An Efficient, Identifiable and Abortable Multi-party Signature Scheme for VANETs

Gang Shen, Zhenhua Han, Weizhi Meng, *Senior Member, IEEE*, and Mingwu Zhang*

***Abstract*—Vehicular ad-hoc networks (VANETs) are networks based on short-range wireless communication technology, mainly used for direct communication between vehicles and interaction with roadside infrastructures. The emergence of VANETs has improved the efficiency and safety of vehicle travel. However, malicious vehicles may intentionally send incorrect messages to mislead other vehicles for personal gain, and this behavior cannot be identified yet. In this paper, we propose an efficient, identifiable and abortable multi-party signature scheme for VANETs. We utilize the property of zero-knowledge proofs to design a method that can efficiently identify malicious vehicles during the signature process, and define a strategy to quickly locate false proofs. Through rigorous security analysis, it is demonstrated that the proposed scheme satisfies essential security requirements for VANETs, including message unlinkability, vehicle anonymity, malicious traceability, message authentication, collusion resistance, replay attack resistance, and identifiable abort. Performance analysis shows that our scheme is more suitable than existing similar multi-party signature schemes for application in VANETs.**

***Index Terms*—Vehicular ad-hoc networks, identifiable abort, multi-party signature, zero-knowledge proofs**

I. Introduction

As an important part of intelligent transportation systems (ITS), vehicular ad-hoc networks (VANETs) are networks based on short-range wireless communication technology to achieve information exchange between vehicles (V2V), vehicles and infrastructure (V2I), and vehicles and pedestrians (V2P) [1]–[3]. By collecting and analyzing real-time information such as the speed, location, and driving status of vehicles, VANETs can not only help drivers perceive the surrounding traffic situation in advance to avoid potential collision risks, but also optimize traffic flow to reduce congestion and improve road traffic efficiency and driving experience. As shown in Fig. 1, the entities in VANETs mainly include cloud platforms (CPs), infrastructure (Infra), roadside units (RSUs), and vehicles (Vs). On-board unit (OBU) is a device installed in vehicles that can provide wireless communication. Usually, these entities use Dedicated Short-Range Communication (DSRC) technology based on the IEEE 802.11p to achieve wireless communication [4], [5].

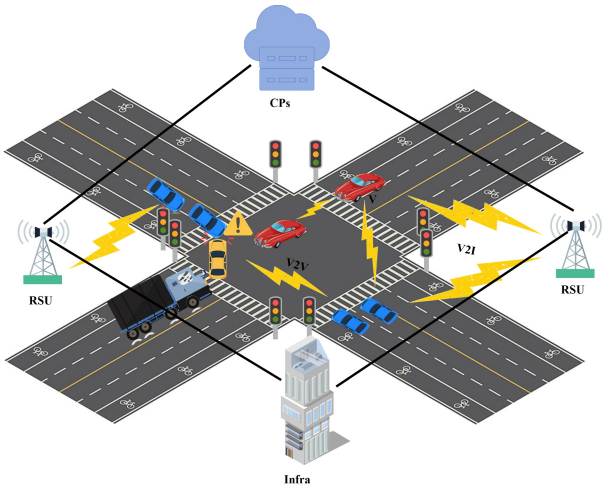


Fig. 1. VANETs architecture.

As the global automotive industry accelerates its shift toward intelligent and connected technologies, the stock of connected vehicles is projected to experience sustained and significant growth going forward [6]. Therefore, open wireless channels, serving as the means of information dissemination, play an essential role in the reliable and stable operation of VANETs. However, with the large-scale deployment of VANETs, the insecure open wireless channels are susceptible to numerous security threats. For instance, adversaries are prone to extracting user' privacy information such as identity, speed and current location from transmitted messages. Armed with this information, malicious adversaries can readily deduce the vehicle owners workplace and home address, frequented locations, and daily habits, potentially leading to numerous complications for the vehicle owner [11]. Moreover, vehicles that have been corrupted by adversaries maliciously inject false information into VANETs to disrupt normal traffic and vehicular services [7]–[10].

To address these issues, researchers have proposed numer-

Manuscript received xx xxx 2025; revised xx xxx 2025 and xx xxx 2025; accepted xx xxx 2025. This work was supported in part by the National Natural Science Foundation of China under Grants 62472150 and 62072134, and the Major Research Plan of Hubei Province under Grant 2023BAA027, and the key Research and Development Program of Hubei Province under Grant 2021BEA163, and the Guiding Program of Scientific Research Plan of Hubei Province under Grant B2023033, and the Natural Science Foundation of Hubei Province under Grant 2023AFB951. *(Corresponding author: Mingwu Zhang)*

Gang Shen, and Mingwu Zhang are with the Hubei Provincial Key Laboratory of Green Intelligent Computing Power Network, the Hubei Provincial Engineering Research Center for Digital & Intelligent Manufacturing Technologies and Applications, the School of Computer Science, Hubei University of Technology, Wuhan 430068, China (e-mail: shengang@hbut.edu.cn; csmwzhang@gmail.com).

Zhenhua Han is with the School of Computer Science, Hubei University of Technology, Wuhan 430068, China (e-mail: hanzhenhua@hbut.edu.cn).

Weizhi Meng is with the School of Computing and Communications, Lancaster University, LA1 4YR Lancaster, U.K. (e-mail: w.meng3@lancaster.ac.uk).

ous authentication schemes for VANETs in recent years. These schemes rely on various cryptographic settings that have been developed, such as traditional public key infrastructure (PKI)-based, identity (ID)-Based, and certificateless [12]–[18]. However, most schemes have encountered challenges in efficiency and security. Additionally, some schemes are not resistant to attacks launched by malicious attackers, such as public key substitution attacks and signature forgery attacks. In order to prevent malicious vehicles from injecting false messages, we propose an efficient, identifiable and abortable multi-party signature scheme for VANETs, which can suspend incorrect vehicle signatures and identify malicious vehicles. Specifically, the main contributions can be summarized in the following three aspects:

1) **Efficient key management and forward security**: The generation of vehicle pseudonyms and partial private keys is delegated to the key generation center (KGC), reducing system redundancy. Additionally, vehicles can derive the private key sk_j for the next time period t_j using the current private key sk_i, the system public key P_{pub}, and t_j. Because compromised keys cannot be used to deduce past keys, the proposed scheme ensures forward security.
2) **Enhanced signature security and verification efficiency**: The signature process incorporates timestamps and signature masks to bolster security. Timestamps ensure signature uniqueness, preventing adversaries from reusing old signatures to forge new ones even if private keys are compromised. Signature masks enable distinct signatures for identical messages and timestamps, enhancing randomness and security. To optimize verification, all commitments and proofs are aggregated, and a rapid localization method is designed to identify faulty proofs when verification fails.
3) **Proven security and performance superiority**: The security of our scheme is rigorously proven under the random oracle model (ROM). Theoretical analysis and simulations demonstrate that the proposed scheme can protect vehicles' privacy, verify the legitimacy of vehicle messages, and identify malicious vehicles. Experimental results confirm the scheme's effectiveness, outperforming existing approaches in both computational and communication overhead.

The remaining parts of this paper are organized as follows. In Section II, we discuss the related work. In order to facilitate understanding of the proposed scheme, the preliminaries are introduced in Section III. Subsequently, in Section IV, the system model and adversary model are expounded upon. Next, the construction details of our scheme and the security analysis are furnished in Section V. The experimental results are provided in Section VI. Finally, Section VII wraps up and summarizes this paper.

II. Related work

The rapid evolution of the Internet of Vehicles (IoV) has intensified security demands for VANETs, where high mobility, dynamic topologies, and vulnerable wireless channels create unprecedented challenges [19]–[21]. Among critical security requirements including authentication, privacy preservation, real-time responsiveness, and trust management, digital signatures serve as the bedrock for ensuring message authenticity and integrity [22]–[24]. The SM2 elliptic curve algorithm, Chinese national cryptographic standard, emerges as a compelling scheme. With short keys and higher hardware efficiency, SM2's resistance to fault injection attacks makes it particularly suitable for resource-constrained V2X communications [25].

Initially, SM2 applications focused on two-party scenarios. Han et al. [26] reduced online computation overhead by replacing homomorphic encryption and secret sharing with Beaver multiplication triples, but their approach requires a trusted third party to pre-distribute the triples and incurs relatively high communication overhead. Li et al. [27] achieved the first non-interactive online signing for two-party SM2 using MtA resharing. However, their scheme incurs significant offline computation overhead and has unproven multi-party extensibility. As IoV systems have evolved toward multi-vehicle collaboration, these schemes have certain limitations in scalability and adaptability.

In order to reduce the complexity introduced by extending SM2 to multi-party signatures, Yu et al. [28] proposed a lightweight batch verification scheme for SM2, which significantly improved efficiency. However, the scalability of this scheme is limited as exceeding 8 signatures can lead to a significant increase in storage costs. In addition, scheme [29] replaced zero-knowledge proofs with lightweight MACs via SPDZ protocols, but this made the scheme more dependent on trusted cloud servers.

Preventing security issues created by malicious individuals is further challenge. Cui et al. [30] implemented accountability through GQ signatures, which can detect signers with malicious behavior. However, their scheme suffered from inefficient class group operations. Scheme [31] used Paillier encryption technology with threshold to achieve identifiable aborts of signatures. However, this scheme also requires a trusted third party to generate the signature key, which introduces a security risk in dynamic VANETs. To ensure the security of signatures, scheme [32] proposed a commitment enforcement verification mechanism based on EdDSA. However, the overhead of exponential proof-checking in this scheme is too high and not suitable for large-scale deployment.

We have summarized the methods, tools, advantages and disadvantages of these signature schemes in Table I. To bridge these gaps, we propose an efficient, identifiable and abortable multi-party signature scheme for VANETs, which can rapid identification of malicious vehicles, eliminate trusted third parties while maintaining sublinear computational complexity.

III. Preliminaries

In this section, we present the relevant knowledge used in the proposed scheme, including the standard SM2 signature algorithm and zero-knowledge proof.

