Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.069236

ARTICLE

A Secure and Efficient Distributed Authentication Scheme for IoV with Reputation-Driven Consensus and SM9

Hui Wei[1,2], **Zhanfei Ma**[1,3,*], **Jing Jiang**[1], **Bisheng Wang**[1] **and Zhong Di**[1]

[1]School of Digital and Intelligence Industry, Inner Mongolia University of Science and Technology, Baotou, 014010, China
[2]School of Information Engineering, Ordos Vocational College, Ordos, 017010, China
[3]School of Information Science and Technology, Baotou Teachers College, Baotou, 014010, China

*Corresponding Author: Zhanfei Ma. Email: mzfmail2025@163.com

Received: 18 June 2025; Accepted: 07 August 2025; Published: 10 November 2025

ABSTRACT: The Internet of Vehicles (IoV) operates in highly dynamic and open network environments and faces serious challenges in secure and real-time authentication and consensus mechanisms. Existing methods often suffer from complex certificate management, inefficient consensus protocols, and poor resilience in high-frequency communication, resulting in high latency, poor scalability, and unstable network performance. To address these issues, this paper proposes a secure and efficient distributed authentication scheme for IoV with reputation-driven consensus and SM9. First, this paper proposes a decentralized authentication architecture that utilizes the certificate-free feature of SM9, enabling lightweight authentication and key negotiation, thereby reducing the complexity of key management. To ensure the traceability and global consistency of authentication data, this scheme also integrates blockchain technology, applying its inherent invariance. Then, this paper introduces a reputation-driven dynamic node grouping mechanism that transparently evaluates and groups' node behavior using smart contracts to enhance network stability. Furthermore, a new RBSFT (Reputation-Based SM9 Friendly-Tolerant) consensus mechanism is proposed for the first time to enhance consensus efficiency by optimizing the PBFT algorithm. RBSFT aims to write authentication information into the blockchain ledger to achieve multi-level optimization of trust management and decision-making efficiency, thereby significantly improving the responsiveness and robustness in high-frequency IoV scenarios. Experimental results show that it excels in authentication, communication efficiency, and computational cost control, making it a feasible solution for achieving IoV security and real-time performance.

KEYWORDS: Internet of vehicles; consensus mechanism; blockchain; SM9

1 Introduction

With the rapid development of intelligent transportation technology and the Internet of Vehicles (IoV), high-frequency data exchange between vehicles and between vehicles and infrastructure is becoming a key method to improve real-world traffic safety and efficiency [1]. However, the highly dynamic, open, and heterogeneous network environment of the IoV poses severe information security challenges [2], particularly in terms of identity authentication [3] and consensus mechanisms. Existing authentication approaches often depend on centralized management and complex certificate mechanisms, which are challenging to meet the requirements of low-latency, high-reliability, and lightweight computing in the IoV environment. Besides, although mainstream consensus mechanisms such as PBFT (Practical Byzantine Fault Tolerance) and Raft have their advantages, they still face problems, including high computational and communication overhead,

complex node management, and insufficient fault tolerance, particularly when dealing with high-frequency communication and large-scale node collaboration.

In recent years, researchers have attempted to combine cryptography and blockchain technology with the security mechanism of IoV [4–6]. Among them, Identity-Based Cryptography (IBC) has received widespread attention due to its advantages of simplifying key management and reducing certificate dependencies. In particular, the SM9 algorithm, as a certificate-free identity-based cryptography [7,8], has the characteristics of lightweight, security, and scalability, making it exceedingly suitable for key negotiation and authentication in resource-constrained and highly dynamic scenarios. Meanwhile, blockchain technology naturally supports building a trusted authentication and consensus environment, thanks to its decentralized, tamper-proof and traceable characteristics.

However, existing research still faces many challenges when combining blockchain with IoV. For example, some blockchain-based anonymous authentication schemes have implemented identity privacy protection, but they have not fully considered the use of key distribution and tamper-proof devices [9], and the delay in ticket processing becomes a significant issue when facing a large number of vehicles. Consequently, specific consensus mechanisms in the IoV environment suffer from high communication complexity and poor scalability, which makes it challenging to meet the demands of real-time and high efficiency.

Regarding the above questions, this paper proposes an optimized distributed authentication scheme that integrates the SM9 algorithm and improved blockchain technology. It has built a multi-level consensus mechanism that is more scalable, stable, and secure.

On balance, the main contributions of this paper are as follows:

- We propose a secure and efficient distributed authentication scheme for IoV with reputation-driven consensus and SM9 to achieve secure and efficient identity verification and key management in IoV environments.
- To achieve transparent evaluation and trusted grouping of node behaviors, and enhance network trust management and system resilience, we introduce a reputation-driven dynamic node grouping mechanism that combines blockchain smart contracts.
- We also integrate the advantages of PBFT and Raft mechanisms, propose a new RBSFT consensus algorithm, and introduce SM9 aggregate signature and reputation scoring mechanisms. This design significantly enhances the efficiency and robustness in high-frequency communication scenarios while ensuring the security of the consensus process.
- The performance advantages of this scheme in typical IoV scenarios have been verified through many experiments, which include multiple dimensions such as authentication delay, communication overhead, and consensus efficiency. A feasible path is offered for building the next-generation vehicle networking security system.

The remaining content is as follows: Section 2 summaries the relevant works and analyses their shortcomings; Section 3 introduces the preliminary work and system model; Then, Section 4 states the proposed plan in detail; And Section 5 conducts security analysis; Section 6 describes performance comparison and simulation experiments; Finally, Section 7 provides a summary of the entire text and outlines future directions for work.

2 Related Works

The latest developments in the IoV have driven extensive research on secure authentication and consensus mechanisms to address their dynamic, open, and resource-constrained environment issues. Blockchain technology has been widely explored in IoV authentication and data management due to its decentralized

and tamper-proof characteristics. For instance, Zheng et al. [10] proposed a VANET authentication scheme based on a traceable blockchain, which enables anonymous Vehicle-RSU communication, but its dependence on frequent Trusted Authority (TA) interactions increases latency and limits scalability. Similarly, Noh et al. [11] introduced a blockchain-based message authentication scheme that achieved decentralization, but incurred high computational overhead due to its complex cryptographic operations. Yao et al. [12] developed a lightweight blockchain-assisted authentication (BLA) mechanism using the PBFT protocol, which reduces duplicate authentication. However, its dependence on traditional Public-Key Infrastructure (PKI) introduced complexity in the management of certificates. Although Shen et al. [13] proposed a mutual authentication scheme for edge computing based on ECC, its certificate overhead places a burden on resource-constrained OBUs. Besides, Bojjagani et al. [14] integrated blockchain and edge computing together for security key management, yet their anonymity is weak, and they are vulnerable to traceability attacks.

Regarding consensus mechanisms, existing research (e.g., PoW and PoS) is computationally intensive, while Wang et al. [15] analyzed the PBFT, which provides reliability, but there is a secondary communication complexity ($O(N^2)$) in large-scale IoV networks. Although the R-PBFT proposed by Kumar et al. [16] is a reputation-based variant of PBFT, it ignores node mobility and reduces efficiency in dynamic scenarios. Then, Xu et al. [17] introduced SG-PBFT, which reduces communication overhead but lacks robust defenses against malicious nodes. Liu et al. [18] combined the blockchain with edge computing to implement low-latency access control, but their simplified PBFT sacrifices Byzantine fault tolerance. Although El-Zawawy et al. [19] used ECC and blockchain for drone-assisted IoV, their scalability is limited by the computational intensity of ECC. Additional studies, such as Karim et al. [20], proposed blockchain-based data exchange solutions, but their high communication costs consistently hinder real-time performance. Chen et al. [21] provided a summary of the blockchain security technology in IoV and highlighted the scalability challenges in high-frequency scenarios. Furthermore, Sharma et al. [22] developed an energy-efficient blockchain transaction model, but it lacks consensus on optimizing the dynamic structure of IoV. Mu et al. [23] explored the secure signature of SM9, which emphasizes its advantages of being certificate-free, but they did not address the consensus optimization problem. Meanwhile, Xu et al. [24] proposed GPU-accelerated SM9 processing, which improves efficiency, but it was not tailored for the distributed consensus needs of IoV. Dong et al. [25] designed an improved PBFT consensus mechanism (IPBFT) that uses group signatures for voting-based sorting and clustering, thereby enhancing privacy, but lacking efficient smart contract triggering and dynamic node management. A dual-blockchain layered PBFT (DBPBFT) for IoT, introduced by Wu et al. [26], is suitable for static scenarios but is limited by complex certificate management and a lack of a fast re-authentication mechanism. The MGRS-PBFT proposed by Zheng et al. [27] combines credit scoring of node behavior, but lacks robust ledger auditing and fast re-authentication mechanisms. Although Zheng et al. [28] developed TRBFT, an efficient blockchain consensus for edge computing IoT, it does not completely solve the high mobility or integrated aggregation signature problem of the IoV. Deshmukh et al. [29] presented a scalable Byzantine fault-tolerant consensus for vehicular networks, but its fixed leader election limits its adaptability in dynamic IoV environments. In addition, a shard-based consensus protocol introduced by Lu et al. [30] advances scalability but lacks reputation-driven trust management and SM9 integration.

Our proposed scheme addresses the shortcomings of existing IoT systems in terms of security, efficiency, and scalability through multiple innovations. Firstly, unlike [10,12,13,26], introducing the identity-based uncertified cryptographic system SM9 simplifies key management and reduces the computational burden on VU and RSU. Then, the RBSFT consensus mechanism integrates the fault tolerance of PBFT, the log synchronization of Raft, and the SM9 aggregate signature, reducing communication complexity from $O(N^2)$ to $O(N)$, which is superior to PBFT, SG-PBFT [17], IPBFT [25], and others. Unlike R-PBFT [16]

and IPBFT [25], we employ a reputation-driven dynamic grouping mechanism to eliminate node location dependence, which is more suitable for the dynamic structure of the IoV and superior to the fixed grouping or voting methods in [29,30]. Besides, building a reputation evaluation mechanism through smart contracts, prioritizing the participation of high-reputation nodes in consensus, and enhancing anti-attack capabilities, is superior to [17,18,25,27]. SM9 aggregate signature optimization in the Prepare and Commit stages reduces verification overhead and solves the issues of high communication costs in [20] and insufficient privacy protection in [11,14]. Finally, the lightweight re-authentication mechanism we designed combines blockchain certificates with SM9 signatures to avoid duplicate calculations and reduce authentication latency, which is superior to the authentication redundancy issues in [12,26,27].

