegistration: the RTMA executes the following algorithm to register a vehicle with an identity ID_k. Firstly, vehicle sends its identity ID_k to the RTMA. Then RTMA randomly selects $\hbar_{1,k} \in \mathbb{Z}_q^*$ and calculates hash chain set $\hbar_{y,k} = \{\hbar_{2,k}, \hbar_{3,k}, \cdots, \hbar_{n,k}\}$, $1 \le y \le n$, where $\hbar_{y,k} = h_0(\hbar_{y-1,k})$.

(3) PartialSecretKeyGeneration: after receiving param and a vehicle with identity ID_k, RTMA runs as follows:

Step 1. The RTMA generates its public key $\text{PK}_{\text{RTMA}} = s \cdot P$, where secret key $s \in \mathbb{Z}_q^*$ is randomly selected.

Step 2. Calculate $\alpha_k = h_2(\text{param} \| P_{\text{pub}} \| \text{ID}_k)$, $\beta_k = h_2(\text{ID}_k \| \text{ID}_{\text{RTMA}} \| s \| \nabla)$, $t = h_2(\nabla)$, and $\xi_k = h_2(\text{ID}_k \| \text{PK}_{\text{RTMA}})$.

Step 3. Compute $A_k = t\beta_k \cdot P$ and $x_k = t\beta_k + \xi_k \alpha_k s$.

Step 4. Publish PK_{RTMA}, send $(\text{PSK}_k = (A_k, x_k), \hbar_{y,k})$ to the vehicle and $(\hbar_{1,k}, \text{ID}_k)$ to TA.

(4) PseudonymGeneration: after receiving the tuple $(\hbar_{y,k}, (A_k, x_k))$ from the RTMA, the vehicle executes the following algorithm:

Step 1. Compute $\alpha_k = h_2(\text{param} \| P_{\text{pub}} \| ID_k)$, $t = h_2(\nabla)$ and $\xi_k = h_2(\text{ID}_k \| \text{PK}_{\text{RTMA}})$.

Step 2. Check PSK_k is valid or not with the equation holds.

$$x_k \cdot P = A_k + \xi_k \alpha_k \cdot \text{PK}_{\text{RTMA}}. \tag{2}$$

Step 3. Compute its pseudonym set as $\{\text{PID}_{1,k}, \text{PID}_{2,k}, \cdots, \text{PID}_{n,k}\}$ at timeslot ts_{cur}, where $\text{PID}_{y,k} = h_1(\text{ID}_k \| \hbar_{y,k} \| \nabla \| ts_{\text{cur}})$.

(5) UserKeyGeneration: vehicle with ID_k uses the algorithm to generate its private key:

Step 1. Choose $a_k, r_k^1, r_k^2 \in \mathbb{Z}_q^*$ in random.

Step 2. Calculate $\text{SK}_k^1 = h_3(r_k^1 \| A_k \| \text{PID}_{y,k})$ and $\text{SK}_k^2 = h_3(r_k^2 \| x_k \| \text{PID}_{y,k})$.

Step 3. Output $\text{SK}_k = a_k(\text{SK}_k^1 + \text{SK}_k^2)$ and $\text{PK}_k = \text{SK}_k \cdot P$ as its private and public keys, respectively.

(6) Sign: after receiving param, PSK_k, SK_k, and PK_k, a vehicle with pseudo identity $\text{PID}_{y,k}$ can sign on a message m_k as follows:

Step 1. Randomly pick $d_k \in \mathbb{Z}_q^*$ and calculate $\omega_k = \xi_k \alpha_k$.

Step 2. Calculate $v_k = h_4(\text{PID}_{y,k} \| m_k \| \text{SK}_k^1 \| \text{SK}_k^2 \| T_k)$, $h_k = h_4(\text{PID}_{y,k} \| m_k \| \omega_k \| \text{PK}_k \| \text{PK}_{\text{RTMA}} \| P_{\text{pub}} \| T_k)$, and $\delta_k = h_4(m_k \| PK_k \| \nabla \| T_k)$.

Step 3. Calculate $y_k = d_k v_k$, $\Omega_k = y_k \cdot P$, $R_k = \delta_k \cdot \text{PK}_k + h_k \cdot A_k + \Omega_k$, and $\mathcal{V}_k = \delta_k \text{SK}_k + h_k x_k + d_k v_k$.

Step 4. Output signature $\sigma_k = (R_k, \mathcal{V}_k)$ on message m_K and transmits $(\text{PID}_{y,k}, m_k, \text{PK}_k, \omega_k, \sigma_k, T_k)$, where T_k is the current timestamp.

(7) Verify: after receiving the tuple $(\text{PID}_{y,k}, m_k, \text{PK}_k, \omega_k, \sigma_k, T_k)$, verifier can use the algorithm to verify any signature with following steps:

Step 1. Check whether the time delay equation holds. If it holds, then T_k is valid and it will accept the signature; otherwise, it will reject it.

Step 2. Calculate $h_k = h_4(\text{PID}_{y,k} \| m_k \| \omega_k \| \text{PK}_k \| \text{PK}_{\text{RTMA}} \| P_{\text{pub}} \| T_k)$.

Step 3. Check whether the following equation holds.

$$\mathcal{V}_k \cdot P = R_k + h_k \omega_k \cdot \text{PK}_{\text{RTMA}}, \tag{3}$$

if this equation holds, then the signature is valid; otherwise, it will be discarded.

4.2. Overview of Kamil et al.'s CL-AS Scheme. The Setup, UserRegistration, PartialPrivateKeyGeneration, PseudonymGeneration, VehicleKeyGeneration, Sign, and Verify algorithms are the same as the above CLS scheme. In addition, the Aggregate and AggregateVerify algorithms are described as follows:

(1) Aggregate: in general, the roadside unit (RSU) acts as the aggregator. When receiving n certificateless signatures $\{\sigma_1 = (R_1, \mathscr{V}_1), \sigma_2 = (R_2, \mathscr{V}_2), \cdots, \sigma_n = (R_n, \mathscr{V}_n)\}$ on messages $\{(m_1 \| T_1), (m_2 \| T_2), \cdots, (m_n \| T_n)\}$ from n pseudo identities $\{\mathrm{PID}_{y,1}, \mathrm{PID}_{y,2}, \cdots, \mathrm{PID}_{y,n}\}$ under the state information ∇, the RSU calculates $\mathscr{V} = \sum_{k=1}^{n} \boxtimes \mathscr{V}_k$ and $R = \sum_{k=1}^{n} \boxtimes R_k$, then outputs an aggregate certificateless signature $\sigma_T = (R, \mathscr{V})$.

(2) AggregateVerify: generally another RSU or AS acts as the verifier. When receiving a certificateless aggregate signature $\sigma_T = (R, \mathscr{V})$ signed by n vehicles. Then it will run as follows:

Step 1. Check whether the timestamp T_k is valid, if not, it aborts, and if it holds, it runs the following steps.

Step 2. Compute $h_k = h_4(\mathrm{PID}_{y,k} \| m_k \| \omega_k \| \mathrm{PK}_k \| \mathrm{PK}_{\mathrm{RTMA}} \| P_{\mathrm{pub}} \| T_k)$.

Step 3. Check whether the following equation holds.

$$\mathscr{V} \cdot P = R + \sum_{k=1}^{n} h_k \omega_k \cdot \mathrm{PK}_{\mathrm{RTMA}}, \tag{4}$$

if it holds, it receives all the signatures; otherwise, the signature is rejected.

4.3. Cryptanalysis of Kamil et al.'s CL-AS Scheme. The security problem in the scheme proposed by Kamil et al. [6] mainly lies in the coalition attack, which is a kind of attack by a number of collusive vehicles.

As is described in Figure 1, in the coalition attack, two or more vehicles secretly change a part of their messages such as locations to hide their real locations and routes since the RSU (verifier) receives the exchanged signature. Then something of the collusive vehicles is exchanged officially. Which will definitely harm the system and even worse cause a serious accident.

We describe the coalition attack on Kamil et al.'s CL-AS scheme to illustrate its security flaws.

Assume that two users $\{U_1, U_2\}$ have pseudonym $\{\mathrm{PID}_{y,1}, \mathrm{PID}_{y,2}\}$ and message $\{m_1, m_2\}$, respectively. We show that two users can cooperate to generate valid aggregate signatures even if their individual signature is invalid. Two users can implement the coalition attack by executing the following algorithms.