A. The SM2 Signature Algorithm

SM2 signature algorithm [33] is a public key digital signature mechanism constructed based on the principle of elliptic

TABLE I
ADVANTAGES AND DISADVANTAGES OF EXISTING SCHEMES

Schemes	Methods and Tools	Advantages	Disadvantages
Han *et al.* [26]	⋆ Beaver's multiplication triples ⋆ Additive secret sharing ⋆ Provably secure based on standard SM2	⋆ Low computational overhead ⋆ Compatible with original SM2 verification	⋆ Relies on TTP for pre-distributing Beaver triples ⋆ Has communication overhead
Li *et al.* [27]	⋆ MtA resharing technique ⋆ Non-interactive online signing for two-party SM2	⋆ First non-interactive online signing for two-party SM2 ⋆ Low online computation and communication overhead	⋆ Significant offline computation overhead ⋆ Unproven multi-party extensibility ⋆ Relies on Paillier encryption for MtA functionality
Yu *et al.* [28]	⋆ KGLP algorithm and multi-parameter inversion algorithm ⋆ Addition chain semi-numeric computation ⋆ Signature validity judged via congruence equations and resultants	⋆ Significantly improves efficiency ⋆ Lightweight	⋆ Negligible performance gains for 8+ signatures ⋆ Surge in storage costs for 8+ signatures
Li *et al.* [29]	⋆ Secure multi-party SM2 signature scheme ⋆ SPDZ protocol ⋆ Low-overhead MACs	⋆ Replaces zero-knowledge proofs with MACs ⋆ Reduces overall complexity	⋆ Preprocessing relies on a trusted server ⋆ Potential single-point failure ⋆ Trust dependencies
Cui *et al.* [30]	⋆ Trustworthy multi-signature scheme ⋆ GQ signatures with identifiable abort	⋆ Detects malicious or dishonest behavior	⋆ Class group operations less efficient than elliptic curve operations
Liang *et al.* [31]	⋆ Non-interactive SM2 threshold signature ⋆ Identifiable abort ⋆ Paillier homomorphic encryption	⋆ Non-interactive ⋆ Identifiable abort	⋆ Requires Paillier encryption ⋆ Trusted third party for parameter generation
Feng *et al.* [32]	⋆ EdDSA-based multi-party signature scheme ⋆ Commitments & zero-knowledge proofs	⋆ Identifiable abort	⋆ All participants must verify all proofs ⋆ Significant computational overhead

curve cryptography. Its security stems from the computational difficulty of the elliptic curve discrete logarithm problem. SM2 signature algorithm consists of the following four algorithms:

Setup: By inputting the security parameter k, the elliptic curve parameter $params = (p, a, b, G, n)$ is generated and output, where G is the generator point.

KeyGen: Given $params$, select a random number d_A as the user's private key and compute the public key P by performing the operation $P = d_A \cdot G = (x_P, y_P)$. The resulting public-private key pair is (P, d_A).

Sign: Given $params$, d_A, and message M, the signature (r, s) is generated as follows:

1) Compute $Z_A = H_v(\text{ENTL}\|\text{ID}_A\|a\|b\|x_G\|y_G\|x_P\|y_P)$, where ID_A is the user's distinguishable identifier, and ENTL is the length of ID_A. Then, calculate $M' = Z_A\|M$.
2) Calculate $e = H_v(M')$ and convert the text e into an integer, where $H_v(\cdot)$ is a one-way hash function.
3) Select a random number $k \in [1, n-1]$, compute the elliptic curve point $Q = k \cdot G = (x_1, y_1)$, and convert data type of x_1 to an integer.
4) Calculate the signature $r = (e + x_1) \mod n$. If $r = 0$ or $r + k = n$, return to step 3.
5) Compute the signature $s = (1+d_A)^{-1} \cdot (k - r d_A) \mod n$. If $s = 0$, return to step 3; otherwise, output the signature result (r, s).

Verify: Given $params$, public key P, and message M', and signature (r', s'), the verification algorithm proceeds as follows:

1) Check whether $r', s' \in [1, n-1]$. If not, the verification fails.
2) Calculate $M'' = Z_A\|M'$ and compute $e' = H_v(M'')$, converting e' to an integer type.
3) Convert the signature (r', s') to integer types and compute $t = (r' + s') \mod n$. If $t = 0$, the verification fails; otherwise, proceed to step 4.
4) Compute the elliptic curve point $(x_1', y_1') = s' \cdot G + t \cdot P$, converting x_1' to an integer type.
5) Calculate $R = (e' + x_1') \mod n$ and check if $R = r'$. If true, output 1; otherwise, output 0.

B. Zero-Knowledge Proof

Zero-knowledge proof is a cryptographic technique that allows untrusted communication parties to prove the validity of a proposition without revealing any additional information. In this paper, we utilize the NP relation R_{DL} to demonstrate knowledge of the discrete logarithm of elliptic curve points and construct a non-interactive zero-knowledge proof using the Fiat-Shamir heuristic [34], as follows:

$$R_{DL} = \{(Q, w) \in G \times \mathbb{Z}_q : Q = [w] \cdot G\}$$

where G is the generator point of order n.

For R_{DL}, we describe the standard Schnorr proof [35] as follows:

- **Proof**: Given the statement Q, witness w and sample $r \leftarrow [1, n-1]$, compute $R = r \cdot G$, $c = H(G, Q, R)$, and $S = r + c \cdot w \mod q$. Output the proof $\pi = (R, S)$.
- **Verification**: Given a statement (G, q, G, Q) and a proof $\pi = (R, S)$, compute $c' = H(G, Q, R)$ and accept the proof if and only if $S \cdot G = R + c' \cdot Q$.

Here, $H(\cdot) : \{0,1\}^* \to \{0,1\}^\lambda$ denotes a cryptographic hash function with a security parameter λ.

IV. System Model and Adversary Model

In this section, we describe the system model, adversary model and security requirements in detail.

A. System Model

As shown in Fig. 2, the system model consists of five parts, i.e., trusted authority (TA), key generation centre (KGC), roadside unit (RSU), vehicles (Vs) and application server (AS). The specific functions of these entities are as follows:

Trusted Authority (TA): TA is a trusted institution that can register entities and verify the integrity of messages in VANETs. It can also trace the true identity of vehicles.

Key Generation Centre (KGC): KGC is a semi-trusted agency whose responsibility is to generate a pseudonym and a partial private key for each vehicle.

Roadside Unit (RSU): RSU is a communication base station deployed on roadside, serving as a communication bridge between entities.

Vehicles (Vs): Vehicles are equipped with an OBU that can exchange data with RSUs and infrastructures.

Application Server (AS): AS has sufficient resources to collect and analyse the information sent by RSUs and provide additional services to the vehicles. In addition, AS is responsible for verifying the validity of the aggregated signatures provided by vehicles.

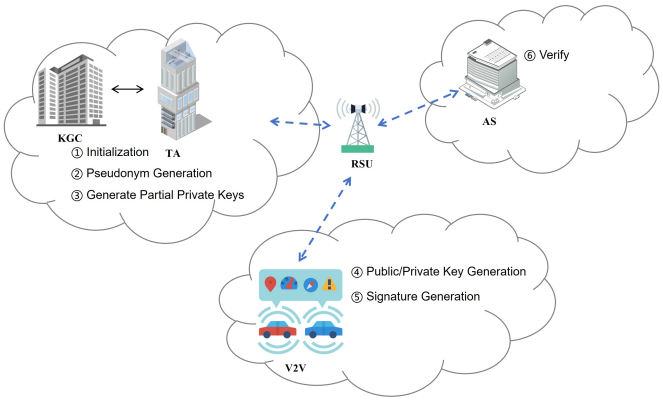


Fig. 2. System model.

B. Adversary Model

In this paper, we consider two types of adversaries: *AdvI* and *AdvII*. *AdvI*, also known as external adversaries, cannot access KGC's master key. However, they have the ability to tamper with any user's public key information. While *AdvII*, usually considered as internal adversaries, can access KGC's master key but cannot replace any user's public key.

For *AdvI* and *AdvII*, we use *Game I* and *Game II* respectively to prove the unforgeability of the signatures. Here are the details of the two games:

Game I is played between a polynomial time algorithm $\mathcal{B}$ and an adversary *AdvI*. $\mathcal{B}$ is a challenger that simulates the environment of *AdvI*. The interaction between *B* and *AdvI* is as follows:

Initialization: $\mathcal{B}$ first executes the initialization algorithm to generate the master key and the public parameters (*params*) required by the system. Subsequently, $\mathcal{B}$ keeps the master key confidential and transmits the *params* to *AdvI*.

Oracle Simulation: *AdvI* adaptively sends the following queries to $\mathcal{B}$ and can receive responses from $\mathcal{B}$.

1) ***Partial-Private-Key Query:*** If *AdvI* issues a query for PID_{V_i}, $\mathcal{B}$ returns a partial private key spk_i to *AdvI*.
2) ***Public-Key Query:*** If *AdvI* issues a query for PID_{V_i}, $\mathcal{B}$ computes public key pk_{V_i} to *AdvI*.
3) ***Secret-Value Query:*** If *AdvI* issues a query for the PID_{V_i}, $\mathcal{B}$ returns the secret value d_i to *AdvI*.
4) ***Replace-Public-Key Query:*** If *AdvI* requests the replacement of the public key pk_{V_i} of PID_{V_i} with the new value pk'_{V_i}, and $\mathcal{B}$ records the replacement.
5) ***Sign Query:*** If *AdvI* submits a query on (PID_{set}, m), $\mathcal{B}$ returns corresponding signature σ to *AdvI*.

Forgery: *AdvI* outputs a tuple $(PID_{set}, m, \sigma^*, PK)$, which includes of signatories, message, signature and shared public key. If the following three conditions are satisfied, then *AdvI* wins the game.

1) The forged signature σ^* must be legitimate, and the target identity PID_{V_i} must in PID_{set}.
2) *AdvI* has never issued a query on the partial private key queries of the target identity $PID^*_{V_i}$.
3) *AdvI* has never issued a query on the tuple (PID_{set}, m) to the sign queries.

Game II is conducted between a polynomial time algorithm $\mathcal{B}$ and the adversary *AdvII*, following these steps:

Initialization: $\mathcal{B}$ executes an algorithm to generate a master key and system public parameters *params*. Upon completing this process, $\mathcal{B}$ securely stores the master key and transmits the public parameters to *AdvII*.

Oracle Simulation: Similar to *Game I*, *AdvII* adaptively poses a series of queries to these prognosticators: ***Public-Key Query***, ***Secret-Value Query***, ***Sign Query***, and $\mathcal{B}$ responds to these queries.