To clearly demonstrate these advantages, we create Table 1 and compare RBSFT with PBFT, Yao et al. [12], R-PBFT [16], and SG-PBFT [17] using key IoV metrics. All schemes maintain Byzantine Fault Tolerance to ensure robustness against malicious nodes. However, our scheme supports the reuse of authentication results, which differs from the partial reuse in [12,17] or none in [16], thereby reducing redundant computations. Due to its reputation-driven grouping, RBSFT offers high support for mobile nodes, surpassing the limited support in [16,17] and the moderate support in [12]. The leader election strategy in RBSFT combines reputation-based selection with local master selection, providing more dynamic and trustworthy leadership than the fixed strategies in [12] or group-based in [17]. Compared with the single signatures in [12,16] and BLS aggregation in [17], the SM9 aggregate signature reduces verification overhead. The latency of IoV (0.31–1.01 s) is lower than that of competitors (0.35–1.10 s), reflecting the efficiency of RBSFT. Finally, due to the lightweight operations and optimized consensus of SM9, the computational cost (6.56 ms) is the lowest, outperforming [12] (9.255 ms), [16] (7.575 ms), and [17] (6.735 ms).

Table 1: Symbol and description

Metric	PBFT	Yao et al. [12]	R-PBFT [16]	SG-PBFT [17]	Ours
Byzantine fault tolerance	Yes	Yes	Yes	Yes	Yes
Authentication result reuse	No	Partial	No	Partial	Yes
Support for mobile nodes	Limited	Moderate	Limited	Limited	High
Leader election strategy	Fixed	Fixed	Reputation-based	Group-based	Reputation & Local master selection
Signature mechanism	Individual	Individual	Individual	BLS Aggregate	SM9 Aggregate
Latency in IoV	0.35–1.05 s	0.38–1.08 s	0.40–1.10 s	0.35–1.05 s	0.31–1.01 s
Computational cost (ms)	7.5	9.255	7.575	6.735	6.56

3 Preliminaries and System Model

This section first introduces the theoretical foundations of blockchain technology, the SM9 signature mechanism, and smart contracts. Subsequently, the system model proposed in this paper is systematically elaborated, laying a theoretical foundation and offering architectural support for subsequent scheme design.

3.1 Blockchain

Blockchain is a decentralized distributed ledger technology that stores transaction records by a chained data structure to ensure data immutability and transparent traceability [31]. All nodes in the system maintain the same ledger copy to record transactions and status updates, thereby eliminating reliance on centralized institutions and providing a solid foundation for establishing decentralized trust mechanisms and secure collaboration. In the IoV scenario, blockchain can not only store authentication results and node reputation scores, but also realize automatic policy execution through smart contracts, ensuring the efficiency and credibility of authentication and consensus processes.

3.1.1 Smart Contract

A smart contract [32] is an automated program deployed on the blockchain that autonomously executes through pre-defined rules and logic. Compared to traditional contracts, smart contracts have the following significant advantages: Firstly, their blockchain-based non-tampering characteristics ensure the transparency and immutability of the contract terms; Then, the automatic execution mechanism eliminates dependence on intermediaries, which significantly improves execution efficiency, and reduces trust costs; Finally, their flexible programmability can support the implementation of complex business logic. In this paper, a smart contract is deployed on the Road Side Units (RSUs), mainly responsible for triggering authentication logic, verifying timestamps, and querying authentication credentials in the blockchain ledger, thereby providing efficient support for distributed authentication and consensus processes.

3.1.2 PBFT Consensus Mechanism

The Practical Byzantine Fault Tolerant (PBFT) algorithm was proposed by Castro et al. in 1999 [33], which solves the Byzantine problem in distributed systems by ensuring consistency in the presence of malicious nodes. The algorithm consists of four phases: Pre-Prepare, Prepare, Commit and Reply. In a system with a total number of N nodes, PBFT can tolerate a maximum of $f = (N - 1)/3$ faulty or malicious nodes. This paper applies the improved PBFT algorithm to the IoV scenario because it can effectively defend against tampering attacks against the centralized database while satisfying the requirements of high security and low latency. However, the communication complexity of traditional PBFT is $O(N^2)$, which can lead to significant performance bottlenecks when the number of nodes is large, limiting its applicability in highly dynamic IoV environments.

3.1.3 Raft Consensus Mechanism

Raft is a distributed consensus algorithm proposed by Ongaro et al. [34]. It is mainly used to ensure consistency in log replication, and its design focuses on simplicity and efficiency. The Raft algorithm consists of two core mechanisms: Leader Election and Log Replication.

Leader Election: Raft divides nodes into Leader, Follower, and Candidate roles. There is usually only one leader in the system, responsible for coordinating log replication and client requests. If the leader fails, the followers will become candidates after a timeout and become the new leader by initiating an election and obtaining the votes of the majority of nodes. This mechanism reduces election conflicts through a randomized timeout strategy, ensures quick recovery, and maintains a single leader in the event of failure, thereby avoiding brain-disruption problems.

Log Replication: The leader receives client requests, packages the operations in the form of log entries to generate blocks, and sends the logs to all follower nodes through the heartbeat mechanism. After confirming the log, the followers append the block to the local ledger.

The communication complexity of the Raft algorithm is $O(N)$. The consensus is efficient and easy to realize, but it lacks Byzantine fault-tolerance to deal with malicious nodes or Byzantine failures. Therefore, the RBSFT consensus mechanism proposed in this paper combines the efficient log replication characteristics of Raft and the Byzantine fault-tolerance capability of PBFT, making it quite suitable for the IoV environments.

3.2 SM9 Signature

SM9 was proposed by the Chinese government in 2016 [35] as an Identity-Based Cryptography (IBC) standard, and was later included in the ISO 2021 international standard. This standard is widely used in secure authentication and key negotiation in certificate-free scenarios. SM9 is built on bilinear pairing and elliptic curve cryptography, and its significant advantage lies in eliminating certificate management in traditional public key infrastructure (PKI), allowing users to directly use identity information as the public key, thereby significantly simplifying the process of key distribution and storage. The SM9 signature mechanism has the characteristics of high generation and verification efficiency, and is quite suitable for resource-constrained IoV scenarios. Additionally, SM9 supports signature aggregation functionality, which can compress the signatures of multiple nodes into a fixed-length unified signature, thereby effectively reducing network traffic and verification burden. Therefore, this article introduces this feature for the first time into the identity authentication system of the Internet of Vehicles, filling a gap in existing research in this field. In the RBSFT consensus mechanism, the SM9 aggregate signature is used to optimize the Prepare and Commit stages. The master node can aggregate the signatures by no less than $2f + 1$ nodes into a single signature, thereby reducing the communication complexity from $O(N^2)$ to $O(N)$. Thanks to the certificate-free nature of SM9 and its high efficiency in signature verification, this solution not only helps to build a distributed system that supports high concurrency and low latency, but also further enhances the overall system's performance in terms of security and anti-counterfeiting capabilities.

3.3 System Model

As shown in Fig. 1, this paper proposes a system model comprising two modules: the blockchain network and the IoT, to create a secure and efficient communication architecture for the IoV. The blockchain module adopts a federation chain consisting of RSUs and ESs, with TAs acting as fully trusted organizations to ensure system security and data reliability. RSUs and ESs are exposed to threats as semi-trusted nodes deployed in open environments, and are thus equipped with SM9 signatures and smart contracts to enhance security. The Telematics module consists of VUs that form a dynamic network through wireless communication for real-time interaction between vehicles and infrastructure. To enhance authentication efficiency, RSU deploys smart contracts that supports SM9 signatures, enabling automatic authentication and access control to provide trusted smart services. The architecture integrates the decentralized trust mechanism of blockchain with the real-time communication advantages of IoT to build a secure, low-latency and scalable solution. A detailed description of each component function will follow.

3.3.1 Trusted Authority (TA)

As a fully trusted entity, the Trusted Authority (TA) is responsible for the initialization and registration of vehicles, and serves as the Key Generation Center (KGC). TA has powerful computing and storage capabilities to ensure the efficiency and security of data processing.

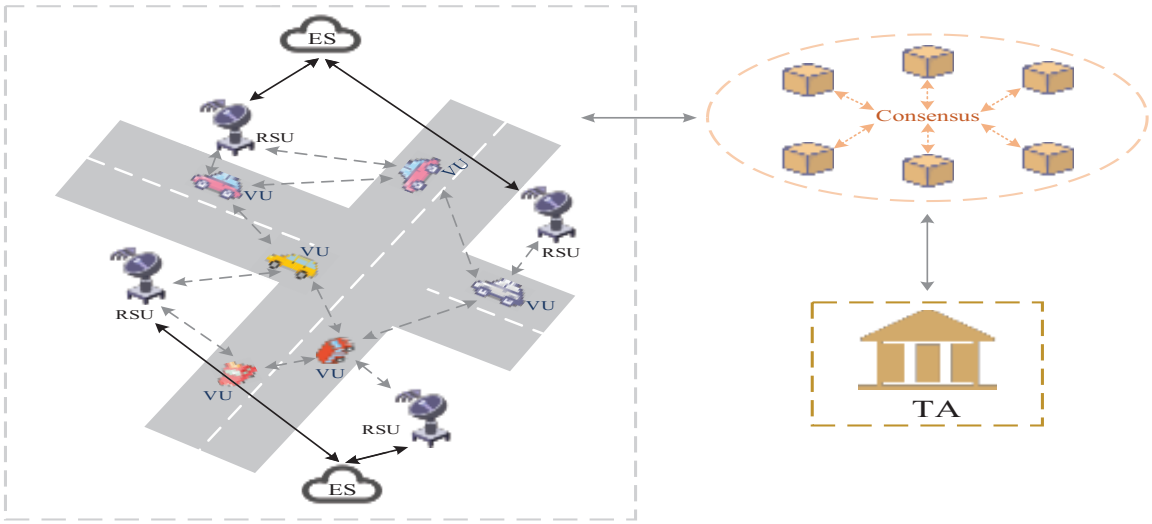

Figure 1: Proposed system model

3.3.2 Edge Server (ES)

ES is equipped with powerful computing and storage resources, deployed near RSUs, and is responsible for supervising operations in its respective regions. The main responsibilities include: unified management of RSUs within the area, collaborative identity verification, participation in blockchain consensus, and maintenance of a shared ledger that records vehicle access and verification results. As a consensus node in the consortium chain, ES has block generation and on-chain permissions to ensure data consistency and security. ES handles consensus computing with high resource consumption, while RSU focuses on local identity authentication, with a clear division of labor to optimize system efficiency, security, and scalability.