Step 1. The user U_i $(i \in \{1, 2\})$ randomly picks $d_i \in \mathbb{Z}_q^*$ and calculates $\omega_i = \xi_i \alpha_i$.

Step 2. Calculate $\beta_i = h_2(\mathrm{ID}_i \| \mathrm{ID}_{\mathrm{RTMA}} \| s \| \nabla)$, $t = h_2(\nabla)$, $\xi_k = h_2(\mathrm{ID}_i \| \mathrm{PK}_{\mathrm{RTMA}})$, $x_i = t\beta_i + \xi_i \alpha_i s$, $\nu_i = h_4(\mathrm{PID}_{y,i} \| m_i \| SK_i^1 \| \mathrm{SK}_i^2 \| T_i)$, $h_i = h_4(\mathrm{PID}_{y,i} \| m_i \| \omega_i \| \mathrm{PK}_i \| \mathrm{PK}_{\mathrm{RTMA}} \| P_{\mathrm{pub}} \| T_i)$, $\delta_i = h_4(m_i \| \mathrm{PK}_i \| \nabla \| T_i)$, $y_i = d_i \nu_i$, $\Omega = y_i \cdot P$, and $R_i = \delta_i \cdot \mathrm{PK}_i + h_i \cdot A_i + \Omega$.

Step 3. Then, U_1 sends $h_1 x_1$ to U_2; likewise, U_2 sends $h_2 x_2$ to U_1 in a secure channel. Then U_1 calculates $\mathscr{V}_1 = \delta_1 \mathrm{SK}_1 + h_2 x_2 + d_1 \nu_1$; likewise, U_2 calculates $\mathscr{V}_2 = \delta_2 \mathrm{SK}_2 + h_1 x_1 + d_2 \nu_2$.

Step 4. Eventually, they can output signature $\sigma_i = (R_i, \mathscr{V}_k)$ and transmits $(\mathrm{PID}_{y,i}, m_i, \mathrm{PK}_i, \omega_i, \sigma_i, T_i)$.

$$\begin{aligned}
\mathscr{V} \cdot P &= (\mathscr{V}_1 + \mathscr{V}_2) \cdot P = (\delta_1 \mathrm{SK}_1 + h_2 x_2 + d_1 \nu_1 \\
&\quad + \delta_2 \mathrm{SK}_2 + h_1 x_1 + d_2 \nu_2) \\
&\quad \cdot P = \left(\sum_{k=1}^{2} [\delta_k \mathrm{SK}_k + h_k (t\beta_k + \xi_k \alpha_k s) + d_k \nu_k] \right) \\
&\quad \cdot P = \left(\sum_{k=1}^{2} \delta_k \cdot \mathrm{PK}_k + h_k \cdot A_k + \omega_k \right) \\
&\quad + \left(\sum_{k=1}^{2} h_k \omega_k \cdot \mathrm{PK}_{\mathrm{RTMA}} \right) = R + \left(\sum_{k=1}^{2} h_k \omega_k \cdot \mathrm{PK}_{\mathrm{RTMA}} \right)
\end{aligned} \tag{5}$$

Obviously, the signature $\sigma_i = (R_i, \mathscr{V}_i)$ is not a valid signature. However, when the RSU or AS aggregates the signature as $\sigma = (R = R_1 + R_2, \mathscr{V} = \mathscr{V}_1 + \mathscr{V}_2)$, it will be a valid signature which satisfies the following equation.

Therefore, the above analysis shows that two malicious users can collude with each other to forge an aggregate signature. Actually, the coalition attack is originally caused by commutative law of addition. Similarly, n users can also forge an aggregate signature with the same algorithms. Hence, Kamil et al.'s CL-AS scheme cannot resist coalition attacks.

5. Our Proposed CLS and CL-AS Schemes

5.1. System Model. In this section, we will try to describe our system model in detail including specific explanations. In order to be more specific, the system model is shown in Figure 2. There are four participants in total: trusted authority (TA), key generation center (KGC), road-side unit (RSU), and vehicle, which can be divided into two layers: the upper layer includes TA and KGC, and the lower layer consists of RSUs and vehicles. The demonstration of each participant is as follows:

(1) TA: it is a fully trusted third party that is responsible for system initialization, user registration, system parameter generation, and system security implementation. If necessary, it can track malicious behavior and catch malicious nodes. In addition, it also has enough computing power and storage capacity.

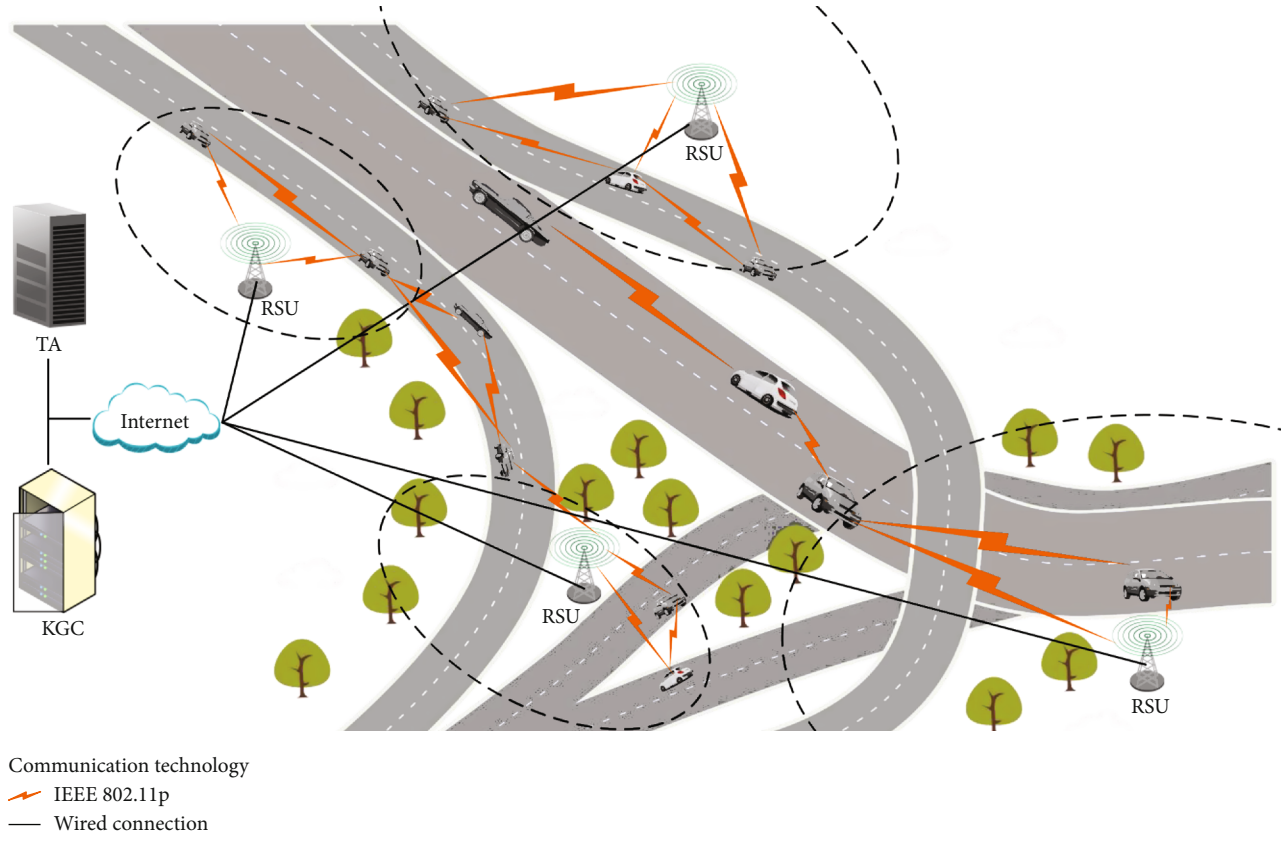


Figure 1: Coalition attack diagram.

(2) KGC: it is a partially trusted party used for generating partial private key. It can help a vehicle generate partial secret key which contribute to its privacy security. Like the TA, it also has sufficient memory, processing, and computing capabilities.

(3) RSU: it is a smart application device installed in the roadside, which is able to transmit and submit information to TA, KGC, vehicles, or other RSUs in a secure wired connection. In addition, RSU commonly has limited computing power and storage capacity.

(4) Vehicle: it is the major and basic member in VANETs, which is generally equipped with a smart device which can perform the basic function such as transmitting the vehicle's message and performing simple calculation. In addition, vehicle commonly has limited computing power and storage capacity.