Forgery: *AdvII* outputs a tuple $(PID_{set}, m, \sigma^*, PK)$, which includes of signatories, message, signature and shared public key. If the following three conditions are satisfied, then *AdvII* wins the game.

1) The forged signature σ^* must be legitimate, and the target identity PID_{V_i} must in PID_{set}.
2) *AdvII* has never issued a query on the secret value queries of the target identity $PID^*_{V_i}$.
3) *AdvII* has never issued a query on the tuple (PID_{set}, m) to the sign queries.

C. Security Requirements

The proposed scheme should meet security requirements such as message authentication, integrity, anonymity, traceability, unlinkability, anticomplicity, identifiable abort, and resistance against various attacks. These properties are elaborated below:

Message unforgeability: Messages cannot be forged during transmission.

Anonymity: The true identity of the vehicle must be protected. During the communication process, no entity other than the vehicle itself and TA can obtain the true identity of the vehicle.

Traceability: TA can trace the true identity of vehicles with malicious behavior from pseudonym.

Message authentication and integrity: TA may validate messages sent by the vehicle to ensure that the adversary has not tampered with the message during transmission.

Collusion resistance: As long as there is an honest participant, even if multiple malicious vehicles collude, it is impossible to forge a legitimate signature.

Replay attack resistance: Malicious vehicles cannot duplicate previously sent message to deceive TA.

Identifiable abort: During the signature process, it is possible to accurately identify the participants who caused the signature to be aborted.

V. Proposed scheme

In this section, we propose an efficient, identifiable and abortable multi-party signature scheme applied in VANETs. For ease of understanding, we give the main symbols used in the proposed scheme and their descriptions, as shown in Table II.

TABLE II
Main Symbols and Descriptions

Symbol	Descripition
N	The total number of participating vehicles
α	Master key of KGC
P_{pub}	The system public key
ID_{V_i}	True identity of V_i
PID_{V_i}	Pseudonym of V_i
β_i	The validation part in PID_{V_i}
G	The base point of SM2 elliptic curve
n	The order of base point G
p	The large prime defining the funite filed
T	Current timestamp
spk_{V_i}	Partial private key of V_i
θ_i, X_i	Part of spk_{V_i}
d_{V_i}	Secret value of V_i
sk_{V_i}	The private key of V_i
pk_{V_i}	The public key of V_i
PK	The share public key
π_{V_i}, π'_{V_i}	Zero-knowledge proof of V_i
B_{V_i}, W_{V_i}	Part of zero-knowledge proof
sid	Stage marker
x_i, k_i	Random numbers
m	Message
σ	Signature

A. The Proposed Concrete Scheme

*1) **System Initialization**:* Given a security parameter κ, KGC chooses a finite field of characteristic p and the generator point G of prime order n. Then, KGC generates the system parameters and publishes them into VANETs. The process of system initiation is as follows.

- KGC randomly selects $\alpha \in [1, n-1]$, and computes $P_{pub} = \alpha \cdot G \mod p$. KGC sets α as the master key and P_{pub} as its public key.
- KGC chooses three general hash functions: $H_1 : \{0,1\}^* \to [1, n-1]$, $H_2 : \{0,1\}^* \times \{0,1\}^* \to [1, n-1]$, $H_3 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to [1, n-1]$.
- KGC keeps α privately, and publishes system parameters $params = \{p, n, G, P_{pub}, H_1, H_2, H_3\}$ to VANETs.

*2) **Pseudonym Generation**:* Vehicles must register with TA to access services provided by AS. Additionally, KGC generates pseudonyms for each registered vehicle (V_i) to preserve their real identities, where V_i denotes the i-th vehicle.

- V_i sends its real identity ID_{V_i} to KGC via a secure channel.
- Upon receiving ID_{V_i}, KGC submits a request to TA to obtain V_i's legitimate identity record (denoted as $ID^*_{V_i}$, stored by TA) for legitimacy verification.
- If $ID_{V_i} = ID^*_{V_i}$, KGC computes $AID_{V_i} = ID_{V_i} \oplus H_2(T_i, \alpha)$ and $\beta_i = H_1(AID_{V_i}) \cdot \alpha$ (with T_i as the current timestamp), then sets pseudonym $PID_{V_i} = \{AID_{V_i}, \beta_i, T_i\}$. Otherwise, identifying information ID_{V_i} is recorded and flagged as a suspicious vehicle.
- KGC maintains a mapping table between V_i's real identities and pseudonyms (V_i, PID_{V_i}) and returns PID_{V_i} to V_i though a secure channel.
- V_i loads PID_{V_i} into its OBU.

Note: Vehicles employ time-valid pseudonyms, which are reissued via KGC requests upon expiration, accompanied by private key refreshment to preserve key security.

*3) **Generate Partial Private Keys**:* KGC will generate a partial private key spk_{V_i} for V_i based on its pseudonym. The specific process is as follows:

- KGC verifies whether the equation $\beta_i \cdot G = H_1(AID_i) \cdot P_{pub}$ holds. If it does not hold, ignore it. Otherwise, KGC randomly selects $x_i \in [1, n-1]$, and computes $X_i = x_i \cdot G \mod p$.
- Next, KGC computes $R_i = H_3(PID_{V_i}, X_i, P_{pub})$ and $\theta_i = (x_i + \alpha \cdot R_i) \mod n$.
- KGC sets partial private key $spk_{V_i} = \{X_i, \theta_i\}$, and then spk_{V_i} will be sent to V_i.

*4) **Public/Private Key Generation**:* Upon receiving spk_{V_i}, V_i computes $R'_i = H_3(PID_{V_i}, X_i, P_{pub})$. If $\theta_i \cdot G \neq X_i + R'_i \cdot P_{pub}$, spk_{V_i} is ignored; otherwise, it performs public/private key generation. The details are as follows:

- V_i randomly selects $d_{V_i} \in [1, n-1]$, and computes $D_{V_i} = d_{V_i} \cdot G$.
- V_i sets private key $sk_{V_i} = d_{V_i} + \theta_i$ and public key $pk_{V_i} = sk_{V_i} \cdot G \mod p = D_{V_i} + X_i + P_{pub} \cdot R'_i \mod p$, respectively.
- Next, V_i calls Algorithm 1 to generate zero-knowledge proofs π_{V_i} for sk_{V_i} and pk_{V_i}.

Algorithm 1 : Generate Zero-knowledge-Proof

Input: Q and w, where $Q = w \cdot G \mod p$.
Output:
1: Randomly selects $b \in [1, n-1]$;
2: Computes $B = b \cdot G \mod p$, $c = H_3(G, Q, B)$ and $W = (b + c \cdot w) \mod n$;
3: **Return** the proof $\pi = \{B, W\}$.

*5) **Public Key Aggregation**:*

- V_i sends $(PID_{V_i}, sid_{kg}, \pi_{V_i})$ to all other vehicles. When V_i receives $(PID_{V_j}, sid_{kg}, \pi_{V_j})$ from all other vehicles, it first verifies whether sid_{kg} is fresh or not. If not, it ignores the message; otherwise, it returns $(PID_{V_i}, sid_{kg}, pk_{V_i})$.
- V_i receives $(PID_{V_j}, sid_{kg}, pk_{V_j})$ from all other vehicles, then calculates $B = \sum_{j=1}^{N} B_{V_j} \mod p$ and $W = \sum_{j=1}^{N} W_{V_j}$.
- V_i first computes $c_j = H_3(G, pk_{V_j}, B_{V_j})$, then computes $a = W \cdot G - B - \sum_{j=1}^{N} c_j \cdot pk_{V_j}$. If $a = 0$, V_i computes

$PK = \sum_{j=1}^{N} pk_{V_j}$; otherwise, it aborts the protocol and invokes Algorithm 2 to identify invalid proofs.

*6) **Signature Generation**:* When the vehicle group $V = \{V_1, V_2, \ldots, V_N\}$ needs to transmit message $m \in (0,1)^*$ to nearby communication units, each vehicle $V_i \in V$ generates a digital signature for m through an interactive process.

- V_i randomly selects $k_{V_i} \in [1, n-1]$, and computes $(x_1, y_1) = K_{V_i} = k_{V_i} \cdot G \mod p$.
- V_i calls Algorithm 1 to generate zero-knowledge proofs π'_{V_i} for k_{V_i} and K_{V_i}, then sends $(PID_{V_i}, sid_k, \pi'_{V_i})$ to all other vehicles. When V_i receives $(PID_{V_j}, sid_k, \pi'_{V_j})$ from all other vehicles, it first verifies whether sid_k is fresh or not. If not, it ignores the message; otherwise, it returns $(PID_{V_i}, sid_k, K_{V_i})$.
- When V_i receives $(PID_{V_j}, sid_k, K_{V_j})$ from all other vehicles, it calculates $B' = \sum_{j=1}^{N} B'_{V_j} \mod p$ and $W' = \sum_{j=1}^{N} W'_{V_j} \mod n$.
- V_i computes $c'_{V_j} = H_3(G, K_{V_j}, B'_{V_j})$, and computes $a = W' \cdot G - B' - \sum_{j=1}^{N} c'_{V_j} \cdot K_{V_j}$. If $a = 0$, V_i computes $K = \sum_{i=1}^{N} K_{V_i}$; otherwise it aborts the protocol and invokes Algorithm 2 to identify invalid proofs.
- Let $(x_1, y_1) = K$, V_i sets $r = x_1 \mod n$, and if r = 0, return to step 1 to reselect k_{V_i}.
- V_i computes $(x_2, y_2) = T \cdot G \mod p$, where T is the current timetamp.
- V_i sets $M = (m||x_2)$, then computes $e = H_1(M)$ and $r' = (r + e) \mod n$.
- V_i computes $S_{V_i} = k_{V_i} + sk_{V_i} \cdot e \mod n$, and sends $(PID_{V_i}, sid_S, S_{V_i})$ to other vehicles.
- V_i receives $(PID_{V_j}, sid_S, S_{V_j})$ from the other vehicle and first verifies whether sid_S is fresh. If it is not fresh, the message is ignored; otherwise it calculates $S = \sum_{j=1}^{N} S_{V_j} \mod n$.
- V_i sets the signature as $\sigma = (r', S)$ and verifies whether $(\sigma, PK, m) = 0$. If the equation holds, abort the protocol.