3.3.3 Road Side Unit (RSU)

Road Side Units (RSUs) are deployed along the road, managed by ES, and wirelessly interact with VUs using the Dedicated Short-Range Communication (DSRC) protocol. RSU is equipped with a data exchange module and a smart contract module, supporting SM9 signature verification to perform efficient identity authentication and data processing. Without participating in blockchain consensus or block generation, RSU focuses on low-latency local identity verification through smart contracts and forwards the results to ES for recording in the blockchain ledger. This division of labor ensures that RSUs focus on local tasks, while ES handles computation-intensive operations.

3.3.4 Vehicle Unit (VU)

Vehicle Unit (VU) is mounted on the vehicle and realizes efficient communication between the vehicle and external entities based on wireless communication technology through two-way authentication with the RSU. The VU supports the provision of road accident information, traffic condition updates, driving safety tips, as well as communication, computation, and data storage functions, thus providing comprehensive assistive services to drivers and passengers.

4 Proposed Scheme

In this section, the authentication mechanism and overall process of the proposed scheme are systematically described, covering five key phases: Initialization, Registration, Authentication, Consensus and Re-Authentication. As shown in Fig. 2, it mainly illustrates the last three stages, the initialization and registration phases are executed only once. The related symbols and their meanings are shown in Table 2.

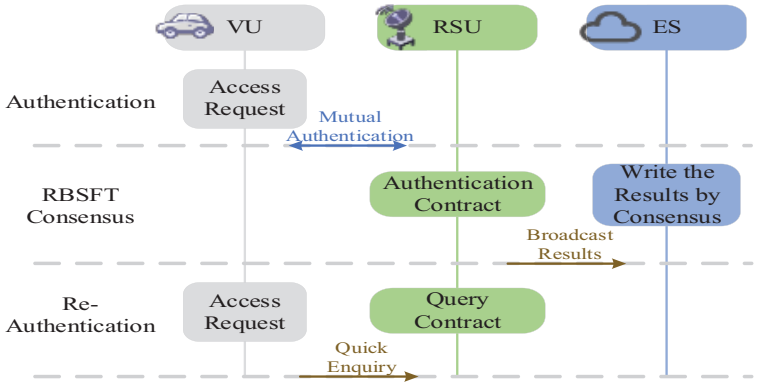

Figure 2: Flowchart of the proposed scheme

Table 2: Symbol and description

Symbol	Description
ID_X	The identity of X
d_X	The private key of X
hid	Key type identifier for SM9 key generation
σ_X	SM9 signature generated by entity (X)
r	Random number used in SM9 signature and KEM operations
$\|\|$	Concatenation operation
ω	Randomized median to hide (r) or (S) in SM9 signature
h	The hash value is used to verify the signature's correspondence to the original message
l	Combination of random number (r) and hash value (h) to hide (r)
S	Signature point carrying the master node's private key information
C_{VU}	SM9 KEM ciphertext generated by VU for the authentication message
C_{RSU}	SM9 KEM ciphertext generated by RSU for authentication response
C_1	An elliptic curve point
C_2	The encrypted message
K	Session key generated via SM9 key exchange between VU and RSU
$Cert_{auth}$	Authentication credential of VU is stored in the blockchain
ΔT	Maximum legitimate time interval for timestamp verification
e	Bilinear pairing function ($G_1 \times G_2 \rightarrow G_T$) in SM9
P_1, P_2	Generating elements of additive cyclic groups (G_1, G_2) in SM9
Q	Intermediate value based on the public key and a random number

4.1 Initialization Phase

In this phase, the TA is responsible for completing the initialization operation of the system. As the Key Generation Center (KGC), the TA selects a 256-bit Barreto-Naehrig (BN) elliptic curve E of order a large prime number N according to the SM9 standard. Meanwhile, TA defines the bilinear pairing function e: $G_1 \times G_2 \to G_T$, where G_1, G_2 are additive cyclic groups on elliptic curves and G_T is a multiplicative cyclic group with an average order N. Then, the generating elements $P_1 \in G_1$, $P_2 \in G_2$ are selected. TA randomly selects the master private key $s \in \mathbb{Z}_N^*$ and calculates the corresponding master public key $P_{pub} = s \cdot P_2$. Finally, two hash functions are defined, $H_1: \{0,1\}^* \to \mathbb{Z}_N^*$ for identity mapping and $H_2: \{0,1\}^* \to \mathbb{Z}_N^*$ for signature and encryption.

The public keys of RSUs and ESs directly use identifiers (e.g., IP address, device ID) and do not need to be generated independently. TA generates the respective SM9-based private key $h_{RSU} = H_1(ID_{RSU}||hid) mod\ N$, then $d_{RSU} = s \cdot {h_{RSU}}^{-1} \cdot P_1$, for RSUs and ESs, respectively, and similarly generates d_{ES}. At this point, TA generates the private keys by scalar multiplication to ensure the certificate-less feature and reduce the storage overhead. We deploy smart contracts that support SM9 signature verification on RSUs to initialize the SM9 algorithm and the reputation score table.

4.2 Registration Phase

In this phase, the VU completes registration with the TA, generates the SM9 private key and stores the signature record on the blockchain. The specific process is as follows.

1. First, the VU hashes its MAC address to obtain a unique identity $ID_{VU} = H_1(MAC_{VU})$, which will be used as the public key of SM9, eliminating the need for traditional public key management. Meanwhile, the TA generates the SM9 private key $h_{VU} = H_1(ID_{VU}||hid) modN$, then $d_{VU} = s \cdot {h_{VU}}^{-1} \cdot P_1$, at which time the private key is generated by scalar multiplication and sent to the VU through a secure channel.
2. Then, VU generates a timestamp T_{VU} and uses private key d_{VU} on $m = \{ID_{VU}, T_{VU}\}$ to generate SM9 signature. It first chooses random number $r \in \mathbb{Z}_N^*$, computes $\omega = e(P_1, P_{pub})^r$, $h = H_2(m||\omega) modN$, $S = h \cdot d_{VU}$, and then outputs signature $\sigma_{VU} = (h, S)$. Then, it sends a request message $\{ID_{VU}, T_{VU}, \sigma_{VU}\}$ to TA.
3. When TA receives the message, it first verifies $|T_{now} - T_{VU}| \le \Delta T$ and the signature. Then, it computes $\omega' = e(S, P_2) \cdot e(h \cdot (h_{VU} \cdot P_1), P_{pub}$, $h' = H_2(m||\omega') modN$, and checks $h' = h$. After verifying both the timestamp and signature, proceed with the subsequent steps.
4. After the TA check is passed, the initial identity credential $Cert_{init} = \{ID_{VU}, H_2(d_{VU}||ID_{VU})\}$ is generated, for which SM9 is used to sign $\sigma_{TA} = Sign_{d_{TA}}(Cert_{init})$. At the same time, KEM is performed using SM9: A random number $t \in \mathbb{Z}_N^*$ is selected, $C = t \cdot P_1$, $Q = r \cdot P_{pub} + H_1(ID_{VU}||hid) modN \cdot P_2, K = e(Q, P_1)^r$, derive encryption key via KDF (KDF reference can be found in [35]), and $C_{store} = (C||\{Cert_{init}, \sigma_{TA}\} \oplus K)$ are computed. And finally, the encrypted C_{store} is stored in the blockchain to ensure that only authorized RSUs can decrypt it, completing the registration process. The encrypted $Cert_{init}$ is stored in the blockchain to ensure that only authorized RSUs can decrypt it, completing the registration process. Note that only the initial credential $Cert_{init}$ is stored during registration, as full authentication parameters (e.g., $Cert_{auth}, \sigma_{auth}$) are session-specific and generated during the mutual authentication phase (Section 4.3) between VU and RSU. This design ensures that the registration phase establishes the VU's identity securely, while the authentication phase dynamically generates parameters to establish mutual trust.

4.3 Authentication Phase

After the registration is completed, VU and nearby RSUs authenticate each other, and RSUs broadcast the authentication results to ES. ES then uses an improved consensus mechanism to write the results into the blockchain ledger. When VU initiates access again, the specific process is shown in Fig. 3.

Step 1: VU generates a random number $r_{VU} \in \mathbb{Z}_N^*$, a timestamp T_{VU}, and computes the message $m_{VU} = ID_{VU}||r_{VU}||T_{VU}$. Then, VU generates an SM9 signature σ_{VU} on m_{VU}: select $\omega \in \mathbb{Z}_N^*$, compute $l = H_2(m_{VU}||\omega) mod N$, compute $S = l \cdot P_1 + d_{VU}$, output $\sigma_{VU} = (h, S)$, where $h = H_2(m_{VU}||\omega) mod N$. Afterwards, VU performs SM9 KEM to encrypt $m_{VU}||\sigma_{VU}$: select $r \in \mathbb{Z}_N^*$, compute $C_1 = r \cdot P_1$, $Q = r \cdot P_{pub} + H_1(ID_{RSU}||hid) mod N \cdot P_2$, compute $K = e(Q, P_1)^r$, derive encryption key via KDF. VU encrypts $m_{VU}||\sigma_{VU}$ to obtain ciphertext $C_{VU} = (C_1, C_2)$, $C_2 = Enc_K(m_{VU}||\sigma_{VU})$. Enc_K represents the length of the required key. Finally, VU sends $M_1 = (ID_{VU}, C_{VU})$ to RSU.

Step 2: RSU receives M_1, extracts ID_{VU}, and decrypts C_{VU} using its private key d_{RSU}: compute $Q = d_{RSU} \cdot C_1$, derive decryption key $K = e(Q, P_1)^r$. VU decrypts C_2 to recover $m_{VU}||\sigma_{VU}$. Then, RSU triggers the smart contract to verify T_{VU}: check if $|T_{now} - T_{VU}| \leq \Delta T$, where ΔT is the maximum legitimate time interval (100–200 ms, validated via SUMO simulations). Afterwards, RSU verifies $\sigma_{VU} = (h, S)$: compute $\omega' = e(S, P_2) \cdot e(h \cdot (h_{VU} \cdot P_1), P_{pub})$, $h'' = H_2(m_{VU}||\omega') mod N$, and checks $h'' = h$. If both checks pass, RSU queries the blockchain for VU's credential C_{store}: decrypt C_{store} using d_{RSU}, verify TA's signature σ_{TA}, and confirm VU's registration legitimacy.

Step 3: RSU generates a random number r_{RSU}, a timestamp T_{RSU}, and computes $m_{RSU} = ID_{RSU}||r_{RSU}||T_{RSU}$. Then, RSU generates an SM9 signature σ_{RSU} on m_{RSU}: select $\omega \in \mathbb{Z}_N^*$, $l = H_2(m_{RSU}||\omega) mod N$, compute $S = l \cdot P_1 + d_{RSU}$, output $\sigma_{RSU} = (h, S)$, where $h = H_2(m_{RSU}||\omega) mod N$. Afterwards, RSU performs SM9 KEM to encrypt $m_{RSU}||\sigma_{RSU}$: select r $\in \mathbb{Z}_N^*$, compute $C_1 = r \cdot P_1$, $Q = r \cdot P_{pub} + H_1(ID_{VU}, hid) mod N \cdot P_2$, compute $K = e(Q, P_1)^r$, derive encryption key via KDF. RSU encrypts $m_{RSU}||\sigma_{RSU}$ to obtain ciphertext $C_{RSU} = (C_1, C_2)$, $C_2 = Enc_K(m_{RSU}||\sigma_{RSU})$. Finally, RSU sends $M_2 = (ID_{RSU}, C_{RSU})$ to VU.