Note that TA and KGC are functionally two completely different entities that can be deployed on a single server during deployment.

5.2. Design Requirements. For the safety of communication in VANETs, security and privacy are crucial. According to the latest research in this field, the proposed scheme for VANETs must satisfy the following security requirements:

(1) Message Integrity and Authentication: an eligible vehicle should be able to check that whether a message is sent and signed by a legitimate vehicle and is not forged or modified by the malicious entity.

(2) Identity Privacy Preservation: a vehicle should remain anonymous in all circumstances, which means that other malicious entities cannot infer its identity by taking and analyzing multiple pieces of messages about it.

(3) Traceability: the TA must have the ability to trace and obtain the vehicle's real identity, even if the vehicle's identity is anonymous.

(4) Unlinkability: a potentially malicious vehicle must not cross-link two messages sent by the same vehicle to prevent them from extrapolating the route of the vehicle from the information.

(5) Resistance to Attacks: a reasonable scheme should have the ability to withstand various general attacks such as the coalition attack, the impersonation attack, the modification attack, and the replay attack.

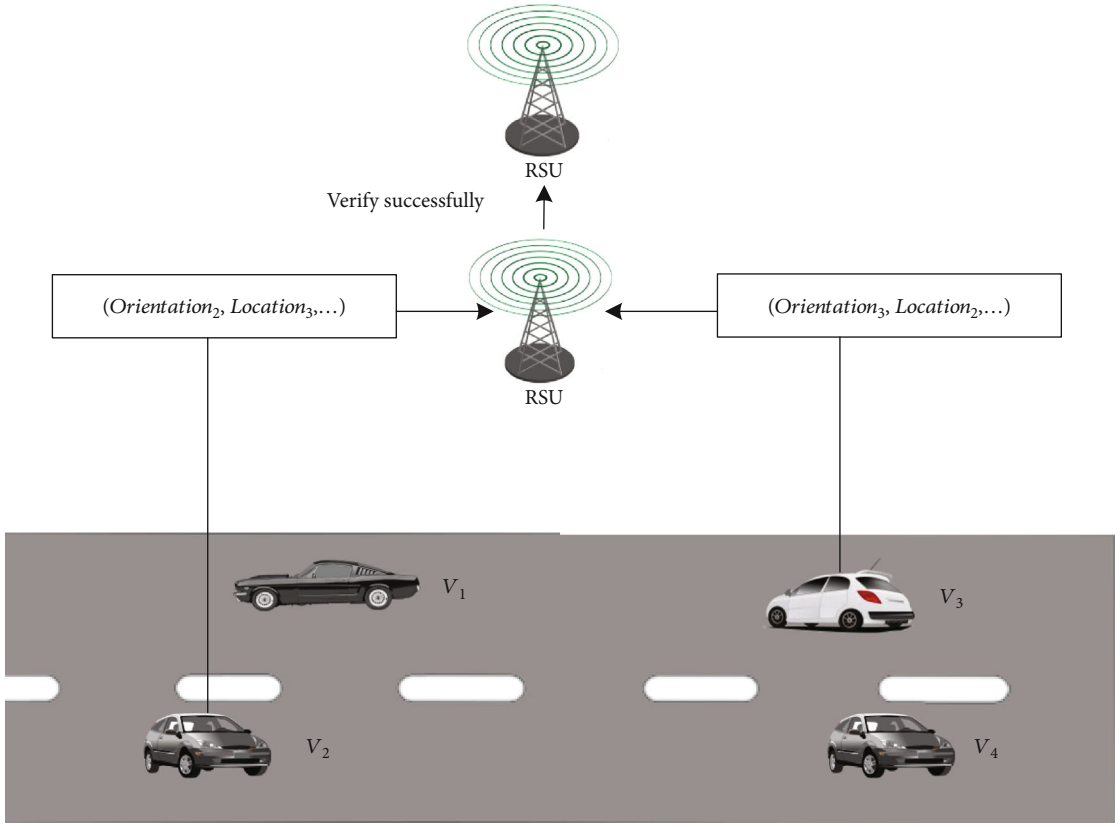


Figure 2: Certificateless aggregate signature system model.

Table 1: List of notations.

Notation	Description
TA	Trusted authority
KGC	Key generation center
RSU	Road-side unit
$h_i(.), i = 1, 2, 3$	One-way collision-resistant hash function
p, q	Two secure prime numbers
E	An elliptic curve:$y^2 = x^3 + ax + b \bmod p$
$\mathbb{G}$	An additive group, the order of which is q
P	A generator of the group $\mathbb{G}$
(PK_{TA}, α)	The public key and private key of the TA
(PK_{KGC}, β)	The public key and private key of the KGC
(PK_{V_i}, ρ_i)	The public key and private key of the vehicle
PPK_i	Partial private key of the vehicle V_i
PID_i	Pseudo identity of the vehicle V_i
TS	The latest timestamp
∇	State information

5.3. Our Proposed CLS Scheme. Our proposed CLS scheme includes five algorithms: Setup, PartialPrivateKeyGeneration, VehicleKeyGeneration, Sign, and Verify. The Notation to be used is listed in Table 1, and descriptions for algorithms are vividly shown in Figure 3 and described as follows:

(1) Setup: when given an appropriate security parameter λ, TA will use the λ to generate and output the param by executing the following algorithms:

(2) PartialPrivateKeyGeneration: the algorithm will eventually generate the vehicle's partial private key through the algorithms as follows:

(3) VehicleKeyGeneration: after receiving the partial private key PPK_i, the vehicle V_i check if the equation $PPK_i \cdot P = Q_i + n_i \cdot PK_{KGC}$ holds. If it holds, the partial private key PPK_i is valid. The vehicle randomly selects its private key $\rho_i \in \mathbb{Z}_q^*$, then calculates its public key $PK_{V_i} = \rho_i \cdot P$.

(4) Sign: in order to achieve authentication and message integrity, when the message is received by any entity, it has to be signed and verified. A vehicle V_i uses its pseudo identity PID_i and picks the latest timestamp TS. The updated timestamp TS protects a signed message against replay attacks. Given the signing key (PPK_i, PK_{V_i}) and a traffic related message m_i, the vehicle V_i performs the following steps, which are repeated every 100 – 300 ms in accordance with DSRC protocol [20]:

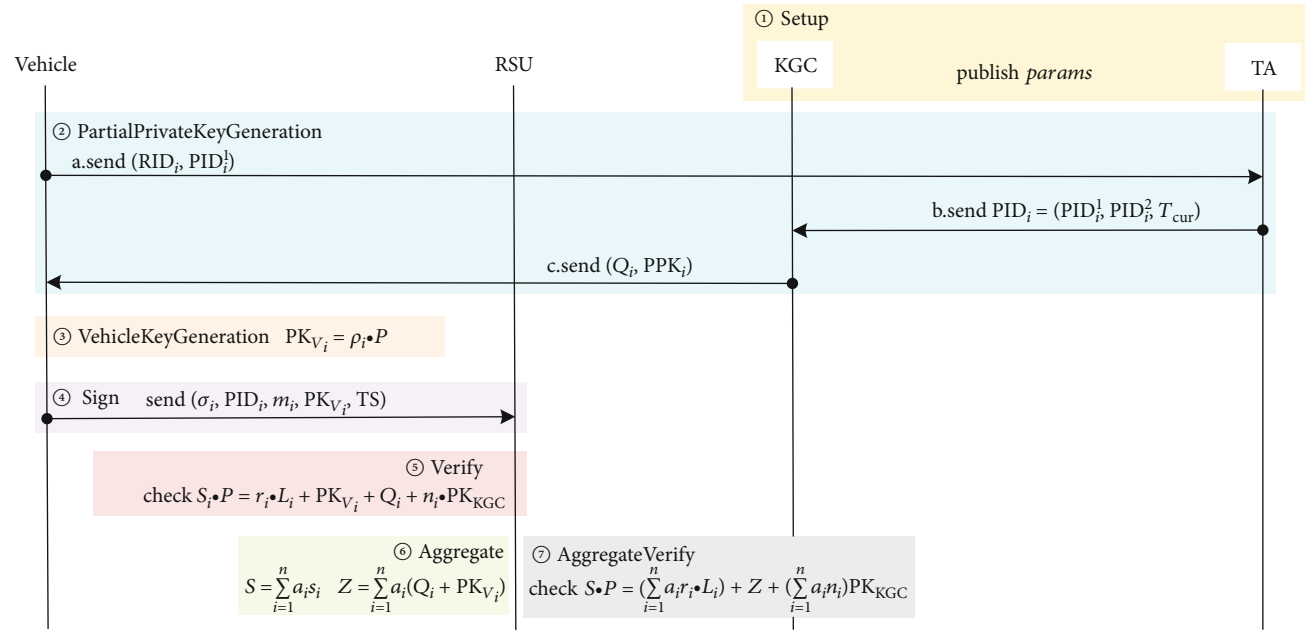


Figure 3: The algorithm procedure.