Algorithm 2 : Invalid Proof Identification

Require: $\{PID_{V_i} \mid i \in N\}, \{\pi_i = \{B_i, W_i\} \mid i \in N\}, \{c_i \mid i \in N\}, \{Q_i \mid i \in N\}$
Ensure:
if there are fewer proofs, validating each proof individually **then**
  for each proof i **do**
    Compute $a^* = W_i \cdot G - B_i - c_i \cdot Q_i$.
    if $a^* = 0$ **then**
      return 1
    else
      Add its PID_{V_i} to the set $\{I\}$.
    end if
  end for
else
  Perform the defined fast localization method for finding, iteratively dividing the message set into 2 smaller subsets:
  Given a set of messages RES, RSU performs an aggregate verification to obtain a^*.
  if $a^* = 0$ **then**
    There are no errors in RES.
  else
    Divide RES into two halves: RES = RES$^+$ $\cup$ RES$^-$.
    Repeat step 1 for RES$^+$ to compute a.
    if a = 0 **then**
      There are no invalid proofs in RES$^+$. Proceed to step 1 for RES$^-$.
    else if $a \neq 0$ and $a = a^*$ **then**
      There are no invalid proofs in RES$^-$. Repeat step 1 for RES$^+$.
    else if $a \neq 0$ and $a \neq a^*$ **then**
      Both parts contain invalid proofs. Repeat steps 1 and 2 for both parts.
    end if
  end if
  Continue this process until all invalid proofs are identified. Add the PID_{V_i} to which the invalid proofs belong to the set $\{I\}$.
  The vehicle then sends the $\{I\}$ to TA for identity disclosure.
end if

*7) **Verify**:* When AS receives (σ, PK, m') from V_i, the legitimacy of the signature is verified.

- Computes $(x'_2, y'_2) = T'' \cdot G \mod p$, and sets $M' = (m'||x'_2)$.
- Computes $e' = H_1(M')$ and $r'' = (r' - e') \mod n$.
- Computes $(x'_1, y'_1) = S \cdot G - e' \cdot PK$.
- Let $r^* = x'_1 \mod n$, check whether $r^* = r''$ holds. If it does then **return** 1; otherwise **return** 0.

If the signature is valid, then we have

$$
\begin{aligned}
(x_1', y_1') &= S \cdot G - e' \cdot PK \\
&= \sum_{i=1}^{N} S_{V_i} \cdot G - e' \cdot PK \\
&= \sum_{i=1}^{N} (k_{V_i} + sk_{V_i} \cdot e) \cdot G - e' \cdot PK \\
&= \sum_{i=1}^{N} (k_{V_i} \cdot G + e \cdot sk_{V_i} \cdot G) - e' \cdot PK \\
&= K + e \cdot PK - e' \cdot PK \\
&= K = (x_1, y_1)
\end{aligned}
$$

Therefore, $r'' = r^*$ is hold, and the proposed scheme satisfies the correctness.

B. Security Analysis

In this part, we establish the security of the proposed scheme through a formal proof of existential unforgeability under adaptive chosen-message attacks (EUF-CMA) within ROM. This proof demonstrates that the scheme satisfies the essential security requirements for VANET deployments.

Theorem 1: Under ECDLP problem assumptions, the proposed scheme can resist existential forgery attacks.

Lemma 1: Assuming that there exists a type I adversary *AdvI* that can successfully forge signatures with non-negligible probability ε, we can construct an algorithm $\mathcal{B}$ to solve the ECDLP problem with success probability associated with ε.

Proof: Given an instance of ECDLP problem $(G, U = u \cdot G)$, the aim of $\mathcal{B}$ is to compute a to solve the ECDLP problem, where u is unknown. For this, sets $PID_{V_l}^*$ as the target identity.

Initialization: Challenger $\mathcal{B}$ first runs the initialization algorithm to generate master key and system parameters. Subsequently, $\mathcal{B}$ securely keeps the master key and transmits the system parameters $params = \{p, n, G, P_{pub}, H_1, H_2, H_3\}$ to *AdvI*. $\mathcal{B}$ keeps originally empty lists: 1) L_{H_1}; 2) L_{H_3}; 3) L_{spk}; 4) L_{pk}; 5) L_{sig}.

Queries Phase: *AdvI* can make the following query, and $\mathcal{B}$ responds as follows:

H_1 ***Query:*** $\mathcal{B}$ keeps a list $L_{H_1} = \{(m \parallel R, h_{1i})\}$. When *AdvI* queries a tuple $(m \parallel R)$ for this randomized prediction, $\mathcal{B}$ first checks whether it has been saved in L_{H_1}. If it is, $\mathcal{B}$ sends h_{1i} to *AdvI*; otherwise, $\mathcal{B}$ randomly chooses $h_{1i} \in [1, n-1]$, sets $h_{1i} = H_1(m \parallel R)$, and adds the tuple $(m \parallel R, h_{1i})$ to L_{H_1}. Then it sends h_{1i} to *AdvI*.

H_3 ***Query:*** $\mathcal{B}$ keeps a list $L_{H_3} = \{(PID_{V_i}, X_i, P_{pub}, h_{3i})\}$. When *AdvI* queries PID_{V_i} for this randomized prediction, $\mathcal{B}$ first checks whether it has been saved in L_{H_3}. If it is, $\mathcal{B}$ sends h_{3i} to *AdvI*; otherwise, $\mathcal{B}$ randomly chooses $x_i \in [1, n-1]$, computes $X_i = x_i \cdot G$ and $h_{3i} = H_3(PID_{V_i}, X_i, P_{pub})$, then sends h_{3i} to *AdvI* and adds the tuple (PID_{V_i}, X_i, h_{3i}) to L_{H_3}.

Partial-Private-Key Query: $\mathcal{B}$ holds a list $L_{spk} = \{(PID_{V_i}, X_i, \theta_i)\}$. When *AdvI* submits a query about PID_{V_i}, if $PID_{V_i} = PID_{V_l}^*$, $\mathcal{B}$ aborts Game I and outputs "fault"; otherwise, $\mathcal{B}$ checks whether PID_{V_i} is in list L_{spk}. If it exists, $\mathcal{B}$ will return $spk_{V_i} = (X_i, \theta_i)$ to *AdvI*. Otherwise, $\mathcal{B}$ randomly selects $\theta_i, x_i \in [1, n-1]$, then computes $X_i = x_i \cdot G$ and $h_{3i} = H_3(PID_{V_i}, X_i, P_{pub})$, and sets $spk_{V_i} = (X_i, \theta_i)$. Finally, $\mathcal{B}$ sends spk_{V_i} to *AdvI*, stores (PID_{V_i}, X_i, h_{3i}) in the list L_{H_3}, and adds the tuple $(PID_{V_i}, X_i, \theta_i)$ to L_{spk}.

Public-Key Query: $\mathcal{B}$ keeps a list $L_{pk} = \{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$. When *AdvI* submits a query to B with PID_{V_i}, and $\mathcal{B}$ will check whether PID_{V_i} is in list L_{pk}. If it exists, $\mathcal{B}$ recovers the tuple $(PID_{V_i}, d_i, \theta_i, pk_{V_i})$ and return pk_{V_i} to *AdvI*, or else, $\mathcal{B}$ proceeds as below.

- If $PID_{V_i} = PID_{V_l}^*$, $\mathcal{B}$ randomly selects $d_i, \theta_i \in [1, n-1]$ and lets $pk_{V_i} = (d_i + \theta_i) \cdot G \mod p$. Then $\mathcal{B}$ sends pk_{V_i} to *AdvI* and adds the tuple $\{PID_{V_i}, d_i, \perp, pk_{V_i}\}$ to the list L_{pk}.
- If $PID_{V_i} \neq PID_{V_l}^*$, $\mathcal{B}$ recovers the tuple $(PID_{V_i}, X_i, \theta_i)$ from L_{spk}, and randomly selects $d_i \in [1, n-1]$, lets $pk_{V_i} = (d_i + \theta_i) \cdot G \mod p$. Finally $\mathcal{B}$ sends pk_{V_i} to *AdvI* and adds the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ to the list L_{pk}.

Secret-Value Query: When *AdvI* make a query to this oracle with PID_{V_i}, if $PID_{V_i} = PID_{V_l}^*$, $\mathcal{B}$ aborts this simulation and outputs"fault"; otherwise, $\mathcal{B}$ proceeds as follows: if PID_{V_i} is already exists in L_{pk}, $\mathcal{B}$ recovers the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ from L_{pk} and sends the secret value d_i to *AdvI*; otherwise, $\mathcal{B}$ makes a query to the ***Public-key Queries*** on PID_{V_i} to get d_i and saves the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ to L_{pk}. In the end, $\mathcal{B}$ sends d_i to *AdvI*.

Replace-Public-Key Query: When *AdvI* submits this request to $\mathcal{B}$ with (PID_{V_i}, pk'_{V_i}). $\mathcal{B}$ checks whether the PID_{V_i} exists in the list L_{pk}. If it does, replaces $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ with $(PID_{V_i}, \perp, \perp, pk'_{V_i})$.

Sign Query: *AdvI* submits a request with (PID_{set}, m) to $\mathcal{B}$ to obtain the aggregated signature of m. Firstly, for each $PID_{V_i} \in PID_{set}$, $\mathcal{B}$ restores pk_{V_i} form L_{pk} and computes the aggregated public key as $PK = \sum_{i=1}^{N} pk_{V_i}$. Secondly, for each $PID_{V_i} \in PID_{set}$, $\mathcal{B}$ selects a random number $k_{V_i} \in [1, n-1]$ and computes $K_{V_i} = k_{V_i} \cdot G \mod p$. By interacting with other participants, $\mathcal{B}$ obtains $(x_1, y_1) = K = \sum_{j=1}^{N} K_{V_j}$ and sets $r = x_1 \mod n$. Next $\mathcal{B}$ computes $(x_2, y_2) = T' \cdot G \mod p$, $M = (m || x_2)$, $e = H_1(M)$ and $r' = (r + e) \mod n$. Thirdly, for each $PID_{V_i} \in PID_{set}$, if $PID_{V_i} \neq PID_{V_l}^*$, $\mathcal{B}$ recovers the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ from L_{pk} and computes $S_{V_i} = k_{V_i} + (\theta_i + d_i) \cdot e$. If $PID_{V_i} = PID_{V_l}^*$, $\mathcal{B}$ randomly selects $d_i, \theta_i \in [1, n-1]$ and lets $S_{V_i} = k_{V_i} + (\theta_i + d_i) \cdot e$. Finally, $\mathcal{B}$ aggregates all partial signatures by computing $S = \sum_{j=1}^{N} S_{V_j} \mod n$, setting the aggregated signature as $\sigma = (r', S)$, and return (PK, σ) to *AdvI*. Additionally, $\mathcal{B}$ stores $(PID_{set}, m, PK, \sigma)$ into list L_{sig}.