Step 4: VU receives M_2, decrypts C_{RSU} using d_{VU}: compute $Q = d_{VU} \cdot C_1$, derive decryption key $K = e(Q, P_1)^r$. VU decrypts C_2 to recover $m_{RSU}||\sigma_{RSU}$. Then, VU verifies T_{RSU}: check if $|T_{now} - T_{RSU}| \leq \Delta T$. Afterwards, VU verifies $\sigma_{RSU} = (h, S)$: compute $\omega' = e(S, P_2) \cdot e(h \cdot (h \cdot P_1), P_{pub})$, $h''' = H_2(m_{RSU}||\omega') mod N$, and checks $h''' = h$. If both checks pass, VU confirms RSU's legitimacy.

Step 5: Upon successful mutual authentication, VU and RSU derive a session key K' via SM9 key exchange: both parties compute $K' = H_2(ID_{VU}||ID_{RSU}||r_{VU}||r_{RSU})$. RSU generates an authentication credential $Cert_{auth} = (ID_{VU}, T_{RSU}, K')$, compute $\sigma_{auth} = (h, S)$ on $Cert_{auth}$ as in Step 3. Then, the RSU broadcasts the result $\{Cert_{auth}, \sigma_{auth}\}$ to the ES for consensus via the RBSFT algorithm. Finally, ES writes $Cert_{auth}$ to the blockchain ledger for re-authentication.

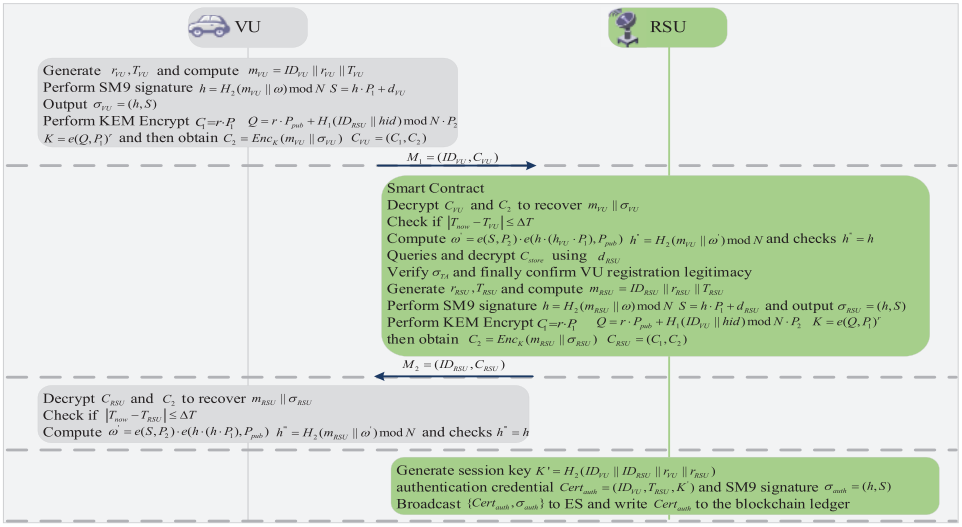

Figure 3: Mutual authentication mechanism diagram

4.4 RBSFT Consensus Phase

After receiving the broadcast authentication result from RSU, ES first verifies its legitimacy and securely writes the result into the blockchain ledger by consensus algorithm. To address the disadvantages of PBFT and Raft mechanisms in terms of performance and security, this paper proposes an improved consensus mechanism RBSFT, which combines the advantages of both and the SM9 algorithm. This mechanism inherits the Byzantine fault-tolerant features of PBFT, while drawing on Raft's efficient log synchronization design, and introducing a reputation-driven node grouping strategy, a dynamic leader election mechanism, and an improved SM9 signature aggregation algorithm. It improves the scalability and response speed of the system structurally, constructs a more elastic and adaptive consensus system, and significantly advances the security and privacy protection capabilities of the consensus process.

RBSFT supports aggregating multiple node signatures into a single verification object, which effectively reduces computational and communication overhead and improves system performance in high-frequency interaction scenarios. Meanwhile, the SM9 algorithm supports the encryption processing of log data, further improving the confidentiality and tamper resistance of on-chain data, and building a dual defense line of consensus layer and privacy protection layer.

The system divides nodes into High Reputation Group (HRG) and Regular Group (RG) based on their historical behavior and dynamic reputation rating. HRG nodes have higher priority in consensus due to their stable participation and complete records. Before each round of consensus, the system automatically selects the node with the highest reputation in HRG as the leader, and the rest as followers to jointly complete the consensus task. The system regularly updates the reputation score, removes nodes below the threshold, and adopts an adaptive management strategy based on reputation to ensure the dynamic effectiveness of the mechanism.

Besides, SM9 aggregate signatures play a crucial role in node elections and role adjustments. By aggregating signatures of election messages and identity authentication information, the authenticity and non-repudiation of node identities are ensured, effectively preventing attacks such as man-in-the-middle forgery and replay, and consolidating the trust foundation of the consensus process. The specific steps are as follows.

1. Pre-prepare phase: After receiving the client request, the master node constructs a message $m = \{d, h, v\}$, where h represents the height of the block, d represents the digest, and v represents the view number. Then, it uses SM9 signature $\sigma_{Leader} = (h_{Leader}, S_{Leader})$ and broadcasts $\{m, \sigma_{Leader}\}$ to $n-1$ replica nodes.
2. Prepare phase: The replica node verifies the signature: calculates $\omega' = e(S, P_2) \cdot e(h \cdot (h_{Leader} \cdot P_1), P_{pub})$, $h' = H_2(m||\omega')modN$, checks $h' = h_{Leader}$, and generates the SM9 signature $\sigma_i = (h_i, S_i)$ and sends it to the master node after the verification passes. If the verification fails, the reputation score of the master node is reduced, and the current consensus process is terminated.
3. Commit phase: The master node collects at least $2f + 1$ SM9 signatures and aggregates them into a single SM9 signature $\sigma_{agg} = \sum_{i=1}^{2f+1} S_i$. It broadcasts a message $\{h, v, d, \sigma_{agg}, node_{list}\}$ to the replica node, where $node_{list}$ is a list of node IDs participating in the aggregated signature. After receiving the message, the replica node verifies the aggregated signature, and if it passes, appends it to the blockchain.
4. Consensus phase within the group: The master node signs the Log using SM9 and then broadcasts it to the HRG through a secure channel. All replica nodes in the HRG receive the message, verify the signature, and after they all pass, the replica node appends the Log to the local log storage and sends an acknowledgement message to the master node to ensure that the master node is aware of the Log replication status. If there are more than $2k/3$ (k is the number of HRG nodes, which ensures Byzantine fault tolerance) valid confirmations, the Log is considered to have been successfully replicated. The master node submits the log to the blockchain, triggering the smart contract to update the global state.
5. Reply phase: The master node sends a confirmation message to the client, while the replica node updates its reputation score based on the consensus results of this round. If the node's judgment is consistent with the final consensus result, its reputation score will increase; On the contrary, it will decrease. The system dynamically adjusts the membership composition of HRG and the RG based on the updated reputation scores.

4.5 Re-Authentication Phase

Given the high mobility of vehicles, when VU initiates another authentication request, the system simplifies the authentication process to a query operation on the blockchain ledger to avoid duplicate authentication. Thus, it reduces communication overhead and computational costs and realizes lightweight re-authentication. The authentication credential $Cert_{auth}$, generated during the initial authentication phase (Section 4.3), is stored in the blockchain and typically not updated unless the authentication fails (e.g., if $Cert_{auth}$ is absent or invalid, a new mutual authentication will be triggered to generate a fresh $Cert_{auth}$). The specific flow of the re-authentication stage is described below.

1. The VU generates the current timestamp T_{VU}, uses SM9 signature $\sigma_{VU} = (h, S)$ for the message $m_{VU} = \{ID_{VU}, T_{VU}\}$, encrypts it using SM9 KEM, and sends it over a secure channel to a nearby RSU.
2. The RSU decrypts it using d_{RSU}, automatically triggers the smart contract authentication logic, verifies that the timestamp is within the valid range $|T_{now} - T_{VU}| \leq \Delta T$, and then queries $Cert_{auth}$ in the blockchain ledger record to decrypt and verify the SM9 signature: $\omega' = e(S, P_2) \cdot e(h \cdot (h_{VU} \cdot P_1), P_{pub}$, $h' = H_2(Cert_{auth}||\omega')\, modN$. Then, RSU checks $h' = h$. If the timestamp and signature are valid, the authentication is successful, and the reputation score of the VU is updated. If there is no $Cert_{auth}$, it queries the initial credential C_{store}, decrypts and verifies the signature σ_{TA} of the TA. And if it is valid, the system will re-trigger the mutual authentication. To ensure unlinkability, each re-authentication session uses a fresh random number r and timestamp T_{VU}, making σ_{VU} unique across sessions despite using the same d_{VU}. This randomization, combined with SM9's cryptographic properties, prevents an attacker from linking multiple re-authentication sessions to the same VU, as detailed in Section 5.4.9. The

use of $Cert_{auth}$ for re-authentication significantly reduces computational and communication overhead compared to full mutual authentication, enhancing efficiency in high-mobility IoV scenarios.

5 Security Analysis

This section first elaborates on the threat model of the proposed scheme and then comprehensively evaluates its security using various approaches, including security analysis based on the Real-Or-Random (ROR) model, formal verification using the AVISPA tool, and informal security discussions.

5.1 Threat Model

The threat model presented in this paper describes the potential security risks and the capabilities of attackers in the IoV network environment. We assume that TAs are usually operated by government departments or highly trusted entities that are fully credible and immune to attacks, ensuring that all data they generate or store remains secure. In contrast, ESs, RSUs, and VUs are considered semi-trusted entities because they operate in open and dynamic environments, making them vulnerable to multiple attacks. Based on the information security CIA triad model, they can be compromised in several ways. An attacker can compromise the confidentiality of the system and eavesdrop on authentication messages to extract sensitive information. Besides, an attacker can destroy the integrity of the system by forging or tampering with authentication messages, signatures, or blockchain ledger records (The attacks that adversaries can perform are detailed in the following ROR analysis).