(5) Verify: when an RSU or other entity receives the signature σ and the tuple $(Q_i, \mathrm{PID}_i, m_i, \mathrm{PK}_{V_i}, \mathrm{TS})$ from the vehicle V_i, it can execute the algorithms to verify the message as follows:

Step 1. Firstly, select two secure prime numbers p and q, then choose $a,b \in F_p$, which generate an ellipic curve E defined by the equation $y^2 = x^3 + ax + b \bmod p$, where $\Delta = 4a^3 + 27b^2 \neq 0 \pmod p$ and generator P of the additive group $\mathbb{G}$ consisting of all the points on E.

Step 2. Choose $\alpha \in \mathbb{Z}_q^*$ in random, which serves as the master secret key and computes master public key $\mathrm{PK}_{\mathrm{TA}} = \alpha \cdot P$. KGC selects $\beta \in \mathbb{Z}_q^*$ in random, then calculates $\mathrm{PK}_{\mathrm{KGC}} = \beta \cdot P$ which is the public key of KGC.

Step 3. Select three secure hash functions in random: $h_1 : \mathbb{G} \times \{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_q^*$, $h_2 : \{0,1\}^* \times \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_q^*$, $h_3 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_q^*$.

Step 4. Store its master secret key α in its repository and keep it safe. Then publish all the system parameter:

$$\mathrm{param} = \{p, q, a, b, P, PK_{\mathrm{TA}}, PK_{\mathrm{KGC}}, h_1, h_2, h_3\}. \quad (6)$$

Step 1. The vehicle V_i with its real identity RID_i randomly selects $x_i \in \mathbb{Z}_q^*$ as its private key and calculates its partial pseudo identity $\mathrm{PID}_i^1 = x_i \cdot P$. Then vehicle V_i transmits $(\mathrm{RID}_i, \mathrm{PID}_i^1)$ to TA.

Step 2. After receiving the tuple, TA calculates another pseudo identity $\mathrm{PID}_i^2 = \mathrm{RID}_i \oplus h_1(\alpha \mathrm{PID}_i^1 \| T_{\mathrm{cur}} \| \mathrm{PK}_{\mathrm{TA}} \| \nabla)$, where ∇ is the system state information [19]; then TA sends the vehicle's pseudo identity $\mathrm{PID}_i = (\mathrm{PID}_i^1, \mathrm{PID}_i^2, T_{\mathrm{cur}})$ to KGC in a secure way.

Step 3. KGC calculates $Q_i = y_i \cdot P$, $n_i = h_2(\mathrm{PID}_i \| Q_i \| \nabla)$ and the vehicle's partial private key $\mathrm{PPK}_i = y_i + h_2(\mathrm{PID}_i \| Q_i \| \nabla) \times \beta \pmod p$. At last, KGC transmits the tuple (Q_i, PPK_i) to vehicle V_i.

Step 1. Choose a random number $l_i \in \mathbb{Z}_q^*$ and calculate $L_i = l_i \cdot P$.

Step 2. Calculate $r_i = h_3(m_i \| \mathrm{PID}_i \| \nabla \| \mathrm{PK}_{V_i} \| L_i \| \mathrm{TS})$, where timestamp TS is used to confirm time, and $S_i = r_i l_i + \rho_i + \mathrm{PP}\,\mathrm{K}_i \bmod p$.

Step 3. The signature on message m_i is $\sigma = (L_i, S_i)$; then the vehicle transmits the signature σ and $(Q_i, \mathrm{PID}_i, m_i, \mathrm{PK}_{V_i}, \mathrm{TS})$ to the verifier.

Step 1. Check whether the TS is valid, if not, the algorithm aborts; otherwise, execute the next step.

Step 2. Calculate $r_i = h_3(m_i \| \mathrm{PID}_i \| \nabla \| \mathrm{PK}_{V_i} \| L_i \| \mathrm{TS})$ and $n_i = h_2(\mathrm{PID}_i \| Q_i \| \nabla)$

Step 3. Check whether the following equation

$$S_i \cdot P = r_i \cdot L_i + \mathrm{PK}_{V_i} + Q_i + n_i \cdot \mathrm{PK}_{\mathrm{KGC}} \quad (7)$$

holds or not; if it holds, then the RSU or other entity will

accept the signature and the massage; otherwise, it will reject the message.

5.4. Our Proposed CL-AS Scheme. The Setup, PartialPrivateKeyGeneration, VehicleKeyGeneration, Sign, and Verify algorithms of CL-AS are similar to the proposed CLS scheme. In addition, the Aggregate and AggregateVerify are described as follows. Note that the Aggregate and AggregateVerify algorithms are usually executed by the same RSU to transmit less data in the communication process.

(1) Aggregate: when an aggregator such as a RSU receives n vehicles' messages $M = \{m_1, m_2, \cdots, m_n\}$, signatures $\sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_n\}$, timestamps $\mathrm{TS} = \{\mathrm{TS}_1, \mathrm{TS}_2, \cdots, \mathrm{TS}_n\}$, $Q = \{Q_1, Q_2, \cdots, Q_n\}$, public key of each vehicle $\mathrm{PK}_V = \{\mathrm{PK}_{V_1}, \mathrm{PK}_{V_2}, \cdots, \mathrm{PK}_{V_n}\}$, $L = \{L_1, L_2, \cdots, L_n\}$, and pseudo identities $\mathrm{PID} = \{\mathrm{PID}_1, \mathrm{PID}_2, \cdots, \mathrm{PID}_n\}$. It can execute the following algorithms to aggregate the signature:

(2) AggregateVerify: after aggregating n vehicles' messages, the same RSU will execute the following algorithms to verify the aggregate signature as follows:

Step 1. Randomly choose a random list $\mathrm{RL} = \{a_1, a_2, \cdots, a_n\}$, where $a_i \in \mathbb{Z}_q^*$, $1 \le i \le n$. Note that the random list RL is firstly introduced in [21, 22] and used for resisting coalition attacks here.

Step 2. Calculate $S = \sum_{i=1}^{n} a_i S_i$ and $Z = \sum_{i=1}^{n} a_i (Q_i + \mathrm{PK}_{V_i})$.

Step 3. Outputs the signature $\sigma = (L, S)$ and transmits $(M, \sigma, Z, \mathrm{RL}, \mathrm{PID}, Q, \mathrm{TS})$ to the verifier.

Step 1. Check whether the timestamp list TS is valid, if not, the algorithm aborts; otherwise, it executes next step.

Step 2. For every vehicle, calculate $r_i = h_3(m_i \| \mathrm{PID}_i \| \nabla \| \mathrm{PK}_{V_i} \| L_i \| \mathrm{TS})$ and $n_i = h_2(\mathrm{PID}_i \| Q_i \| \nabla)$.

Step 3. Check whether the following equation

$$S \cdot P = \left(\sum_{i=1}^{n} a_i r_i \cdot L_i \right) + Z + \left(\sum_{i=1}^{n} a_i n_i \right) \cdot \mathrm{PK}_{\mathrm{KGC}} \tag{8}$$

holds or not, if it holds, then the RSU or other entity will accept the signature and the message, then the RSU can transmit them to other entities; otherwise, it will reject the message.

5.5. Correctness of Individual Message Verification. The individual verification in the proposed scheme is correct. The correctness proof is as follows:

$$\begin{aligned} S_i \cdot P &= (r_i l_i + \rho_i + \mathrm{PPK}_i) \cdot P = (r_i l_i + \rho_i + y_i + n_i \beta) \\ &\cdot P = r_i l_i \cdot P + \rho_i \cdot P + y_i \cdot P + n_i \beta \\ &\cdot P = r_i \cdot L_i + \mathrm{PK}_{V_i} + Q_i + n_i \mathrm{PK}_{\mathrm{KGC}}. \end{aligned} \tag{9}$$

5.6. Correctness of Aggregate Message Verification. The aggregate verification in the proposed scheme is correct. The correctness proof is as follows:

$$\begin{aligned} S \cdot P &= \sum_{i=1}^{n} a_i S_i \cdot P = \left(\sum_{i=1}^{n} a_i r_i l_i + a_i \rho_i + a_i \mathrm{PPK}_i \right) \\ &\cdot P = \sum_{i=1}^{n} a_i (r_i l_i \cdot P + \rho_i \cdot P + y_i \cdot P + n_i \beta \cdot P) \\ &= \left(\sum_{i=1}^{n} a_i r_i \cdot L_i \right) + Z + \left(\sum_{i=1}^{n} a_i n_i \right) \cdot \mathrm{PK}_{\mathrm{KGC}}. \end{aligned} \tag{10}$$

5.7. Security Proof of the Proposed CLS Scheme. According to Definition 3, it is extremely hard to solve ECDLP. Therefore, we can prove that our CLS scheme is able to enforce nonforgery.