Forgery: *AdvI* outputs a forged signature $\sigma^* = (r^*, S^*)$ corresponding to message m^* and the identity set PID_{set}^*. *AdvI* is successful if all of the following conditions are met:

- Eve_1**:** *Verify*(m^*, σ^*, PK^*) = ***true***, where PK^* is the shared public key generated from PID_{set}^*.
- Eve_2**:** During the queries phases, Game I cannot be aborted.
- Eve_3**:** During the forgery phase, the signature of the target identity $PID_{V_l}^*$ must be forged by *AdvI* and

aggregated in the forged aggregate signature.

- Eve_4**:** *AdvI* has not made a ***Sign Query*** on(PID^*_{set}, m^*).

Probability analysis: If *AdvI* is successful, the equation $S^* \cdot G = (r^* - e^*) + e^* \cdot PK$ holds true, where $e^* = H_1(m^*||x_2^*)$, $x_2^* = T^* \cdot G$, and T^* is the timestamp used in the forged signature. Since $pk^*_{V_i} = U = D_i^* + X_i^* + u \cdot R^* \cdot G$, we can rewrite the equation as:

$$\begin{aligned} S^* \cdot G &= (r^* - e^*) \cdot G + e^* \cdot (D_i^* + X_i^* + u \cdot R^* \cdot G + \textstyle\sum_{j\neq i}^{N} pk_{V_j}) \\ e^* \cdot u \cdot G \cdot R^* &= (S^* - r^* + e^*) \cdot G - e^* \cdot (D_i^* + X_i^* + \textstyle\sum_{j\neq i}^{N} pk_{V_j}) \\ e^* \cdot u \cdot G &= \left((S^* - r^* + e^*) \cdot G - e^* \cdot (D_i^* + X_i^* + \textstyle\sum_{j\neq i}^{N} pk_{V_j})\right) \cdot R^{*-1} \\ u &= \left[(S^* - r^* + e^*) - e^* \cdot \left(\tfrac{D_i^* + X_i^* + \sum_{j\neq i}^{N} pk_{V_j}}{G}\right)\right] \cdot (e^* \cdot R^*)^{-1} \end{aligned}$$

Then the probability that $\mathcal{B}$ succeeds in solving the ECDLP problem is $\omega \geq \varepsilon(\varepsilon/q_{H_1} - 1/n) \cdot (1 - (q_{ppk} + q_{sv})/N) \cdot (1/N) \cdot (1 - 1/n)^{q_s}$. The probability that $AdvI$ solves ECDLP is represented by the joint event probability: $\Pr[\text{Eve}_1 \wedge \text{Eve}_2 \wedge \text{Eve}_3 \wedge \text{Eve}_4] = \Pr[\text{Eve}_1] \cdot \Pr[\text{Eve}_2] \cdot \Pr[\text{Eve}_3] \cdot \Pr[\text{Eve}_4]$. Clearly, $\Pr[\text{Eve}_1] > \varepsilon(\varepsilon/q_{H_1} - 1/n)$, where q_{H_1} is the number of queries by Adv_1 to the oracle H_1, and n is the prime order of the field group; $\Pr[\text{Eve}_2] > (1 - (q_{ppk} + q_{sv})/N)$, where q_{ppk} is the maximum number of partial private key queries, q_{sv} is the maximum number of secret value queries that *AdvI* may perform during the whole attack process; $\Pr[\text{Eve}_3] > 1/N$ where N is the total number of signatures to be aggregated; $\Pr[\text{Eve}_4] > (1 - 1/n)^{q_s}$ where q_s is the maximum number of sign queries. Nevertheless, it is intractable for the ECDLP problem to be solved in polynomial time. There, the proposed scheme is EUF-CMA secure against type I adversary *AdvI* in ROM.

Lemma 2: If an adversary of type II (*AdvII*) can forge a valid aggregate signature with non-negligible probability ε, then $\mathcal{B}$ can solve the ECDLP Problem with non-negligible probability.

Proof: We construct an algorithm $\mathcal{B}$ which receives as input the ECDLP instance $(G, U = u \cdot G)$ with the goal of computing u. $\mathcal{B}$ solves the ECDLP by modeling the environment of *AdvII*. For this, sets $PID^*_{V_l}$ as the target identity.

Initialization: Challenger $\mathcal{B}$ runs the initialization algorithm to create master key and system parameters, then $\mathcal{B}$ sends the master key and the system parameters $params = \{p, n, G, P_{pub}, H_1, H_2, H_3\}$ to *AdvII*. $\mathcal{B}$ maintains initially empty lists: 1) L_{H_1}; 2) L_{H_3}; 3) L_{spk}; 4) L_{pk}; 5) L_{sig}.

Queries Phase: In this phase, *AdvII* adaptively asks a series of queries and $\mathcal{B}$ responds to them.

H_1 ***Query:*** $\mathcal{B}$ keeps a list $L_{H_1} = \{(m \parallel R, h_{1i})\}$. When *AdvII* queries a tuple $(m \parallel R)$ for this randomized prediction, $\mathcal{B}$ first checks whether it has been saved in L_{H_1}. If it is, $\mathcal{B}$ sends h_{1i} to *AdvII*; otherwise, $\mathcal{B}$ randomly chooses $h_{1i} \in [1, n-1]$, sets $h_{1i} = H_1(m \parallel R)$, and adds the tuple $(m \parallel R, h_{1i})$ to L_{H_1}. Then it sends h_{1i} to *AdvII*.

H_3 ***Query:*** $\mathcal{B}$ keeps a list $L_{H_3} = \{(PID_{V_i}, X_i, P_{pub}, h_{3i})\}$. Upon receiving a query for PID_{V_i} from *AdvII* for the purpose of randomized prediction, $\mathcal{B}$ first verifies whether this query has already been recorded in L_{H_3}. If it exists, $\mathcal{B}$ relays h_{3i} back to *AdvII*. If not, $\mathcal{B}$ selects a random $x_i \in [1, n-1]$, computes $X_i = x_i \cdot G$ and $h_{3i} = H_3(PID_{V_i}, X_i, P_{pub})$. Afterward, $\mathcal{B}$ forwards h_{3i} to *AdvII* and appends the tuple (PID_{V_i}, X_i, h_{3i}) into L_{H_3}.

Public-Key Query: $\mathcal{B}$ keeps a list $L_{pk} = \{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$. When *AdvII* submits a query to $\mathcal{B}$ with PID_{V_i}, and $\mathcal{B}$ will check whether PID_{V_i} is in list L_{pk}. If it exists, $\mathcal{B}$ recovers the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ and return pk_{V_i} to *AdvII*, or else, $\mathcal{B}$ proceeds as below.

- If $PID_{V_i} = PID^*_{V_l}$, $\mathcal{B}$ randomly selects $d_i, \theta_i \in [1, n-1]$ and lets $pk_{V_i} = (d_i + \theta_i) \cdot G \mod p$. Then $\mathcal{B}$ sends pk_{V_i} to *AdvII* and adds the tuple $\{PID_{V_i}, d_i, \perp, pk_{V_i}\}$ to the list L_{pk}.
- If $PID_{V_i} \neq PID^*_{V_l}$, $\mathcal{B}$ recovers the tuple $(PID_{V_i}, X_i, \theta_i)$ from L_{spk}, randomly selects $d_i \in [1, n-1]$, lets $pk_{V_i} = (d_i + \theta_i) \cdot G \mod p$. Finally $\mathcal{B}$ sends pk_{V_i} to *AdvII* and adds the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ to the list L_{pk}.

Secret-Value Query: When *AdvII* make a query to this oracle with PID_{V_i}, if $PID_{V_i} = PID^*_{V_l}$, $\mathcal{B}$ aborts this simulation and outputs "fault"; otherwise, $\mathcal{B}$ proceeds as follows: if PID_{V_i} is already exists in L_{pk}, $\mathcal{B}$ recovers the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ from L_{pk} and sends the secret value d_i to *AdvII*; otherwise, $\mathcal{B}$ makes a query to the ***Public-key Queries*** on PID_{V_i} to get d_i and saves the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ to L_{pk}. In the end, $\mathcal{B}$ sends d_i to *AdvII*.

Sign Query: *AdvII* submits a request with (PID_{set}, m) to $\mathcal{B}$ to obtain the aggregated signature of m. Firstly, for each $PID_{V_i} \in PID_{set}$, $\mathcal{B}$ restores pk_{V_i} form L_{pk} and computes the aggregated public key as $PK = \sum_{i=1}^{N} pk_{V_i}$. Secondly, for each $PID_{V_i} \in PID_{set}$, $\mathcal{B}$ selects a random number $k_{V_i} \in [1, n-1]$ and computes $K_{V_i} = k_{V_i} \cdot G \mod p$. By interacting with other participants, $\mathcal{B}$ obtains $(x_1, y_1) = K = \sum_{j=1}^{N} K_{V_j}$ and sets $r = x_1 \mod n$. Next $\mathcal{B}$ computes $(x_2, y_2) = T' \cdot G \mod p$, $M = (m||x_2)$, $e = H_1(M)$ and $r' = (r + e) \mod n$. Thirdly, for each $PID_{V_i} \in PID_{set}$, if $PID_{V_i} \neq PID^*_{V_l}$, $\mathcal{B}$ recovers the tuple $\{PID_{V_i}, d_i, \theta_i, pk_{V_i}\}$ from L_{pk} and computes $S_{V_i} = k_{V_i} + (\theta_i + d_i) \cdot e$. If $PID_{V_i} = PID^*_{V_l}$, B recovers the tuple $\{PID_{V_i}, d_i, \perp, pk_{V_i}\}$ from L_{pk}, then $\mathcal{B}$ randomly chooses $\theta_i \in [1, n-1]$ and calculates $S_{V_i} = k_{V_i} + (\theta_i + d_i) \cdot e$. Finally, $\mathcal{B}$ aggregates all partial signatures by computing $S = \sum_{j=1}^{N} S_{V_j} \mod n$, sets the aggregated signature as $\sigma = (r', S)$, and returns (PK, σ) to *AdvII*. Additionally, $\mathcal{B}$ stores $(PID_{set}, m, PK, \sigma)$ into list L_{sig}.