5.2 Formal Security Analysis Using ROR Model

In authentication protocols for the IoV, we prove the security of the proposed scheme using the ROR model, a widely used tool for proving the security of authentication protocols, whose effectiveness is recognized by the academic community. By simulating various passive and active attacks, we verified the resistance of session keys to attacks and designed multiple game experiments in an instantiated network to distinguish between random numbers and session keys by testing queries.

For this purpose, we define an adversary A_v that is given multiple attack capabilities to model potential security threats. Specifically, A_v can perform the following query operations:

1. Message Interception Query: A_v can capture messages transmitted on the public channel and obtain the content of the communication.
2. Message Sending Query: A_v may send forged or altered messages to participating entities in the system to interfere with normal communications.
3. Identity Disguise Query: A_v can obtain some of the secret parameters and impersonate a legitimate vehicle (VU) or RSU to initiate an illegal authentication request.
4. Key Attack Query: A_v may attempt to crack or derive the session key in various ways to challenge the confidentiality of the key.
5. Security Test Query: A_v performs a test query by tossing a coin. If the result is 1, it indicates that the session key may be insecure; if the result is 0, it indicates that the session key can withstand attacks and is secure.

Definition 1: *SM9 Bilinear Pairing Problem (SBPP): Given a bilinear pair e in the SM9 algorithm,* $G_1 \times G_2 \rightarrow G_T$, *where* G_1, G_2 *are additive cyclic groups,* G_T *is a multiplicative cyclic group, the order is a large prime number N, and the generating element* $P_1 \in G_1, P_2 \in G_2$, *the master private key* $k \in \mathbb{Z}_N^*$, *and the master public key* $P_{pub} = k \cdot P_2$. *The SBPP problem requires either the computation of k in polynomial time or the derivation of the signature or key-related parameters based on* (P_1, P_2, P_{pub}, e).

Theorem 1: *Assume that $Adv_{A_v}^{ROR}(t)$ denotes the probability that the adversary A_v cracks the session key of the proposed scheme in polynomial time t. Define the hash query and the total number of sent queries as q_h, q_s respectively. The range space of the hash function is $|Hash|$. The cracking probability of the SM9 Bilinear Pairing Problem (SBPP) is ε_{SBPP}, and the reputation scoring parameter is ρ. Then the security of the proposed scheme satisfies the following inequality:*

$$Adv_{A_v}^{ROR}(t) \leq \frac{q_h^2}{|Hash|} + 2 \cdot (q_s \cdot \varepsilon_{SBPP}) + 2 \cdot \frac{q_s}{\rho} \tag{1}$$

Proof: We construct three games G_0, G_1, G_2, analyze the attacks and their win rates, respectively, and derive security bounds by Eq. □

Initial game G_0: In G_0, A_v has not obtained any message about the session key and can only guess the key randomly by test query. A_v chooses a random bit b (0 or 1), and according to the definition of the ROR model, Adv_{G_0} denotes the advantage of distinguishing between the real key and the random number, satisfying the following equation:

$$Adv_{G_0} = \left| Pr\left[b' = b\right] - \frac{1}{2} \right| \tag{2}$$

Since A_v lacks any prior information, the probability that its guess is correct is 1/2, i.e.:

$$Pr[b' = b] = \frac{1}{2}, Adv_{G_0} = 0 \tag{3}$$

Hash Query Attack G_1: In G_1, A_v performs a message interception query and a send query to capture the authentication message (e.g., $\{ID_{VU}, r_{VU}, T_{VU}, \sigma_{VU}, C_{VU}\}$) and tries to decipher the parameters (e.g., $h_1 = H_1(ID_{VU}||hid||P_{pub})$) via a hash query. However, the proposed scheme uses cryptographically secure one-way hash functions H_1, H_2 whose collision resistance is based on the birthday paradox [36]. The probability that A_v triggers a collision via q_h hash query is limited:

$$Pr\left[Coll_h\right] \leq \frac{q_h^2}{2 \cdot |Hash|} \tag{4}$$

Due to the strong collision resistance of the hash function, A_v is unable to derive the secret parameter. Thus, the increment of G_1's win rate is determined only by the hash collision probability:

$$|Pr\left[G_1\right] - Pr\left[G_0\right]| \leq \frac{q_h^2}{2 \cdot |Hash|} \tag{5}$$

SBPP and Reputation Scoring Attack G_2: In G_2, A_v tries to derive the session key in conjunction with the SBPP problem, using the captured authentication message and the SM9 aggregated signature σ_{agg} from the RBSFT consensus. Specifically, A_v needs to solve the SBPP problem (e.g., derive the master private key k or the signature parameter S) in polynomial time with probability ε_{SBPP}. In addition, A_v may try to interfere with the RBSFT consensus via a reputation score attack by masquerading as a High-Reputation Node (HRG) to tamper with the consensus result. However, reputation scores are based on historical behavior and dynamic adjustments (e.g., reputation value ρ and node behavior records), and A_v needs to guess the reputation scores of multiple nodes at the same time with restricted probability:

$$Pr\left[Coll_\rho\right] \leq \frac{1}{\rho} \tag{6}$$

The reputation scoring mechanism ensures that low-reputation nodes are removed from the HRG, limiting the success of A_v attack. Therefore, the increment of G_2's win rate is:

$$|Pr[G_2] - Pr[G_1]| \leq q_s \cdot \varepsilon_{SBPP} + \frac{q_s}{\rho} \tag{7}$$

After G_2 ends, A_v guesses bit b through testing queries. Due to the security and reputation score protection of SBPP issues, A_v's winning rate in G_2 is ultimately close to $\frac{1}{2}$. By combining the win rate G_0, G_1, G_2 and using the triangle inequality, we obtain:

$$Adv_{A_v}^{ROR}(t) = 2 \cdot \left| Pr[G_2] - \frac{1}{2} \right|, \tag{8}$$

$$\left| Pr[G_2] - \frac{1}{2} \right| \leq |Pr[G_1] - Pr[G_0]| + |Pr[G_2] - Pr[G_1]| \tag{9}$$

Substitute the probability boundaries of G_1 and G_2:

$$\left| Pr[G_2] - \frac{1}{2} \right| \leq \frac{q_h^2}{2 \cdot |Hash|} + q_s \cdot \varepsilon_{SBPP} + \frac{q_s}{\rho} \tag{10}$$

Finally, multiply both sides by 2 to obtain:

$$Adv_{A_v}^{ROR}(t) \leq \frac{q_h^2}{|Hash|} + 2 \cdot (q_s \cdot \varepsilon_{SBPP}) + 2 \cdot \frac{q_s}{\rho} \tag{11}$$

This formula indicates that the probability of A_v cracking the session key is limited by the collision resistance of the hash function, the difficulty of the SM9 bilinear pairing problem, and the protection of the reputation scoring mechanism, thereby proving the security of the proposed scheme under the ROR model.

5.3 Security Analysis of AVISPA Tool

To further verify the security of the proposed scheme, we employed formal security validation using the Automated Validation of Internet Security Protocols and Applications (AVISPA). AVISPA is a powerful and widely used open-source security protocol analysis toolkit [37], which supports protocol modelling, automatic verification, and result visualization, and is particularly suitable for complex distributed system scenarios such as IoV. Its back-end analysis engines include OFMC (On-the-Fly Model Checker), CL-AtSe (Constraint-Logic-based Attack Searcher), SATMC, and TA4SP, among which OFMC and CL-AtSe are the most commonly used and stable modules.

During the formal modelling process, the core entity roles including VU, RSU, ES and TA, are defined, and the corresponding session logic and communication environment are established. The protocol model covers the Registration, Authentication, Consensus and Re-Authentication phases, focusing on verifying the security of SM9 signatures, the Key Encapsulation Mechanism (KEM), and the RBSFT consensus. We verify the protocol through OFMC and CL-AtSe back-end engines separately to detect the presence of potential vulnerabilities.

As shown in Fig. 4a,b, the protocol is determined to be "SAFE" in both OFMC and CL-AtSe environments, and no feasible attack paths are found. This indicates that the proposed scheme can effectively resist common active and passive attacks, including message tampering, signature forgery, and malicious node jamming. The formal validation results of AVISPA further confirm the security and robustness of the protocols in highly dynamic IoV environments.

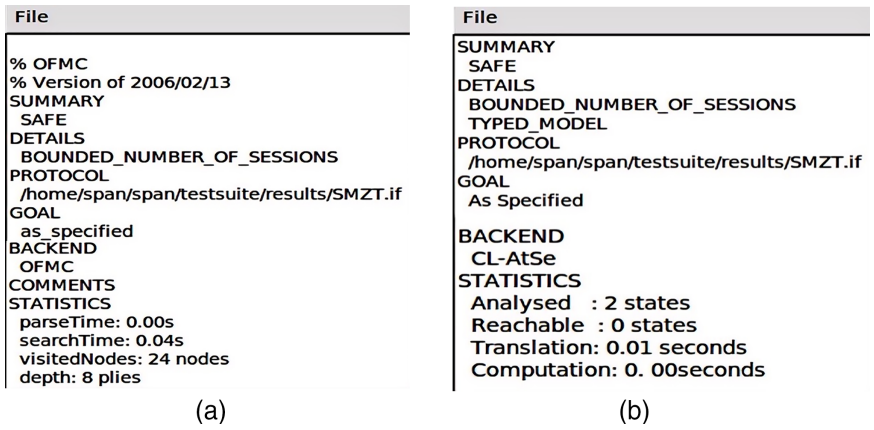

Figure 4: The simulation results of the proposed method in the AVISPA environment: (**a**) Under OFMC circumstance; (**b**) Under CL-AtSe circumstance

5.4 Informal Security Analysis

This section discusses the proposed scheme's defense against common attacks, which contains Sybil attacks, Masking attacks, Man-in-the-Middle attacks, Distributed Denial-of-Service (DDoS) attacks, Replay attacks, and Forgery attacks and so on, by an informal analysis.

5.4.1 Sybil Attack

In Sybil attack, an attacker tries to control the network or disrupt normal operations by forging multiple false identities (Sybil nodes). In the proposed scheme, the identification ID_{VU} of the VU is generated by hashing its MAC address ($ID_{VU} = H_1(MAC_{VU})$), which ensures the uniqueness and unforgeability of the identity. The VU stores the identity credentials C_{store} to the blockchain ledger during the registration phase, and the immutability and traceability of the ledger ensure that forged identities cannot be verified. The smart contract automatically checks the validity of the ID_{VU}, σ_{VU} in the authentication phase, preventing Sybil nodes from impersonating legitimate VUs for communication.