On the basis of Definition 4, assume that a probabilistic polynomial-time forger A_1 can forge a signature with an advantage ε. In addition, q_{h_i} denotes random oracles h_i for $i = 2, 3$, q_{GU} denotes the Generate-User oracle, q_{PPK} denotes Partial-Private-Key oracle, and q_{SK} denotes the Secret-Key oracle. Then, we can know that a challenger C_1 can solve ECDLP during a time scope T, where $T \le 120686QT/\varepsilon$, if $\varepsilon \ge 10(q_S + 1)(q_{h_2} + q_{h_3} + q_{\mathrm{PPK}} + q_{\mathrm{GU}} + q_{\mathrm{SK}} + q_S)/q$.

(1) Setup: C_1 chooses α and calculates $\mathrm{PK}_{\mathrm{TA}} = \alpha \cdot P$ which serves as its private key and master public key. Then, C_1 will generate the system parameters $\mathrm{param} = (P, p, q, E, G, h_1, h_2, h_3, \mathrm{PK}_{\mathrm{TA}}, \mathrm{PK}_{\mathrm{KGC}})$, and transmit it to A_1.

(i) $\boldsymbol{h_2}$ Hash Query: C_1 will examine whether the hash list L_{h_2} has the corresponding tuple if it receives the query with parameter (PID_i, Q_i) from A_1. If not, C_1 will select a random number $\tau_{h_2} \in \mathbb{Z}_q^*$ and put it in the list L_{h_2}. If so, it needs to transmit $\tau_{h_2} = h_2(\mathrm{PID}_i \| Q_i \| \nabla)$ to A_1.

(ii) $\boldsymbol{h_3}$ Hash Query: C_1 will examine whether the hash list L_{h_3} has the corresponding tuple $(m_i, \mathrm{PID}_i, \mathrm{PK}_{V_i}, Z_i, \mathrm{TS}, \tau_{h_3})$ if it receives the query with parameter $\mathrm{param} = (m_i, \mathrm{PID}_i, \mathrm{PK}_{V_i}, L_i, \mathrm{TS})$ from A_1. If not, C_1 will choose a random number $\tau_{h_3} \in \mathbb{Z}_q^*$ and put the tuple $(m_i, \mathrm{PID}_i, \mathrm{PK}_{V_i}, L_i, \mathrm{TS}, \tau_{h_3})$ in the list $L_h 3$. If so, it will transmit $\tau_{h_3} = h_3(m_i \| \mathrm{PID}_i \| \nabla \| \mathrm{PK}_{V_i} \| L_i \| \mathrm{TS})$ to A_1. Eventually, C_1 will transmit $\tau_{h_3} = h_3(m_i \| \mathrm{PID}_i \| \nabla \| \mathrm{PK}_{V_i} \| L_i \| \mathrm{TS})$ to A_1.

(2) Partial-Private-Key Query: after receiving a query about the identity PID_i from A_1, C_1 will calculate $Q_i = y_i \cdot P$, where y_i is randomly selected, and check whether the hash list L_{h_2} has the corresponding tuple (PID_i, Q_i, τ_{h_2}). If so, C_1 will calculate $\mathrm{PPK}_i = y_i + h_2(\mathrm{PID}_i \| Q_i \| \nabla) \times \alpha \bmod p$ and transmit the pairial private key of vehicle $V_i \mathrm{PPK}_i$ to A_1. If not, it will halt.

(3) User-Generation Query: suppose that the query is on the basis of the pseudo identity PID_i

(i) C_1 will check whether PK_{V_i} exists in the list L, if the list L includes $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$. If not, a random number $\rho_i \in \mathbb{Z}_q^*$ will be selected and C_1 will calculate $\mathrm{PK}_{V_i} = \rho_i \cdot P$. If so, it will transmit PK_{V_i} to A_1. Eventually, the chanllenger C_1 will transmit PK_{V_i} to A_1 and update the list

(ii) C_1 will set $\mathrm{PK}_{V_i} = \perp$ if the tuple $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$ does not exist in the list L. Then, a random number $\rho_i \in \mathbb{Z}_q^*$ will be chosen and $\mathrm{PK}_{V_i} = \rho_i \cdot P$ will be calculated and ρ_i will be regarded as a private key. Eventually, C_1 will transmit PK_{V_i} to A_1 and put the tuple $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$ to the list L

(4) Private-Key Query:

(i) C_1 will check whether ρ_i exists in the list L, if the list L includes $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$. If not, it will access a User-Generation query to output the public key $\mathrm{PK}_{V_i} = \rho_i \cdot P$. Eventually, the chanllenger C_1 will transmit ρ_i to A_1 and update the list

(ii) C_1 will access a User-Generation query if the tuple $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$ does not exist int he list L. Eventually, C_1 will transmit ρ_i to A_1 and put the tuple $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, \rho_i)$ to the list L

(5) Sign Query: after receiving a legitimate query about the message m_i of pseudo identity PID_i, C_1 will check the tuple $(\mathrm{PID}_i, Q_i, \tau_{h_2})$ in the hash list L_{h2}. Hence, it can easily get the value τ_{h_2} from the tuple and select two random numbers l_i and r_i. Then, C_1 will choose another two random numbers s_i and n_i. Furthermore, C_1 will calculate $Z_i = s_i \cdot P - n_i \cdot \mathrm{PK}_{\mathrm{KGC}}$ and $S_i = s_i$. Eventually, it will transmit (L_i, S_i) to A_1 and put the tuple $(m_i, \mathrm{PID}_i, \mathrm{PK}_{V_i}, L_i, \mathrm{TS}, \tau_{h_3})$ in the list L_{h3}.

Theorem 7. *According to the random oracle, when faced with an adaptive chosen message attack, our proposed scheme has the capacity of unforgeability.*

Proof. Assume that an ECDLP sample $(P, \mathrm{Q} = x \cdot P)$ is given, the elliptic curve E holds two points P and Q, and an adversary A_1 is able to forge message $(\mathrm{PID}_i, \mathrm{PK}_{V_i}, m_i, \mathrm{TS}, \sigma_i)$. Hence, we start a game between a challenger C_1 and the adversary A_1, which can execute and manipulate A_1 to solve ECDLP with a nonnegligible probability.

We know the forking lemma in Definition 4 and apply it to our proposed scheme. After using the same random elements to replay A_1, C_1 succeeds in getting two legitimate signatures $\sigma_i = (Z_i, S_i)$ and $\sigma'_i = (L'_i, S'_i)$ during a polynomial time period, where $S_i = r_i \cdot L_i + Q_i + \mathrm{PK}_{V_i} \pmod p$ and $S'_i = t'_i \cdot L_i + Q_i + \mathrm{PK}_{V_i} \pmod p$ by computing.

$$\frac{t'_i S_i - r_i S'_i}{t'_i - r_i} \bmod p = \frac{t'_i r_i l_i + t'_i Q_i + t'_i PK_{V_i} - r_i t'_i l_i - r_i PPK_i - r_i Q_i + r_i PK_{V_i}}{t'_i - r_i} \bmod p = Q_i + PK_{V_i}. \tag{11}$$

In conclusion, if $\varepsilon \geq 10(q_S + 1)(q_{h2} + q_{h3} + q_{\mathrm{PPK}} + q_{\mathrm{UG}} + q_{\mathrm{SK}} + q_S)/q$, then C_1 is able to break the ECDLP during a time period which is less than $120686QT/\varepsilon$. However, this conclusion is inconsistent with the difficulty of solving the ECDLP. Therefore, we can define that our proposed CLS scheme can resist a forgery attack.

5.8. Security Proof of the Proposed CL-AS Scheme. According to Definition 3, it is extremely hard to solve ECDLP. Therefore, we can prove that our scheme is able to enforce nonforgery. Furthermore, we will prove that our CL-AS scheme can also resist coalition attack.