Forgery: After a series of queries, *AdvII* outputs a forged signature $\sigma^* = (r^*, S^*)$ corresponding to the message m^* and the identity set PID^*_{set}. *AdvII* is successful if all of the following conditions are met:

- Eve_1**:** *Verify*$(m^*, \sigma^*, PK^*) =$ *true*, where PK^* is the shared public key generated from PID^*_{set}.
- Eve_2**:** During the queries phases, Game II cannot be aborted.
- Eve_3**:** During the forgery phase, the signature of the target identity $PID^*_{V_l}$ must be forged by *AdvI* and aggregated in the forged aggregate signature.
- Eve_4**:** *AdvI* has not made a ***Sercet Value Query*** on $PID^*_{V_l}$.

Probability analysis: If *AdvII* is successful, the equation $S^* \cdot G = (r^* - e^*) + e^* \cdot PK$ holds true, where $e^* = H_1(m^*||x_2^*)$, $x_2^* = T^* \cdot G$, and T^* is the timestamp used in the forged

signature. Since $pk^*_{V_i} = U = D^*_i + X^*_i + u \cdot R^* \cdot G$, we can compute $D = \sum_{j=1}^{N} D_j$, $X = \sum_{j=1}^{N} X_j$, $R = \sum_{j=1}^{N} R^*_j$. Utilizing the verification equation:

$$\begin{aligned} S^* \cdot G &= (r^* - e^*) \cdot G + e^* \cdot (D + X + u \cdot R \cdot G) \\ e^* \cdot u \cdot G \cdot R &= (S^* - r^* + e^*) \cdot G - e^* \cdot (D + X) \\ e^* \cdot u \cdot G &= ((S^* - r^* + e^*) \cdot G - e^* \cdot (D + X)) \cdot R^{-1} \\ u &= \left[(S^* - (r^* - e^*)) - e^* \cdot \frac{(D+X)}{G}\right] \cdot (e^* \cdot R)^{-1} \end{aligned}$$

Where u is the solution of the ECDLP instance $U = u \cdot G$. The probability that $\mathcal{B}$ succeeds in solving the ECDLP problem is $\omega \geq \varepsilon \cdot (1 - 1/n)^{q_s + q_{H_1}} \cdot (1/N) \cdot (1 - q_{sv}/N)$. The probability that $AdvII$ solves ECDLP is represented by the joint event probability: $\Pr[\text{Eve}_1 \wedge \text{Eve}_2 \wedge \text{Eve}_3 \wedge \text{Eve}_4] = \Pr[\text{Eve}_1] \cdot \Pr[\text{Eve}_2] \cdot \Pr[\text{Eve}_3] \cdot \Pr[\text{Eve}_4]$. Clearly, $\Pr[\text{Eve}_1] > \varepsilon$, where ε is the probability that *AdvII* succeeds in forging; $\Pr[\text{Eve}_2] > (1 - 1/n)^{q_s + q_{H_1}}$, where n is the prime order of the field group, q_s is the maximum number of sign queries, q_{H_1} is the maximum number of H_1 queries; $\Pr[\text{Eve}_3] > 1/N$, where N is the total number of signatures to be aggregated; $\Pr[\text{Eve}_4] > (1 - q_{sv}/N)$, where q_{sv} is the maximum number of secret value queries. Nevertheless, it is intractable for the ECDLP problem to be solved in polynomial time. Therefore, our proposed scheme is EUF-CMA secure against type II adversary *AdvII* in the ROM.

In summary, the messages in our scheme cannot be forged during transmission.

C. Security Discussion

In this section, we will demonstrate that the proposed scheme meets the following security requirements.

1) *Anonymity:* KGC is responsible for generating a pseudonym (PID_{V_i}) for the vehicle, which is composed of (AID_{V_i}, β_i), where AID_{V_i} is computed from the real identity ID_{V_i} of the vehicle and the secret value α of KGC through the hash function H_2, and β_i is used to verify whether the pseudonym was generated by KGC and has not been tampered with. Specifically, the formulae are $AID_{V_i} = ID_{V_i} \oplus H_2(T_i, \alpha)$ and $\beta_i = H_1(\text{AID}_i) \cdot \alpha$. From this, it can be seen that the anonymity of vehicle depends on their real identity ID_{V_i} and the master key α held by KGC. Therefore, only the vehicle itself and KGC can know the true identity of the vehicle, which ensures the anonymity of the vehicle and prevents identity leakage.
2) *Traceability:* If a malicious user disguises themselves as a regular vehicle to carry out illegal operations and causes a traffic accident, TA can restore their real identity by using his or her pseudonym.
3) *Message authentication:* In our scheme, after a message from one entity is accepted by another entity, the message is validated to ensure the integrity of the message. Moreover, according to the security analysis in the previous section, no adversary can generate a valid signature in polynomial time. Therefore, our scheme meets this security requirement.
4) *Collusion resistance:* As long as there is an honest vehicle, multiple malicious vehicles cannot collude to forge a legitimate signatures. The reason is that in the signature generation phase, the signer needs to generate a zero-knowledge proof $\pi'_{V_j} = (B'_{V_j}, W'_{V_j})$ for signature, and the forged π'_{V_j} by the malicious vehicle cannot make the aggregated verification equation $\sum_{j=1}^{N} W'_{V_j} \cdot G = \sum_{j=1}^{N} B'_{V_j} + \sum_{j=1}^{N} c'_{V_j} \cdot K_{V_j}$ hold. Consequently, our scheme achieves collusion resistance.
5) *Replay attack resistance:* In the signature generation phase, our scheme obtains the current timestamp T_i to compute the signature mask. After the receiver receives the message and the signature, the receiver lifts the signature mask and verifies the legitimacy of the signature by the timestamp T_i. This makes it resistant to replay attacks.
6) *Identifiable abort:* In the signature generation phase, when participants sign the message, they perform a zero-knowledge proof on the interacted information to ensure that the information is correctly calculated. After receiving the signature, the recipient verifies the proof. If the verification fails, the signature will be aborted and the error will be located. This achieves an identifiable abort.

VI. Performance Comparison

In this section, we compare our proposed scheme with state-of-the-art schemes in terms of security features, computational cost and communication overhead. We implement the proposed scheme on a personal computer (with i5-9300H 2.40GHz core, 16 GB RAM, Windows 10 operating system) using the SM2 elliptic curve, which enables 256-bit security. After several tests, the various operation times are averaged and the results are shown in Table III. Considering the relatively small execution times of scalar addition, scalar multiplication and hash functions reflected in the metrics T_{padd} and T_{pmul}, we ignore these times in the overall performance evaluation.

A. Comparison of Security Features

The comparison results of security features are listed in Table IV. Security features mainly include message unforgeability, anonymity, traceability, message authentication (MA), collusion resistance (CR), the ability to resist $AdvI$, the ability to resist $AdvII$, resistance to replay attack (RRA) and identifiable abort (IA). Here, '$\surd$' indicates that the scheme satisfies the corresponding security features, and '$\times$' indicates that the scheme does not satisfy the corresponding security features. According to Table IV, in contrast to the above schemes [32], [33], our scheme can satisfy all the security features.

B. Comparison of Computational Cost

In this section, we compare the computational cost of our scheme with existing elliptic curve-based schemes [32] and [33]. Among them, scheme [32] uses the Ed25519 elliptic curve, which has 128-bit security. Scheme [33] uses the standard SM2 elliptic curve with 256-bit security. Table V illustrates the theoretical analysis results of computational cost. Based on the results in Table V, we plot the computational

TABLE III
NOTATION, DESCRIPTION AND OPERATION TIME

Notation	Description	operation time
$T_{SM2pmul}$	Point multiplication operation time on SM2 elliptic curve	6.980 ms
$T_{SM2padd}$	Point addition operation time on SM2 elliptic curve	0.999 ms
$T_{Ed25519pmul}$	Point multiplication operation time on Ed25519 elliptic curve	58.856 ms
$T_{Ed25519padd}$	Point addition operation time on Ed25519 elliptic curve	1.000 ms

TABLE IV
COMPARISON OF SECURITY FEATURES

Schemes	Message unforgeability	Anonymity	Traceability	MA	CR	Resist *AdvI*	Resist *AdvII*	RRA	IA
Feng et al.s [32]	√	×	×	√	√	√	×	√	×
Xiao et al.s [33]	√	×	√	√	√	√	×	√	√
Our Scheme	√	√	√	√	√	√	√	√	√

TABLE V
COMPARISON OF COMPUTATIONAL COST

Schemes	Signature Generation Cost (ms)	Single Signature Verification Cost (ms)	Aggregate Verification (ms)
Feng et al.s [32]	$11T_{Ed25519pmul} + 3T_{Ed25519padd} \approx 650.416$	$2T_{Ed25519pmul} + T_{Ed25519padd} \approx 114.712$	$2T_{Ed25519pmul} + T_{Ed25519padd}$
Xiao et al.s [33]	$2T_{SM2pmul} \approx 13.960$	$2T_{SM2pmul} + T_{SM2padd} \approx 14.960$	$2nT_{SM2pmul} + nT_{SM2padd}$
Our Scheme	$5T_{SM2pmul} + T_{SM2padd} \approx 35.899$	$3T_{SM2pmul} + T_{SM2padd} \approx 21.940$	$3T_{SM2pmul} + T_{SM2padd}$

TABLE VI
COMPARISON OF COMMUNICATION OVERHEAD

Schemes	Single signature transmit (bytes)	n signature transmit (bytes, n=10)
Feng et al.s [32]	$2\|\mathbb{E}\| = 64$	$2n\|\mathbb{E}\| = 640$
Xiao et al.s [33]	$2\|\mathbb{E}\| = 64$	$n(\|\mathbb{E}\| + n\|\mathbb{E}\|) = 3520$
Our Scheme	$4\|\mathbb{E}\| + \|T_i\| = 132$	$4n\|\mathbb{E}\| + n\|T_i\| = 1320$

costs of signature generation and individual signature verification, as shown in Fig. 3, as well as the computational costs of aggregate verification, as shown in Fig. 4.