5.4.2 Masking Attack

Masking attack is when an attacker impersonates a legitimate entity (e.g., VU or RSU) to gain unauthorized access. In the proposed scheme, all entities must register with the TA to obtain SM9-based private keys (e.g., $d_{VU} = s \cdot H_1(ID_{VU}||hid)^{-1} \cdot P_1$) before joining the network. The authentication phase utilizes a mutual authentication mechanism, where VUs and RSUs need to verify each other's SM9 signatures (e.g., $\sigma_{VU}, \sigma_{RSU}$). An attacker who tries to disguise a legitimate VU needs to forge d_{VU} or σ_{VU}, but this requires cracking the SM9 bilinear pairing problem (SBPP), which has extremely high computational complexity. In addition, the timestamp T_{VU}, T_{RSU} in the authentication message ensure the freshness of the message and prevent the forger from using expired authentication data.

5.4.3 Man-in-the-Middle Attack

In a Man-in-the-Middle attack, the attacker intercepts and may tamper with communication messages between VU and RSU. In the proposed scheme, after passing through the SM9 KEM mechanism, the authentication message $\{C_{VU}, \sigma_{VU}\}$ can only be decrypted by the legitimate recipient using the corresponding private key, preventing attackers from directly accessing the message content. The T_1, T_2 in the message contain timestamps, and RSU and VU verify the freshness of the message by checking $|T_{now} - T_i| \leq \Delta T$,

preventing attackers from reusing intercepted messages. In addition, if an attacker attempts to tamper with a message, they need to regenerate a valid σ_{VU} or C_{VU}, but this requires cracking the SM9 signature or KEM, which is computationally infeasible due to the difficulty of SBPP. The SM9 aggregate signature σ_{agg} in the RBSFT consensus stage further ensures the integrity of the authentication results written into the ledger, preventing intermediaries from tampering with consensus data.

5.4.4 Distributed Denial of Service (DDoS) Attacks

DDoS attacks paralyze the system through a large number of invalid requests or forged signatures. To cope with this threat, the smart contract verifies the message timestamp during authentication, promptly discards invalid or malicious requests, and reduces the computational burden on RSUs. The RBSFT consensus mechanism identifies and isolates malicious nodes by reputation grouping (HRG and RG), reduces the reputation score of nodes that send false signatures or disrupt the consensus, and excludes HRGs from participating in the subsequent consensus. The distributed nature of the blockchain avoids a single point of failure, and the authentication results are stored in the distributed ledger in real-time, which enhances the DDoS resistance of the system.

5.4.5 Replay Attack

Replay attacks involve attackers repeatedly sending expired legitimate messages to deceive the system. In the proposed scheme, each authentication message contains a dynamic timestamp T_{VU}, T_{RSU}. RSU and VU ensure message freshness by checking $|T_{now} - T_i| \leq \Delta T$. The replayed old message was discarded because the timestamp exceeded the legal range. In addition, the random numbers r_{VU}, r_{RSU} are regenerated in each authentication to ensure the uniqueness of signatures $\sigma_{VU}, \sigma_{RSU}$, preventing attackers from using old messages to pass the verification. In the re-authentication stage, the smart contract queries the $Cert_{VU}$ in the ledger and verifies its SM9 signature and time block validity, further preventing replay attacks.

5.4.6 Forgery Attack

Forgery attacks involve an attacker forging authentication messages or signatures to spoof the system. In the proposed scheme, the authentication message is encrypted by SM9 KEM using the public key of the RSU and can be decrypted only by the private key of the RSU. The signature is generated based on the SM9 algorithm, which depends on a random number r, a hash value h and a private key d_{RSU}. The randomness and irreversibility of these parameters make it extremely difficult to forge signatures. The aggregated signatures in the consensus phase of RBSFT are based on the SM9 signature scheme. To forge valid signatures, the attacker must crack the private keys of multiple nodes. The resistance of SM9 signatures to forgery is based on the difficulty of solving the SBPP.

5.4.7 Smart Contract Vulnerability Attack

Attackers may exploit logic vulnerabilities (e.g., reentry attacks or overflow errors) in the smart contract code to corrupt the authentication process. The smart contract of the proposed scheme is deployed in the RSU with clear responsibilities, which is only responsible for triggering the authentication logic, verifying the timestamp and querying the $Cert_{VU}$ in the ledger, which has low code complexity and reduces the risk of vulnerability. The smart contract runs based on the blockchain, and its execution result is verified by all consensus nodes, and the error of a single node will not affect the overall result. The RBSFT consensus mechanism protects the ledger writes through reputation scoring and SM9 aggregated signatures, preventing attackers from utilizing contract vulnerabilities to tamper with the authentication result.

5.4.8 Resistance to ESL, Desynchronization, and Side-Channel Attacks

The proposed scheme resists ESL attacks in which temporary secrets (e.g., r, K') are compromised. Each authentication session generates a fresh r and K'. If an attacker compromises them, they cannot obtain long-term secrets (d_{VU}, d_{RSU}) due to the computational difficulty of the SBPP. The one-way hash function further ensures that leaks do not reveal prior or future keys, thus maintaining session independence.

The asynchronous attack disrupts the synchronization between VU and RSU by delaying or discarding messages. The scheme adopts internally validated timestamps (T_{VU},T_{RSU}) and discards delayed messages. Identity credentials ($Cert_{auth}$, $Cert_{init}$) are stored on the blockchain to ensure consistent node status. When VU switches RSU, the blockchain ledger is synchronized to prevent de synchronization ($Cert_{auth}$). RBSFT consensus ensures consistent updates to the ledger and reduces malicious interference.

Side-channel attacks exploit physical features such as time and power consumption. The scheme uses optimized SM9 operation (based on GMSSL/MIRACLE) and constant-time algorithm to defend against time and power consumption analysis. The RSU smart contract code is concise and reduces side-channel vulnerabilities. The RBSFT consensus mechanism mitigates the risk of side-channel attacks in consensus by restricting the participation of malicious nodes through reputation management.

5.4.9 Unlinkability, Indistinguishability, and Forward/Backward Secrecy

The scheme ensures unlinkability and prevents an attacker from linking multiple authentication sessions to the same VU. Each session uses a fresh r and T_{VU}, making messages independent across sessions. The SM9 signature and KEM cipher are randomized to ensure no correlation between sessions, even with the same ID_{VU}.

For an attacker without a private key, the authentication message is indistinguishable from random data. SM9 KEM encryption offers semantic security, as C_2 is computationally indistinguishable from a random string under the SBPP assumption. Similarly, SM9 signatures are randomized by r, which ensures indistinguishability from random pairs.

This scheme implements forward and backward confidentiality to ensure that leaked session keys do not affect historical or subsequent sessions. Generate new keys K' based on short-term random numbers r_{VU} for each session. Since SM9 key exchange is based on the SBPP problem, key leakage of d_{VU}, d_{RSU} cannot be deduced. The randomness of r_{RSU} between keys ensures session independence and prevents information leakage.

6 Performance Evaluation

This section comprehensively evaluates the performance of the proposed SM9-based distributed identity authentication and RBSFT consensus scheme in the IoV environment, focusing on three key aspects: the efficiency of the RBSFT consensus algorithm, the computational and communication overhead in the authentication process, and the authentication delay under different network conditions.

6.1 Consensus Evaluation

The RBSFT consensus algorithm combines the advantages of PBFT, Raft, and SM9 aggregated signatures to achieve efficient and secure consensus in IoV. To evaluate the performance, RBSFT is compared to traditional PBFT and SG-PBFT [17]. Assuming that the network has N ESs, RBSFT and SG-PBFT use $N/2$ consensus nodes, and PBFT uses all N nodes. The communication cost is measured by the number of messages exchanged in each phase of consensus (Pre-prepare, Prepare, Commit, Reply).

For traditional PBFT, the communication cost is $(2N(N-1))$, because all nodes broadcast messages in the Prepare and Commit phases. SG-PBFT reduces it to $(N/2-1)\ (N/2+1)$ by restricting consensus to half of the nodes. In our RBSFT scheme, the use of SM9 aggregate signatures converts the Prepare phase into a unicast message to the leader node, and the Commit phase broadcasts a single aggregated signature. Besides, the reputation-based grouping mechanism ensures that only HRG participate, further reducing the overhead. The total communication cost of RBSFT is $(3(N/2-1))$. The exact expression is shown in Table 3.

Table 3: Comparison of communication costs for consensus algorithms

Consensus phase	PBFT	SG-PBFT	Proposed RBSFT
Pre-Prepare	$(N-1)$	$(N/2-1)$	$(N/2-1)$
Prepare	$((N-1)\ (N-1))$	$((N/2-1)\ (N/2-1))$	$(N/2-1)$
Commit	$(N(N-1))$	$(N/2-1)$	$(N/2-1)$
Total	$(2N(N-1))$	$((N/2-1)\ (N/2+1))$	$(3(N/2-1))$

As shown in Table 2, the communication complexity of PBFT and SG-PBFT is quadratic $O(N^2)$, while the RBSFT scheme achieves linear communication complexity $O(N)$. To quantify the performance of SM9 aggregate signatures, we implemented the SM9 signature algorithm based on the C++ GMSSL library. In the RBSFT consensus stage, the SM9 aggregate signature significantly reduces computational overhead by merging multiple signatures into a single verification. Fig. 5 shows that it takes approximately 0.45 ms to aggregate and verify 500 signatures, while verifying 500 signatures individually takes approximately 287 ms (500 × 0.574 ms). For scenarios with different numbers of nodes (N = 50 to 500, representing vehicles or RSU signature contributors), the verification time of SM9 aggregated signatures remains relatively stable (about 0.45–0.50 ms), while the single signature verification time increases linearly with the number of nodes (for example, about 28.7 ms for 50 nodes and 287 ms for 500 nodes). The results show that RBSFT effectively improves the scalability of large-scale IoT networks by utilizing aggregate signatures, thereby avoiding the problem of verification costs doubling with the increase of nodes. This is significantly better than the multiple individual verification methods of traditional PBFT and SG-PBFT.