(1) Setup: a random number α is selected as the master secret key, and the public key can also be calculated as $\mathrm{PK}_{\mathrm{TA}} = \alpha \cdot P$. Then, the oracle simulation is ready to run. In this whole game, C_2 maintains a list $L = \{\mathrm{PID}_i, \mathrm{PPK}_i, \mathrm{PK}_{V_i}, \rho_i\}$ and responds to A_2's oracle as follows.

(i) $\boldsymbol{h_2}$ Query: after receiving a pseudo identity PID_i, C_2 will throw a coin $c_i \in \{0, 1\}$, where 0 holds a probability ε, and 1 holds a probability $1 - \varepsilon$, then C_2 will select

$w_i^1 \in \mathbb{Z}_q^*$. If $c_i = 1$, C_2 will output $Q_i = w_i^1 \cdot P$. Otherwise, it will define $Q_i = \omega_i^1 \cdot P$. C_2 will put the tuple $(\mathrm{PID}_i, w_i^1, c_i, Q_i)$ in a list $L_{h2} = (\mathrm{PID}_i, w_i^1, c_i, Q_i)$ to trace what the queries respond no mather what the value c_i is.

Theorem 8. *According to the random oracle, when faced with an adaptive chosen message attack, our proposed CL-AS scheme has the capacity of unforgeability.*

Proof. Suppose that our CL-AS scheme can be broken by forger A_2. We can construct a challenger C_2 using forgery algorithm A_2. Challenger C_2 is able to execute the following steps by interacting with A_2.

Then, A_2 will transmit n vehicles with identities from the list $L_{\mathrm{PID}}^* = \{\mathrm{PID}_1^*, \mathrm{PID}_2^*, \cdots, \mathrm{PID}_n^*\}$, public keys from the list $L_{\mathrm{PK}_V}^* = \{\mathrm{PK}_{V_1}^*, \mathrm{PK}_{V_2}^*, \cdots, \mathrm{PK}_{V_n}^*\}$, n messages $L_M^* = \{m_1^*, m_2^*, \cdots, m_n^*\}$, a random list $RL^* = \{a_1^*, a_2^*, \cdots, a_n^*\}$, and a certificateless aggregate signature $\sigma^* = \{L^*, S^*\}$. At the beginning, C_2 will select the n tuples $(\mathrm{PID}_i^*, w_i^{1*}, c_i^*, Q_i^*)$ for $i = 1, 2, \cdots, n$ in the list L_{h2} and precede only $c_k = 1$ and $c_j = 0$ for $j = 1, 2, \cdots, n, j \neq k$. Note that the Sign oracle has not received the tuple $(\mathrm{PID}_k^*, \mathrm{PK}_{V_k}^*, m_k^*)$. Otherwise, C_2 will halt and fail. This success case signifies that $Q_k = w_k^1 \cdot \mathrm{PK}_{\mathrm{TA}}$ and $Q_j = w_j^1 \cdot P$ for $j = 1, 2, \cdots, n, j \neq k$. In addition, the aggregate signature $\sigma^* = (L^*, S^*)$ is supposed to satisfy the aggregate verification equation $S \cdot P = \sum_{i=1}^n (a_i r_i \cdot L_i) + Z + (\sum_{i=1}^n a_i n_i) \cdot \mathrm{PK}_{\mathrm{KGC}}$.

Accordingly, C_2 checks the tuples $(m_i^*, \mathrm{PID}_i^*, \mathrm{PK}_{V_i}^*, Z_i^*, w_i^{2*})$ in the list L_{h3} and the tuple $(\mathrm{PID}_i^*, \mathrm{PPK}_i^*, \mathrm{PK}_{V_i}^*, \rho_i)$ from L. Later, it calculates $S_i^* = w_i^{1*} \cdot \alpha \bmod p$, which will satisfy $S_i^* \cdot P = w_i^{1*} \cdot \mathrm{PK}_{\mathrm{TA}} = Q_i^*$ for $i = 1, 2, \cdots, n, i \neq k$. Eventually, C_2 constructs S'^* as $S'^* = S^* - \sum_{i=1, i\neq k}^n a_i S_i^*$, $S'^* = \mathrm{PPK}_i^* + \sum_{i=1}^n w_i^{2*} r_i^* \pmod p$ for $L^* = l_i^*$ and $l_i^* \in \mathbb{Z}_q^*$ for $1 \leq i \leq n$. C_2 will select a random number $h_k^* \in \mathbb{Z}_q^*$ and calculate $Z^* = (h_k^*)^{-1} \sum_{i=1}^n w_i^{2*} a_i (r_i \cdot L_i + Q_i + \mathrm{PK}_{V_i})$.

Hence, the hash value $h_3(m_k^*, \mathrm{PID}_k^*, \mathrm{PK}_{V_i}^*, Z_k^*)$ is defined as h_k^*. It will use h_k^* until it does not repeat if the list L_{h3} holds the tuple $h_3(\mathrm{m}_k^*, \mathrm{PID}_k^*, \mathrm{PK}_{V_k}^*, Z_k^*)$. Consequently, the signature (L'^*, S'^*) is a legitimate certificateless signature on message m_k^* for the reason that the equation below:

$$\begin{aligned} Q_{v_k^*}^* + h_{k,0}^* + h_k^* Z'^* &= Q_{v_k^*}^* \\ &+ h_{k,0}^* + h_k^* (h_k^*)^{-1} \sum_{i=1}^n w_i^{2*} a_i \left(r_i \cdot L_i + Q_i + \mathrm{PK}_{V_i}\right) \\ &= Q_{v_k^*}^* + h_{k,0}^* + \sum_{i=1}^n w_i^{2*} a_i \left(r_i \cdot L_i + Q_i + \mathrm{PK}_{V_i}\right) \\ &= \mathrm{PPK}_k^* \cdot P + \sum_{i=1}^n w_i^{2*} a_i \left(r_i \cdot L_i + Q_i + \mathrm{PK}_{V_i}\right) = S'^* \cdot P. \end{aligned} \tag{12}$$

Eventually, S can get the signature (L'^*, S'^*) as a forgery of the certificateless signature scheme. However, this conclusion is inconsistent with the difficulty of solving the ECDLP. Therefore, we can define that our proposed CLS scheme can resist a forgery attack.

Theorem 9. *The proposed certificateless aggregate signature (CL-AS) scheme can resist coalition attacks.*

Proof. Assume that there are two malicious vehicles V_1 and V_2 with pseudonyms PID_1 and PID_2 and messages m_1 and m_2, respectively, and that all other system params are published by TA and KGC. According to the description in Subsection 4.3, two vehicles V_1 and V_2 would like to execute similar algorithms to forge valid signatures. However, our proposed scheme can perfectly resist the coalition attacks; the detailed descriptions are as follows:

To begin with, two vehicles V_i $(i \in \{1, 2\})$ pick their own private key ρ_i and calculate their corresponding public key $PK_{V_i} = \rho_i \cdot P$.

According to the aforementioned algorithms in Subsection 5.7, two malicious vehicles execute the algorithms in order but secretly exchange their $r_i L_i$, which is a part of the signature. Eventually, two vehicles transmit their messages m_i, signatures σ_i, timestamp TS_i, and pseudo identity PID_i to the aggregator.

When the aggregator receives the above information, it will aggregate the signature as follows: firstly choose a random list $\mathrm{RL} = \{a_1, a_2\}$, where $a_i \in \mathbb{Z}_q^*$, $1 \leq i \leq n$, then calculate $S = \sum_{i=1}^2 a_i S_i$ and $Z = \sum_{i=1}^2 a_i (Q_i + \mathrm{PK}_{V_i})$. Finally, the aggregator will output the signature $\sigma = (L, S)$ and transmits $(M, \sigma, Z, \mathrm{RL}, \mathrm{PID}, Q, T)$ to the verifier.