As shown in Fig. 3, our scheme is not optimal in terms of the computational costs for generating and verifying an individual signature. However, Fig. 4 illustrates that as the number of vehicles involved in the signature increases, our scheme's aggregate verification cost remains nearly unchanged and is the lowest among all compared schemesparticularly when the number of participating vehicles exceeds 2. Specifically, when $n = 16$, our scheme's aggregate verification cost is approximately 22 ms, compared to around 115 ms for scheme [32] and over 238 ms for scheme [33].

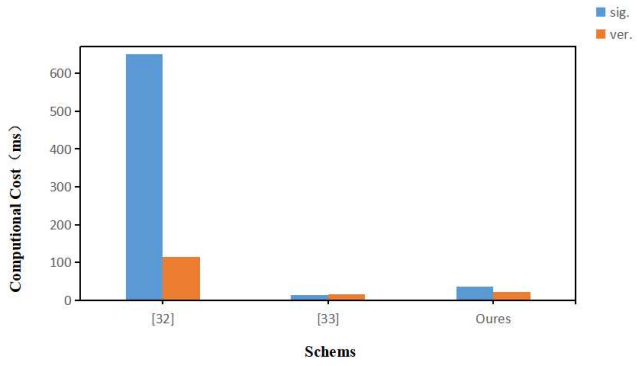


Fig. 3. Computational cost of signature and verification

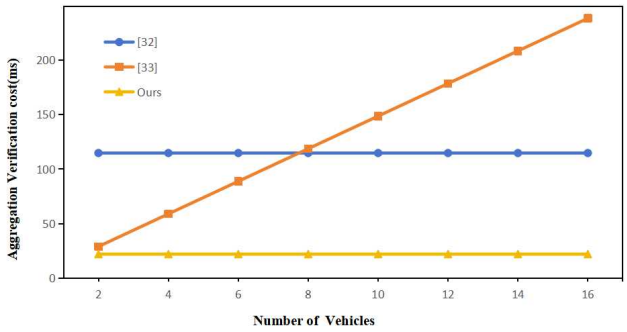


Fig. 4. Computational cost of aggregate verification

C. Comparison of Communication Overhead

In this section, we analyze and compare the communication overhead of our scheme with that of the above schemes. Whether in SM2 elliptic curve or Ed25519 elliptic curve, the length of a scalar is usually $|\mathbb{E}|$= 32 bytes, and it is assumed that the length of a timestamp is $|T_i|$ = 4 bytes. For convenience, we only consider the length of scalars and timestamps in the sent message. For example, in our scheme, the signed message sent by a vehicle to TA is (PID_{V_i}, σ, m), where $PID_{V_i} = \{AID_{V_i}, \beta_i, T_i\}$. Therefore, the length of a message sent from a vehicle to TA is $4|\mathbb{E}| + |T_i|$ = 132 bytes, and the length of message sent simultaneously by multiple vehicles to TA is $4n|\mathbb{E}| + n|T_i|$ = 1320 bytes, where n = 10. Similarly, we calculate the communication overhead of other schemes [32] and [33], and list them in Table VI. To see the

TABLE VII
SIMULATION PARAMETERS

Parameter	Value
Simulation area	2500250050 (m)
Wireless communication protocol	IEEE 802.11p
Simulation duration	300 (s)
Vehicle generation interval	1 (s)
Maximum transmission power	25 (mW)
Transmission range	1500 (m)
Data transfer rate	3 (Mbps)
Data transfer interval	1 (s)

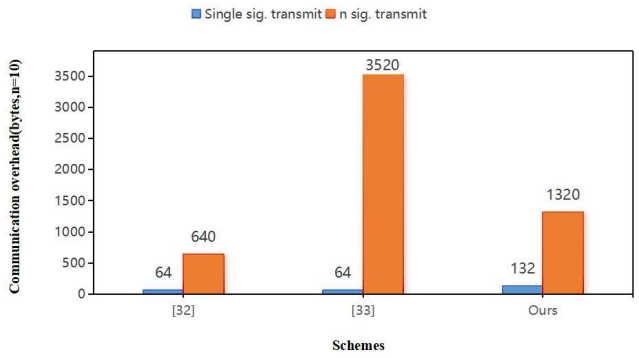


Fig. 5. Communication overhead of signature transmit

Fig. 6. The simulation result

comparison differences more clearly, we draw the communication overhead of the signature transmission diagram according to the results in Table VI, as shown in Fig. 5. As can be seen from Fig. 5, the single signature transmission cost of our proposed scheme is 132 bytes, which is 68 bytes more than that of other schemes. These 68 bytes are the length of the pseudonym, used by TA to trace the source after discovering incorrect signatures, ensuring anonymity, unlinkability, and signature traceability, which can greatly improve the security of the system. The overhead of multi-signature transmission is 1320 bytes (n = 10), which is 206.25% of that of scheme [32] and 37.5% of that of scheme [33]. It adds anonymity, unlinkability, and signature traceability are in terms of security. Compared with the above schemes, although our scheme does not have the lowest communication overhead, it has a great improvement in security. Therefore, our scheme is more suitable for application in VANETs.

D. Simulation

To further evaluate the efficiency of the proposed scheme, we established an experimental environment using Veins 5.2, SUMO 1.11.0, and OMNeT++ 5.7 to simulate communication overhead, as shown in Fig. 6. In addition, some simulation parameters are detailed in Table VII. Among these parameters, “simulation area” defines the 3D spatial extent governing vehicle nodes potential communication range. We configured a 300-second “simulation duration” to fully capture target traffic eventsincluding vehicle interactions and communication processes. With zero initial vehicles, both “vehicle generation interval” and “data transfer interval” were set to 1 second at a “data transfer rate” of 3 Mbps, ensuring sufficient data for analysis. “Transmission range” was set to 1500 meters and “maximum transmission power” of 25 mW guaranteed near-complete message reception at the RSU. The IEEE 802.11p protocol is used for V2X communications, and we modeled only vehicle-to-single-RSU wireless links for simplicity.

Figs. 7 and 8 show that both the total number of uploaded packets and the associated communication overhead increase nearly linearly with vehicle density. On average, each vehicle uploads about 5 packets (approximately 0.535 KB) to RSU. To assess transmission reliability, we measured the packet loss rate by comparing packets sent by vehicles against those successfully received at RSU. As shown in Figs. 9, packet loss remains low in our realistic simulations, ranging from 0.154% to 0.573%. These results demonstrate that the proposed scheme delivers efficient communication performance while maintaining packet delivery reliability at acceptable levels for VANET applications.

E. Limitations and Future Work

Although the proposed scheme has the advantages mentioned above, it still needs to consider the following issues: 1) the multi-round interaction of zero-knowledge proofs leads to high communication overhead; 2) aggregate signatures lack fault tolerance, which means that a single malicious node may paralyze the entire verification process; 3) the scheme only

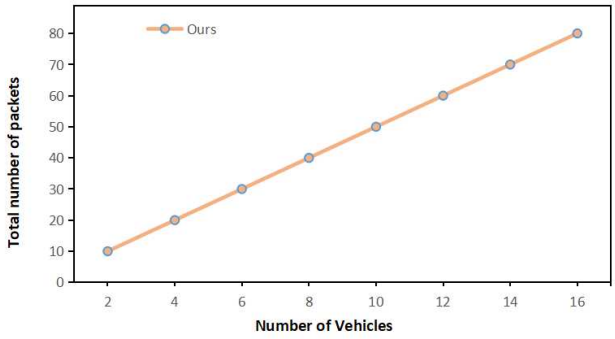


Fig. 7. Vehicle numbers impact on total packets, communication overhead, and packet loss rate

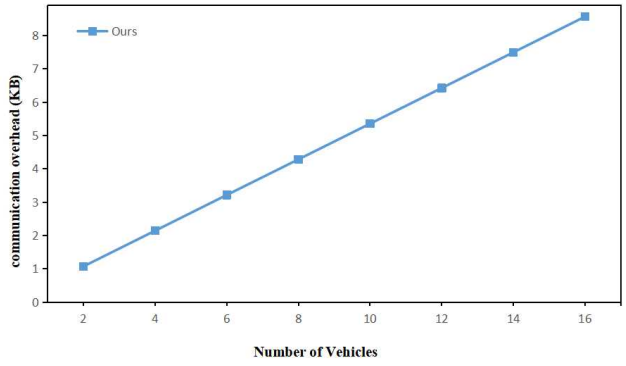


Fig. 8. The communication overhead of the simulation

supports multiple signers signing the same message, which limits its practical applications.

In the future, we will consider combining lightweight interaction methods, asynchronous aggregation mechanisms, and Byzantine fault-tolerance algorithms to improve the efficiency and fault tolerance of the scheme. In addition, it is necessary to consider enabling multiple signers to sign different messages with aggregated verification, in order to meet the requirements of message intensive and resource limited VANETs scenarios.

VII. CONCLUSION

For the purpose of swiftly identifying malicious vehicles in VANETs, this paper proposes an efficient, identifiable and abortable multi-party signature scheme for VANETs. First, a secure and efficient multi-party SM2 signature protocol was designed, which resists internal and external attacks and meets diverse security requirements of VANETs. Next, zero-knowledge proofs were employed to authenticate shared parameters generated during signing, ensuring computational correctness. Additionally, a rapid bisection method was devised to locate invalid proofs, enabling identifiable verification. Finally, under the ECDLP assumption, comprehensive security analysis in the ROM and formal security validation were conducted. Experimental results demonstrated that the proposed scheme achieves lower computational and communication costs compared to existing schemes, with enhanced security and efficiency, making it suitable for VANETs environments with numerous nodes.