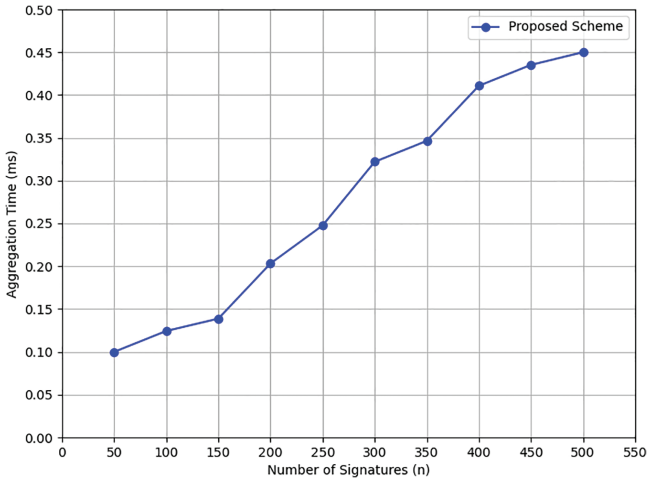

Figure 5: Time consumed for aggregating (N) signatures

6.2 Authentication Evaluation

6.2.1 Computational Cost

The computational cost of the proposed scheme is evaluated based on the execution time of cryptographic primitives in the registration, authentication, and re-authentication phases. We use the Multi-precision Integer and Rational Arithmetic Library (MIRACL) [38] to measure the runtime of key operations: one-way hash function T_h, SM9 signature generation T_{sig}, SM9 signature verification T_{ver}, and SM9 key encapsulation mechanism (KEM) encryption/decryption T_{kem}. Table 4 summarizes the approximate runtime of different cryptographic primitives based on MIRACL and applicable to SM9, based on experiments in [39,40]. Similarly, T_{ecm}, T_{eca}, T_{ckdf} can be obtained.

Table 4: Approximate runtime of different cryptographic primitives

Operations	Explanation	Time (ms)
T_h	One-way hash function	0.055
T_{sig}	SM9 signature generation	0.500
T_{ver}	SM9 signature verification	0.574
T_{kem}	SM9 key encapsulation mechanism	0.682
T_{ecm}	Elliptic curve point multiplication	0.675
T_{eca}	Elliptic curve point addition	0.005
T_{ckdf}	Cryptographic key derivation function	0.675

In the registration phase, the VU performs one hash operation T_h and one SM9 signature T_{sig}, while the TA performs one signature T_{sig} and one KEM encryption T_{kem}. The authentication phase involves mutual authentication between VU and RSU, requiring two hash operations $2T_h$ (VU, RSU), two signatures $2T_{sig}$ (VU, RSU), two verifications $2T_{ver}$ (VU, RSU), and two KEM operations $2T_{kem}$ (VU, RSU). The re-authentication phase between VU and RSU is lightweight, involving one hash T_h (VU), one SM9 T_{sig} (VU), two KEM encryption T_{kem} (VU, RSU), and one verification T_{ver} (RSU). The total computational cost is: $2T_h + 4T_{sig} + 3T_{ver} + 4T_{kem}$.

We compare our scheme with schemes [12,16,17,41–43], as shown in Table 5. Although ECC point multiplication (0.675 ms) is quicker, our scheme's certificate-free SM9 operations simplify key management, offsetting the slightly higher computational cost.

Table 5: Comparison of computational costs for various schemes

Schemes	Computational cost	Time (ms)
Yao et al. [12]	$21T_h + 12T_{ecm}$	9.255
Kumar et al. [16]	$15T_h + 10T_{ecm}$	7.575
Xu et al. [17]	$12T_h + 9T_{ecm}$	6.735
Eddine et al. [41]	$18T_h + 10T_{ecm} + 4T_{ckdf}$	10.44
Zhang et al. [42]	$10T_h + 10T_{ecm} + 12T_{eca}$	7.36
Dwivedi et al. [43]	$21T_h + 12T_{ecm}$	9.255
Proposed	$2T_h + 4T_{sig} + 3T_{ver} + 4T_{kem}$	6.56

6.2.2 Communication Cost

The communication cost is measured as the total number of bits exchanged during the registration, authentication, and re-authentication phases. We assume the following bit lengths: identity (160 bits), random number (160 bits), timestamp (32 bits), elliptic curve cryptography (ECC) (160 bits), SM9 signature (256 bits), SM9 KEM ciphertext (320 bits), and hash function (256 bits).

We compare the proposed scheme with [12,16,17,41,42] as shown in Fig. 6. In the scheme proposed by Eddine et al. [41], the number of exchanged messages is six, consisting of two message exchanges during the registration phase (1280 bits), three message exchanges during the authentication phase (2688 bits), and one message exchange during the re-authentication phase (448 bits). Therefore, the total communication cost is 4416 bits. In contrast, Zhang et al. [42] scheme requires six message interactions with a communication overhead of 4800 bits; The scheme proposed by Yao et al. [12] is implemented through four message interactions, with a communication overhead of 4000 bits; Kumar et al. [16] conducted a total of five message interactions with a communication overhead of 4500 bits; The scheme proposed by Xu et al. [17] also involves four message exchanges, with relatively low communication overhead of 3800 bits.

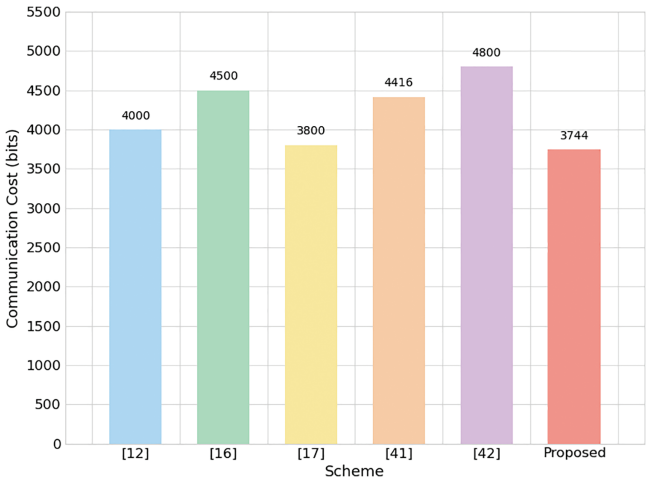

Figure 6: Comparison of communication cost

The communication costs of this scheme mainly come from the registration, authentication, and re-authentication stages. In the registration phase, the VU sends a message $\{ID_{VU}, T_{VU}, \sigma_{VU}\}$ of (160 + 32 + 256) = 448 bits, while the TA responds $C_{store} = (C, \{Cert_{init}, \sigma_{TA}\} \oplus K)$ with (320 + 512) = 832 bits, resulting in a total communication cost of 1280 bits. During the authentication phase, the VU transmits a message $\{ID_{VU}, r_{VU}, T_{VU}, \sigma_{VU}, C_{VU}\}$ of (160 + 160 + 32 + 256 + 320) = 928 bits, and the RSU replies $\{r_{RSU}, T_{RSU}, \sigma_{RSU}, C_{RSU}\}$ with (160 + 32 + 256 + 320) = 768 bits, leading to a total of 1696 bits. In the re-authentication phase, the VU sends a single message $\{ID_{VU}, T_{VU}, \sigma_{VU}, C_{VU}\}$, totaling (160 + 32 + 256 + 320) = 768 bits. Therefore, the overall communication cost of the proposed scheme amounts to (1280 + 1696 + 768) = 3744 bits. Compared with existing schemes, the proposed approach significantly reduces communication overhead while maintaining security and authentication efficiency, making it particularly suitable for resource-constrained and high-frequency interaction environments.

6.2.3 Authentication Delay

This section focuses on the performance evaluation of identity verification delay. The experiment uses the urban transportation simulation tool SUMO, and the relevant parameters are detailed in Table 6.

SUMO is an open-source traffic simulation software developed in C++, widely used to measure the average authentication delay in the IoV. Identity verification delay is defined as the total time spent on the entire process of registration, authentication, and re-authentication. Fig. 7a shows the test results at vehicle speeds of 10 m/s, 20 m/s, and 30 m/s. The results show that as the vehicle speed increases, the overall authentication delay shows a slight upward trend, but the difference in delay at different speeds is small, indicating that this scheme still has good stability and robustness in the dynamic environment of vehicle speed changes.

Table 6: Simulation setup

Parameters	Values
Simulation area	1000 m × 1000 m
Simulation time	30 s
Number of vehicles	50–200
Speed of vehicles	10–30 m/s
Wireless protocol	802.11 p
Channel bandwidth	6 Mbps

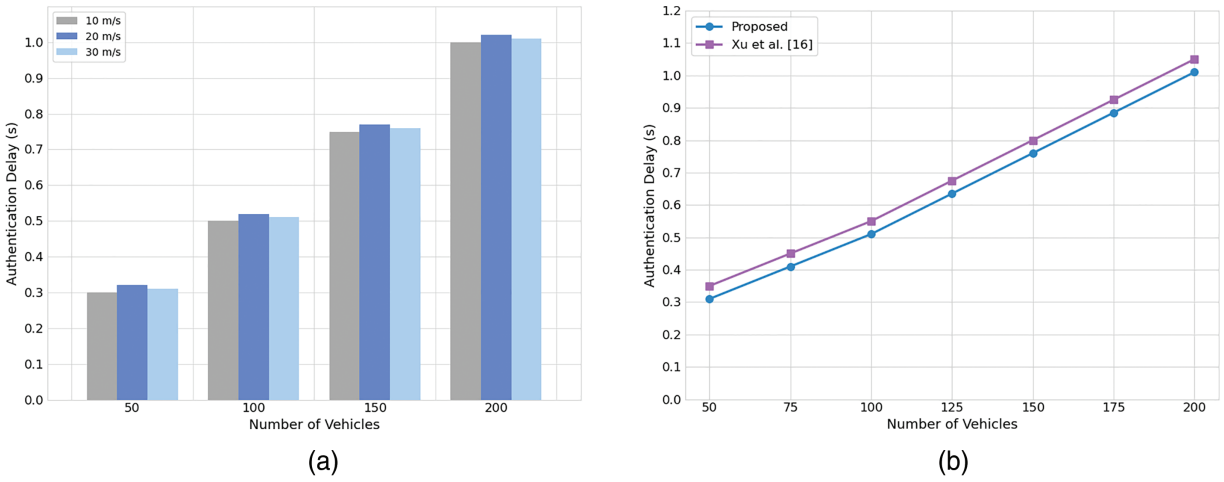

Figure 7: (**a**) Comparison of authentication delay for different speeds; (**b**) Average authentication delay

To further validate the performance, we compare the average authentication delay of the proposed scheme with Xu et al. [17], which employs the SG-PBFT consensus algorithm and has a computational cost (6.735 ms) and communication cost (3800 bits) comparable to our scheme (6.56 ms, 3744 bits). Fig. 7b shows a comparison of the average authentication delay for vehicle counts ranging from 50 to 200, with data points for every 25 vehicles. The proposed scheme's delay ranges from 0.31 to 1.01 s, consistently lower than Xu et al.'s 0.35 to 1.05 s. This advantage is attributed to the proposed scheme's optimized RBSFT consensus algorithm, which leverages SM9 aggregate signatures and reputation-based node grouping, and its lower communication overhead (3744 bits vs. 3800 bits).