In the last step, the verifier will check the equation $S \cdot P = \sum_{i=1}^2 (a_i r_i \cdot L_i) + Z + (\sum_{i=1}^n a_i n_i) \cdot \mathrm{PK}_{\mathrm{KGC}}$ holds or not. Unfortunately, the equation is impossible as follows:

$$\begin{aligned} S \cdot P &= (a_1 \mathrm{r}_2 l_2 + a_1 \rho_i + a_1 \mathrm{PPK}_1 + a_2 t_1 l_1 + a_2 \rho_i + a_2 \mathrm{PPK}_2) \\ &\cdot P = a_1 t_2 \cdot L_2 + a_1 \cdot \mathrm{PK}_{V_1} + a_1 \cdot Q_1 + a_1 n_1 \cdot \mathrm{PK}_{\mathrm{KGC}} + a_2 t_1 \\ &\cdot L_1 + a_2 \cdot \mathrm{PK}_{V_2} + a_2 \cdot Q_2 + a_2 n_2 \cdot \mathrm{PK}_{\mathrm{KGC}} = a_1 t_2 \cdot L_2 + a_2 \\ &\cdot t_1 L_1 + Z + \sum_{n=1}^2 (a_i n_i) \cdot \mathrm{PK}_{\mathrm{KGC}} \neq \sum_{i=1}^2 (a_i r_i \cdot L_i) + Z \\ &+ \sum_{n=1}^2 (a_i n_i) \cdot \mathrm{PK}_{\mathrm{KGC}}. \end{aligned} \tag{13}$$

One can find that the random list plays an important role in resisting the coalition attacks. And the 2-vehicle situation can also be developed to n vehicles simply with a fully the same method and algorithm, which can prove that our proposed certificateless aggregate signature (CL-AS) scheme can resist coalition attacks.

Table 2: Cryptographic operation notations and executing time.

Operation	Description	Time (ms)
T_{bp}	Bilinear pairing operation	4.1603
$T_{\text{bp}-m}$	Scalar multiplication in bilinear pairing operation	1.6722
$T_{\text{bp}-a}$	Addition in bilinear pairing operation	0.0069
T_{e-m}	Scalar multiplication in ECC operation	1.1280
T_{e-a}	Addition in ECC operation	0.0339
T_h	One-way hash function	0.0360

6. Performance and Security Analysis

6.1. Security Analysis

(1) Traceability: in the proposed scheme, only TA has the real identity of the certain vehicle. After submitting the pseudo identity $\text{PID}_i = (\text{PID}_i^1, \text{PID}_i^2, T_{\text{cur}})$, TA can easily trace back to the vehicle's real identity RID_i in accordance with the equation $\text{PID}_i^2 = \text{RID}_i \oplus h_1(\alpha \text{PID}_i^1 \| T_{\text{cur}} \| \text{PK}_{\text{TA}} \| \nabla)$. Therefore, according to the RID list, TA can trace back to the certain vehicle V_i, even revoke it. $(\text{RID}_i, \text{PID}_i^1)$

(2) Message integrity and authentication: according to Definition 3, the ECDLP problem is hard, so that no polynomial adversary can forge a valid message. Therefore, the verifier can verify the validity and integrity of the message $(Q_i, \text{PID}_i, \text{PK}_{V_i}, m_i, \text{TS}, \sigma_i)$ by verifying whether the equation $S_i \cdot P = r_i L_i + \text{P}\text{K}_{V_i} + Q_i + n_i \cdot \text{PK}_{\text{KGC}}$ holds or not. Therefore, our proposed scheme for VANETs provides message authentication and integrity.

(3) Resistance to replay attacks: the proposed scheme can resist the replay attack for the reason that the tuple $(Q_i, \text{PID}_i, \text{PK}_{V_i}, m_i, \text{TS}, \sigma_i)$ includes the timestamp TS. RSU and other vehicles will check the validity of the signature, so they are able to detect the replay of the message. Hence, our proposed scheme for VANETs can resist replay attacks.

(4) Resistance to coalition attacks: our proposed scheme can resist the coalition attacks, because we improve the signature generation process. To be specific, we choose a random list to change the ratio in the equation $S = \sum_{i=1}^{n} a_i S_i$. Therefore, our scheme uses this method to resist the coalition attacks.

(5) Resistance to stealing of the check table: in the proposed scheme, TA, KGC, vehicles, and RSUs do not require a check list. Therefore, an attacker cannot complete an attack by stealing any checklist. Hence, the proposed scheme can resist the attack of the checklist.

6.2. Performance Analysis. In this section, we will discuss the performance of the proposed scheme and related schemes and make a comparison in detail. We adopt the method of computation evaluation where the bilinear pairing on the security level of 80 bits is created as follows: $\bar{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_1$ is an additive group generated by a point $\bar{P}$ with order $\bar{q}$ on a super singular elliptic curve $\bar{E} : y^2 = x^3 + x \bmod \bar{p}$ with embedding degree 2, $\bar{p}$ is a 512-bit prime number, $\bar{q}$ is a 160-bit prime number [25]. The ECC on the security of 80 bits is constructed as follows: $\mathbb{G}$ is an additive group with order q that is generated on a nonsingular elliptic curve $E : y^2 = x^3 + ax + b \bmod p$, where p, q are 160-bit primes and $a, b \in \mathbb{Z}_q^*$. The experiment is conducted using the well-known python cryptographic library PyCryptodome on a desktop running Intel I5-9400 @ 2.90 GHz processor, with 16 GB memory running Windows 10 operating system. The notations of the cryptographic operations used in this paper and their running times are given in Table 2. Table 3 shows the summary of the computation costs in terms of signing a message, verifying a single message, and verifying n messages.

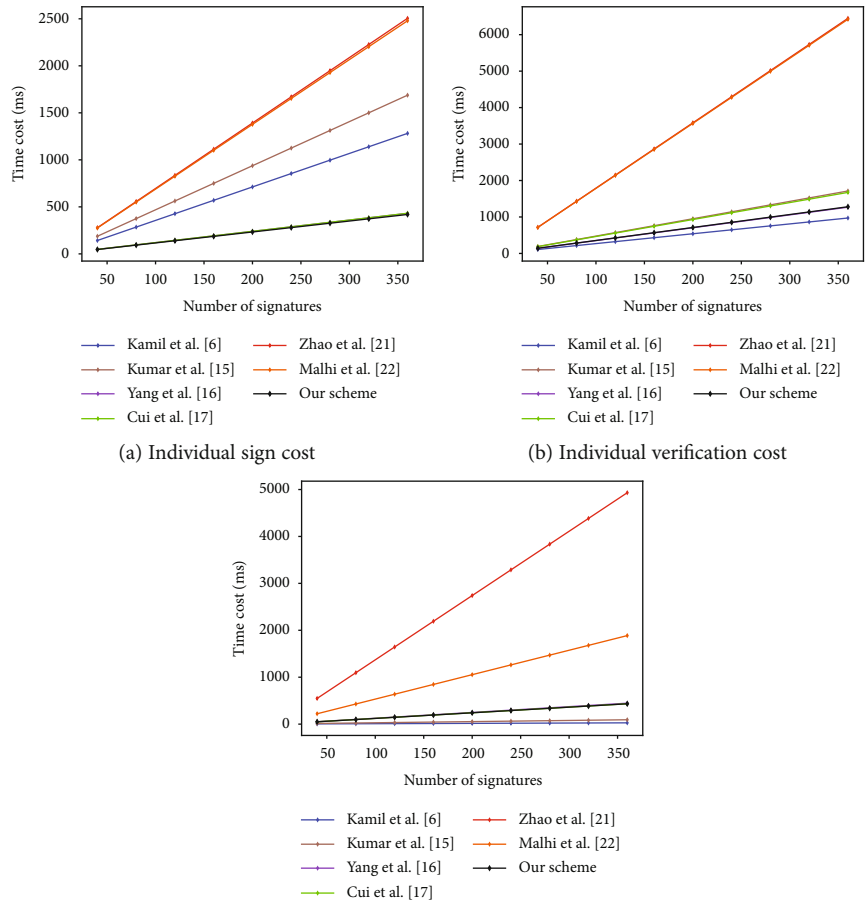
In [13, 24], their schemes choose to use bilinear pairing, which significantly increases their operation time. As a contrast, other four schemes [6, 12, 17, 23] do not use bilinear pairing, which can substantially reduce computation time.

In our scheme, $L_i = l_i \cdot P$ uses a scalar multiplication in ECC operation and the calculation of r_i uses a one-way hash function during the individual sign process. In individual verification, we use three scalar multiplication operations for $S_i \cdot P$, $r_i \cdot L_i$, and $n_i \cdot \text{PK}_{\text{KGC}}$, three addition operations, and two one-way hash function operations for the calculations of r_i and n_i. In aggregate verification process, we use $n + 2$ scalar multiplication operations for $\sum_{i=1}^{n} a_i r_i \cdot L_i$, $S \cdot P$, and $\sum_{i=1}^{n} a_i n_i \cdot \text{PK}_{\text{KGC}}$, two addition operations, and 2n one-way hash function operations for each n_i and r_i. By comprision, our scheme has low time complexity and high efficiency. In addition, our scheme can resist coalition attacks, which are a special and security feature that no other scheme has.