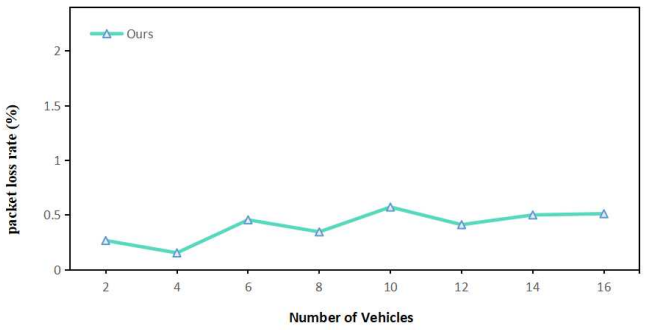


Fig. 9. The packet loss rate of the simulation

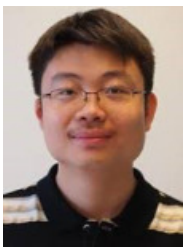
REFERENCES

[1] J. Zhao, F. Huang, L. Liao, and Q. Zhang, "Blockchain-based trust management model for vehicular ad hoc networks," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 8118–8132, Mar. 2024.
[2] G. Shen, K. Xiao, J. Tu, H. Shen, and M. Zhang, "A privacy-preserving and robust aggregation scheme for multi-dimensional data in VANETs," *Comput. Electr. Eng.*, vol. 123, pp. 110–145, Apr. 2025.
[3] B. He and Y. Li, "Blockchain-based key management and security decisions in internet of vehicles," *IEEE Internet Things J.*, vol. 12, no. 11, pp. 17456–17472, Jan. 2025.
[4] M. Azizi and S. Shokrollahi, "RTRV: An RSU-assisted trust-based routing protocol for VANETs," *Ad Hoc Networks*, vol. 154, pp. 103387, Jan. 2024.
[5] X. Ma, and K. S. Trivedi, "SINR-based analysis of IEEE 802.11p/bd broadcast VANETs for safety services," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 3, pp. 2672–2686, Sep. 2021.
[6] M. Ma, B. Chen, D. Tang, M. Deng, T. Xiang, and D. He, "Certificateless searchable public key encryption with trapdoor indistinguishability for IoV," *IEEE Trans. Veh. Technol.*, vol. 74, no. 3, pp. 5085–5096, Mar. 2025.
[7] X. Cao, L. Dang, K. Fan, X. Zhao, Y. Fu, and Y. Luan, "A dynamic and efficient self-certified authenticated group key agreement protocol for VANET," *IEEE Internet Things J.*, vol. 11, no. 17, pp. 29146–29156, Sep. 2024.
[8] M. S. AlMarshoud, M. S. Kiraz, and A. H. Al-Bayatti, "Security, privacy, and decentralized trust management in VANETs: A review of current research and future directions," *ACM Comput. Surv.*, vol. 56, no. 10, pp. 260, Jan. 2024.
[9] C. Chen, L. Wang, and Q. Shi, "A lattice-based certificateless secure data transmission scheme for Internet of Vehicles based-blockchain," *IEEE Trans. Veh. Technol.*, vol. 74, no. 2, pp. 2308–2322, Feb. 2025.
[10] R. Meneguette, R. De Grande, J. Ueyama, G. P. Rocha Filho, and E. Madeira, "Vehicular edge computing: Architecture, resource management, security, and challenges," *ACM Comput. Surv.*, vol. 55, no. 1, pp. 1–46, Jan. 2023.
[11] W. Huang, H. Du, J. Feng, G. Han, and W. Zhang, "A dynamic anonymous authentication scheme with trusted fog computing in V2G networks," *J. Inf. Secur. Appl.*, vol. 79, pp. 103648, Nov. 2023.
[12] J. Shahparian, S. H. Erfani, and A. Zamanifar, "A secure and efficient authentication and key agreement protocol in blockchain-enabled VANETs," *Comput. Electr. Eng.*, vol. 122, pp. 109947, Jan. 2025.
[13] M. Umar, S. K. H. Islam, K. Mahmood, S. Ahmed, Z. Ghaffar, and M. A. Saleem, "Provable secure identity-based anonymous and privacy-preserving inter-vehicular authentication protocol for VANETs using PUF," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 12158–12167, Dec. 2021.
[14] Z. Qiao, K. Ma, Y. Zhou, Q. Yang, Z. Xia, B. Yang, and M. Zhang, "An anonymous and efficient certificate-based identity authentication protocol for VANET," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 11232–11245, Apr. 2024.
[15] Y. Zhou, Z. Wang, Z. Qiao, B. Yang, and M. Zhang, "An efficient and provably secure identity authentication scheme for VANET," *IEEE Internet Things J.*, vol. 10, no. 19, pp. 17170–17183, Oct. 2023.

[16] Y. Zhu, Y. Zhou, J. Wang, B. Yang, and M. Zhang, "A lightweight cross-domain direct identity authentication protocol for VANETs," *IEEE Internet Things J.*, vol. 11, no. 23, pp. 37741–37757, Dec. 2024.
[17] Y. Liang, and Y. Liu, "Analysis and improvement of an efficient certificateless aggregate signature with conditional privacy preservation in VANETs," *IEEE Syst. J.*, vol. 17, no. 1, pp. 664–672, Mar. 2023.
[18] G. Shen, C. Xia, Y. Li, H. Shen, W. Meng, and M. Zhang, "Traceable and privacy-preserving authentication scheme for energy trading in V2G networks," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6664–6676, Feb. 2024.
[19] Q. Hou, C.-F. Hsu, M. H. Au, H. Hu, Z. Zhao, and Z. Wu, "Efficient and provably secure privacy-preserving two-factor authentication and key-agreement using blockchain and TEE for IoV environments," *J. Syst. Archit.*, vol. 164, pp. 103422, Jan. 2025.
[20] H.-T. Lin and W.-L. Jhuang, "Blockchain-based lightweight certificate-less authenticated key agreement protocol for V2V communications in IoV," *IEEE Internet Things J.*, vol. 11, no. 16, pp. 27744–27759, Aug. 2024.
[21] C. Lai, J. Ma, X. Wang, H. Zhou, and D. Zheng, "A novel authentication and key agreement scheme for in-vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 74, no. 6, pp. 9630–9644, Jun. 2025.
[22] Q. Wu, L. Zhang, Y. Yang, and K.-K. R. Choo, "Certificateless signature scheme with batch verification for secure and privacy-preserving V2V communications in VANETs," *IEEE Trans. Dependable Secur. Comput.*, vol. 22, no. 2, pp. 1448–1459, Mar. 2025.
[23] M. A. Karabulut, A. F. M. S. Shah, H. Ilhan, et al., "Inspecting VANET with various critical aspects: A systematic review," *Ad Hoc Networks*, vol. 150, pp. 103281, Nov. 2023.
[24] Z. Xia, L. Zeng, K. Gu, C. Su, H. Hu, and K. Long, "Secure and lightweight vehicular privacy preservation scheme under fog computing-based IoVs," *IEEE Trans. Intell. Veh.*, vol. 9, no. 2, pp. 4115–4129, Apr. 2024.
[25] Z. Ying, K. Wang, J. Xiong, and M. Ma, "A literature review on V2X communications security: Foundation, solutions, status, and future," *IET Commun.*, vol. 18, pp. 16831715, 2024.
[26] G. Han, X. Bai, S. Geng, and B. Qin, "Efficient two-party SM2 signing protocol based on secret sharing," *J. Syst. Archit.*, vol. 132, pp. 102738, Nov. 2022.
[27] S. Li, W. Yang, F. Zhang, X. Huang, and R. Chen, "Practical two-party SM2 signing using multiplicative-to-additive functionality," *Comput. Stand. Interfaces*, vol. 92, pp. 103928, Jan. 2025.
[28] J. Yu, J. Cui, H. Tu, C. Yu, and M. Zhou, "A SM2-based efficient and lightweight batch verification approach for IC cards," *J. Inf. Secur. Appl.*, vol. 73, pp. 103409, Feb. 2023.
[29] X. Li, H. Wang, J. Chen, S. Li, Y. Sun, and Y. Su, "Secure multi-party SM2 signature based on SPDZ protocol," in *Int. Conf. Inf. Secur. Cryptol.*, 2024, pp. 85–103.
[30] H. Cui and T. H. Yuen, "A trustless GQ multi-signature scheme with identifiable abort," in *Proc. 26th Australas. Conf. Inf. Secur. Privacy* , 2021, pp. 673–693.
[31] H. Liang, and J. Chen, "Non-interactive SM2 threshold signature scheme with identifiable abort," *Front. Comput. Sci.*, vol. 18, pp. 181802, Aug. 2024.
[32] Q. Feng, K. Yang, M. Ma, and D. He, "Efficient multi-party EdDSA signature with identifiable aborts and its applications to blockchain," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 1937–1950, Apr. 2023.
[33] Y. Xiao, L. Zhang, Y. Yang, W. Wu, J. Ning, and X. Huang, "Provably secure multi-signature scheme based on the standard SM2 signature scheme," *Comput. Stand. Interfaces*, vol. 89, art. 103819, Aug. 2024.
[34] A. Fiat, and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *CRYPTO 1986*, 1987, pp. 186–194.
[35] C. P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, Jan. 1991.

Gang Shen received the B.S. degree in electrical automation from Wuhan Institute of Technology, Wuhan, China, in 2002, and the M.S. degree in control theory and control engineering from Huazhong University of Science and Technology, Wuhan, China, in 2009, and the Ph.D. degree in traffic information engineering and control from Wuhan University of Technology, Wuhan, China, in 2019. He is currently a Lecturer with the School of Computer Science, Hubei University of Technology, Wuhan, China. His primary research interests include cryptography technology for cyber security, privacy preservation in artificial intelligence and cloud computing.

Zhenhua Han is currently pursuing the M.S. degree with the School of Computer Science, Hubei University of Technology, Wuhan, China. His current research interests include wireless network security and privacy preservation.



Weizhi Meng (Senior Member, IEEE) received the Ph.D. degree in computer science from City University of Hong Kong, Hong Kong, SAR, in 2013. He is currently a Full Professor with the School of Computing and Communications, Lancaster University, Lancaster, U.K.

He won the Outstanding Academic Performance Award during his doctoral study, and is a recipient of the Hong Kong Institution of Engineers Outstanding Paper Award for Young Engineers/Researchers in both 2014 and 2017. He received the IEEE ComSoc Best Young Researcher Award for Europe, Middle East, and Africa Region in 2020, and the IEEE MGA Young Professionals Achievement Award in 2020. His primary research interests are cyber security and intelligent technology in security, including intrusion detection, smartphone security, biometric authentication, HCI security, cloud security, trust management, blockchain in security, cyberphysical system security, and IoT security.