We use the data in Table 3 to generate three figures, which can intuitively compare other related schemes with

Table 3: Comparison of our proposed scheme with other related schemes.

Scheme	Individual sign	Individual verify	Aggregate verify	Coalition attacks resistance	With pairing
Kamil et al. [6]	$3T_{e-m}+2T_{e-a}$ $+3T_h = 3.5598ms$	$2T_{e-m}+T_{e-a}+T_h = 2.6988ms$	$2T_{e-m} + nT_{e-a}+nT_h = (0.0699n + 2.2899)ms$	No	No
Kumar et al. [12]	$4T_{e-m}+2T_{e-a}$ $+3T_h = 4.6878ms$	$4T_{e-m}+3T_{e-a}+4T_h = 4.7577ms$	$4T_{e-m}+3nT_{e-a}+4nT_h = (0.2457n + 4.512)ms$	No	Yes
Yang et al. [13]	$4T_{bp-m}+2T_{bp-a}$ $+3T_h = 6.9582ms$	$3T_{bp}+3T_{bp-m}+T_{bp-a}$ $+4T_h = 17.9111ms$	$2nT_{bp}+3nT_{bp-m}+nT_{bp-a}+4nT_h = 13.7001nms$	Yes	Yes
Cui et al. [17]	$T_{e-m} + T_{e-a} + T_h = 1.1979ms$	$3T_{e-m} + 2T_{e-a} + 2T_h = 3.5238ms$	$(n+2)T_{e-m} + (n+2)T_{e-a} + 2nT_h = (1.2339n + 2.3238)ms$	No	No
Zhao et al. [23]	$T_{e-m}+2T_h = 1.2ms$	$4T_{e-m}+3T_{e-a} + T_h = 4.6497ms$	$(n+2)T_{e-m} + (n+3)T_{e-a} + nT_h = (1.1979n + 2.3577)ms$	No	No
Malhi et al. [24]	$4T_{bp-m}+2T_{bp-a}$ $+T_h = 6.8862ms$	$3T_{bp}+3T_{bp-m}+T_{bp-a}$ $+2T_h = 17.8391ms$	$3T_{bp}+3nT_{bp-m}+nT_{bp-a}+2nT_h = (5.2061n + 12.633)ms$	No	Yes
Our scheme	$T_{e-m}+T_h$=1.164ms	$2T_{e-m}+3T_{e-a}+3T_h = 3.5577ms$	$(n+1)T_{e-m}+2T_{e-a}+2nT_h = (1.2n + 1.1958)ms$	Yes	No



(a) Individual sign cost

(b) Individual verification cost

(c) Aggregate verification cost

Figure 4: Time cost of different schemes in three cases.

our scheme. In Figures 4(a)–4(c), we can get the conclusion that our scheme has a considerably low delay in sign and verification procedure, which reveals that our scheme has a much higher efficiency.

7. Conclusion

Since real application scenarios of VANETs require high efficiency, an efficient certificateless-based anonymous authentication and aggregate signature scheme are proposed. The proposed CLS and its improved scheme CL-AS are appropriate for VANETs duo to analysis and testing. In addition, there is still some work to do in the future such as the low efficiency caused by the illegitimate signature in the aggregate verification process.

Data Availability

The proposed algorithm and its comparison rely on theoretical analysis. No additional test data sets are required in this paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Key International Cooperation Projects of the National Natural Science Foundation of China (No. 61520106007).

References

[1] S. Bitam, A. Mellouk, and S. Zeadally, "Vanet-cloud: a generic cloud computing model for vehicular ad hoc networks," *IEEE Wireless Communications*, vol. 22, no. 1, pp. 96–102, 2015.

[2] C. Lin, D. He, X. Huang, N. Kumar, and K. R. Choo, "Bcppa: a blockchain-based conditional privacy-preserving authentication protocol for vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020.

[3] A. Kumar, G. Thangavel, and K. Subba, "An approach to identify the Sybil attacks in vehicular ad-hoc networks using path signature," *International Journal of Scientific & Technology Research*, vol. 9, pp. 3412–3415, 2020.

[4] Y. Jiang, S. Ge, and X. Shen, "Aaas: an anonymous authentication scheme based on group signature in vanets," *IEEE Access*, vol. 8, pp. 98986–98998, 2020.

[5] Y. Zheng, G. Chen, and L. Guo, "An anonymous authentication scheme in vanets of smart city based on certificateless group signature," *Complexity*, vol. 2020, 7 pages, 2020.

[6] I. A. Kamil and S. O. Ogundoyin, "An improved certificateless aggregate signature scheme without bilinear pairings for vehicular ad hoc networks," *Journal of Information Security and Applications*, vol. 44, pp. 184–200, 2019.

[7] J. P. Hubaux, S. Capkun, and J. Luo, "The security and privacy of smart vehicles," *IEEE Security & Privacy*, vol. 2, no. 3, pp. 49–55, 2004.

[8] Z. Jianhong, X. Min, and L. Liying, "On the security of a secure batch verification with group testing for vanet," *International Journal of Network Security*, vol. 16, no. 5, pp. 351–358, 2014.

[9] R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen, "Ecpp: efficient conditional privacy preservation protocol for secure vehicular communications," in *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, pp. 1229–1237, Orlando, FL, USA, 2008.

[10] J. K. Liu, J. Baek, J. Zhou, Y. Yang, and J. W. Wong, "Efficient online/offline identity-based signature for wireless sensor network," *International Journal of Information Security*, vol. 9, no. 4, pp. 287–296, 2010.

[11] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Advances in Cryptology - ASIACRYPT 2003*, C.-S. Laih, Ed., pp. 452–473, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[12] P. Kumar, S. Kumari, V. Sharma, X. Li, A. K. Sangaiah, and S. K. H. Islam, "Secure cls and cl-as schemes designed for vanets," *The Journal of Supercomputing*, vol. 75, no. 6, pp. 3076–3098, 2019.

[13] X. Yang, C. Chen, T. Ma, Y. Li, and C. Wang, "An improved certificateless aggregate signature scheme for vehicular ad-hoc networks," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 2018.

[14] K. Hashimoto and W. Ogata, "Unrestricted and compact certificateless aggregate signature scheme," *Information Sciences*, vol. 487, pp. 97–114, 2019.

[15] Y. Xie, X. Li, S. Zhang, and Y. Li, "iclas: an improved certificateless aggregate signature scheme for healthcare wireless sensor networks," *IEEE Access*, vol. 7, pp. 15170–15182, 2019.

[16] H. Du, Q. Wen, and S. Zhang, "An efficient certificateless aggregate signature scheme without pairings for healthcare wireless sensor network," *IEEE Access*, vol. 7, pp. 42683–42693, 2019.

[17] J. Cui, J. Zhang, H. Zhong, R. Shi, and Y. Xu, "An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks," *Information Sciences*, vol. 451-452, pp. 1–15, 2018.

[18] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, vol. 13, no. 3, pp. 361–396, 2000.

[19] S. O. Ogundoyin and S. O. Awoyemi, "Edas: efficient data aggregation scheme for internet of things," *Journal of Applied Security Research*, vol. 13, no. 3, pp. 347–375, 2018.

[20] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[21] M. S. Kakkasageri and S. S. Manvi, "Information management in vehicular ad hoc networks: a review," *Journal of Network and Computer Applications*, vol. 39, pp. 334–350, 2014.

[22] M. Rudack, M. Meincke, and M. Lott, "On the dynamics of ad hoc networks for inter vehicle communications (ivc)," in *proc. ICWN*, vol. 2, Citeseer, 2002.

[23] Y. Zhao, Y. Hou, L. Wang, S. Kumari, M. K. Khan, and H. Xiong, "An efficient certificateless aggregate signature scheme for the internet of vehicles," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 5, pp. 3708–3711, 2020.

[24] A. K. Malhi and S. Batra, "An efficient certificateless aggregate signature scheme for vehicular ad-hoc networks," *Discrete Mathematics and Theoretical Computer Science*, vol. 17, no. 1, pp. 317–338, 2015.

[25] O. S. Oyinlola, "An autonomous lightweight conditional privacy-preserving authentication scheme with provable security for vehicular ad-hoc networks," *International Journal of Computers & Applications*, vol. 42, pp. 196–211, 2020.