7 Conclusion and Future

This paper proposes an efficient and secure distributed authentication scheme for the IoT based on SM9 and a reputation-driven consensus mechanism, aiming to solve the security and performance challenges

in highly dynamic and open network environments. Existing solutions are limited by complex certificate management and inefficient consensus protocols, which make it challenging to meet the low latency and high robustness requirements of IoV. Therefore, this paper introduces a dynamic node grouping strategy driven by smart contracts and reputation to enhance network stability and flexibility in trust management. The innovative RBSFT algorithm overcomes the limitations of traditional protocols, enhancing PBFT fault tolerance and Raft election efficiency by integrating SM9 lightweight signatures, while also reducing the computational burden. Compared with previous studies, RBSFT significantly optimizes communication complexity, improves consensus efficiency and system resilience, while ensuring security and meeting the real-time and scalability requirements of IoV.

Future research should focus on building small-scale prototypes or simulation systems to empirically verify end-to-end latency, throughput, and consensus success rates. Meanwhile, SM9 signature is efficient in typical applications, but its computational and storage overhead may still limit its application in large-scale networks. Combining federated learning technology is expected to improve the performance and system scalability of SM9, with significant research prospects.

Acknowledgement: We sincerely thank all the authors for their efforts.

Funding Statement: This study is fully supported by the National Natural Science Foundation of China (Grant No. 61762071, Grant No. 61163025).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Hui Wei and Zhanfei Ma; methodology, Hui Wei and Zhanfei Ma; validation, Hui Wei and Zhanfei Ma; formal analysis, Hui Wei, Jing Jiang and Bisheng Wang; writing—original draft preparation, Hui Wei and Zhanfei Ma; writing—review and editing, Hui Wei, Jing Jiang, Bisheng Wang and Zhong Di; visualization, Hui Wei and Zhanfei Ma; supervision, Hui Wei and Zhanfei Ma; project administration, Hui Wei, Zhanfei Ma, Jing Jiang, Bisheng Wang and Zhong Di; funding acquisition, Hui Wei and Zhanfei Ma. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data available on request from the authors.

Ethics Approval: No applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Aldhanhani T, Abraham A, Hamidouche W, Shaaban M. Future trends in smart green IoV: vehicle-to-everything in the era of electric vehicles. IEEE Open J Veh Technol. 2024;5(10):278–97. doi:10.1109/ojvt.2024.3358893.
2. Ji B, Chen Z, Mumtaz S, Han C, Li C, Wen H, et al. A vision of IoV in 5G HetNets: architecture, key technologies, applications, challenges, and trends. IEEE Netw. 2022;36(2):153–61. doi:10.1109/mnet.012.2000527.
3. Khezri E, Hassanzadeh H, Yahya RO, Mir M. Security challenges in internet of vehicles (IoV) for ITS: a survey. Tsinghua Sci Technol. 2025;30(4):1700–23. doi:10.26599/tst.2024.9010083.
4. Jatoth C, Doriya R. IoV block secure: blockchain based secure data collection and validation framework for internet of vehicles network. Peer Netw Appl. 2024;17(6):3964–90. doi:10.1007/s12083-024-01802-y.
5. Shah K, Chadotra S, Tanwar S, Gupta R, Kumar N. Blockchain for IoV in 6G environment: review solutions and challenges. Clust Comput. 2022;25(3):1927–55. doi:10.1007/s10586-021-03492-0.
6. Tu S, Yu H, Badshah A, Waqas M, Halim Z, Ahmad I. Secure Internet of Vehicles (IoV) with decentralized consensus blockchain mechanism. IEEE Trans Veh Technol. 2023;72(9):11227–36. doi:10.1109/tvt.2023.3268135.
7. Ji H, Zhang H, Shao L, He D, Luo M. An efficient attribute-based encryption scheme based on SM9 encryption algorithm for dispatching and control cloud. Connect Sci. 2021;33(4):1094–115. doi:10.1080/09540091.2020.1858757.

8. Zhang B, Li B, Zhang J, Wei Y, Yan Y, Han H, et al. An efficient SM9 aggregate signature scheme for IoV based on FPGA. Sensors. 2024;24(18):6011. doi:10.3390/s24186011.
9. Laghari AA, Khan AA, Alkanhel R, Elmannai H, Bourouis S. Lightweight-bIoV: blockchain distributed ledger technology (BDLT) for internet of vehicles (IoVs). Electronics. 2023;12(3):677. doi:10.3390/electronics12030677.
10. Zheng D, Jing C, Guo R, Gao S, Wang L. A traceable blockchain-based access authentication system with privacy preservation in VANETs. IEEE Access. 2019;7:117716–26. doi:10.1109/access.2019.2936575.
11. Noh J, Jeon S, Cho S. Distributed blockchain-based message authentication scheme for connected vehicles. Electronics. 2020;9(1):74. doi:10.3390/electronics9010074.
12. Yao Y, Chang X, Mišić J, Mišić VB, Li L. BLA: blockchain-assisted lightweight anonymous authentication for distributed vehicular fog services. IEEE Internet Things J. 2019;6(2):3775–84. doi:10.1109/jiot.2019.2892009.
13. Shen M, Duan J, Zhu L, Zhang J, Du X, Guizani M. Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE J Sel Areas Commun. 2020;38(6):1229–41. doi:10.1109/jsac.2020.2986619.
14. Bojjagani S, Reddy YP, Anuradha T, Rao PV, Reddy BR, Khan MK. Secure authentication and key management protocol for deployment of Internet of Vehicles (IoV) concerning intelligent transport systems. IEEE Trans Intell Transp Syst. 2022;23(12):24698–713. doi:10.1109/tits.2022.3207593.
15. Wang X, Zhu H, Ning Z, Guo L, Zhang Y. Blockchain intelligence for internet of vehicles: challenges and solutions. IEEE Commun Surv Tutor. 2023;25(4):2325–55.
16. Kumar A, Vishwakarma L, Das D. R-PBFT: a secure and intelligent consensus algorithm for Internet of vehicles. Veh Commun. 2023;41(7):100609. doi:10.1016/j.vehcom.2023.100609.
17. Xu G, Bai H, Xing J, Luo T, Xiong NN, Cheng X, et al. SG-PBFT: a secure and highly efficient distributed blockchain PBFT consensus algorithm for intelligent Internet of vehicles. J Parallel Distrib Comput. 2022;164:1–11. doi:10.1016/j.jpdc.2022.01.029.
18. Liu Y, Xiao M, Chen S, Bai F, Pan J, Zhang D. An intelligent edge-chain-enabled access control mechanism for IoV. IEEE Internet Things J. 2021;8(15):12231–41. doi:10.1109/jiot.2021.3061467.
19. El-Zawawy MA, Brighente A, Conti M. Authenticating drone-assisted Internet of Vehicles using elliptic curve cryptography and blockchain. IEEE Trans Netw Serv Manag. 2022;20(2):1775–89. doi:10.1109/tnsm.2022.3217320.
20. Karim SM, Habbal A, Chaudhry SA, Irshad A. BSDCE-IoV: blockchain-based secure data collection and exchange scheme for IoV in 5G environment. IEEE Access. 2023;11:36158–75. doi:10.1109/access.2023.3265959.
21. Chen C, Quan S. A summary of security techniques-based Blockchain in IoV. Secur Commun Netw. 2022;2022(1):8689651. doi:10.1155/2022/8689651.
22. Sharma V. An energy-efficient transaction model for the blockchain-enabled internet of vehicles (IoV). IEEE Commun Lett. 2018;23(2):246–9. doi:10.1109/lcomm.2018.2883629.
23. Mu Y, Xu H, Li P, Ma T. Secure two-party SM9 signing. Sci China Inf Sci. 2020;63(8):189101. doi:10.1007/s11432-018-9589-x.
24. Xu W, Ma H, Zhang R. GAPS: GPU-accelerated processing service for SM9. Cybersecurity. 2024;7(1):29. doi:10.1186/s42400-024-00217-9.
25. Dong S, Su H, Hou R, Shankar A. Improved PBFT consensus mechanism based on voting sort clustering partition with group signature for IoT. IEEE Trans Intell Transp Syst. 2024;26(2):2239–51. doi:10.1109/tits.2024.3495991.
26. Wu X, Wang Z, Li X, Chen L. DBPBFT: a hierarchical PBFT consensus algorithm with dual blockchain for IoT. Future Gener Comput Syst. 2025;162(5):107429. doi:10.1016/j.future.2024.07.007.
27. Zheng J, Li J. MGRS-PBFT: an optimized consensus algorithm based on multi-group ring signatures for blockchain privacy protection. IEEE Trans Netw Serv Manag. 2025;1. doi:10.1109/tnsm.2025.3580403.
28. Zheng J, Zhang Y. TRBFT: an efficient blockchain consensus for edge computing-enabled IoT systems. IEEE Internet Things J. 2025;12(11):15853–68. doi:10.1109/jiot.2025.3529723.
29. Deshmukh A, Baiju D, Atish A, Alladi T, Yu FR. An efficient and scalable byzantine fault tolerant consensus for vehicular networks. IEEE Trans Veh Technol. 2025;1–12. doi:10.1109/tvt.2025.3561360.
30. Lu L, Sun L, Zou Y. An efficient sharding consensus protocol for improving blockchain scalability. Comput Commun. 2025;231(4):108032. doi:10.1016/j.comcom.2024.108032.

31. Nakamoto S, Bitcoin A. A peer-to-peer electronic cash system. Bitcoin [Internet]. [cited 2025 Aug 6]. Available from: https://bitcoin.org/bitcoin.pdf.
32. Gabashvili N, Gabashvili T, Kiknadze M. From paper contracts to smart contracts. Sci Eur. 2022;2022(107):124–7.
33. Castro M, Liskov B. Practical byzantine fault tolerance. OsDI. 1999;99:173–86.
34. Ongaro D, Ousterhout J. The raft consensus algorithm. Lect Notes CS. 2015;190:2022.
35. Yuan F, Cheng Z. Overview on SM9 identity-based cryptographic algorithm. J Inf Secur Res. 2016;2(11):1008–27.
36. Suzuki K, Tonien D, Kurosawa K, Toyota K. Birthday paradox for multi-collisions. In: Proceedings of the Information Security and Cryptology–ICISC 2006: 9th International Conference; 2006 Nov 30–Dec 1; Busan, Republic of Korea.
37. Vigano L. Automated security protocol analysis with the AVISPA tool. Electron Notes Theor Comput Sci. 2006;155:61–86. doi:10.1016/j.entcs.2005.11.052.
38. Sdk MC. MIRACL Cryptographic SDK: multiprecision integer and rational arithmetic cryptographic library [Internet]. [cited 2022 Jun 1]. Available from: https://github.com/miracl/MIRACL.
39. Roy S, Nandi S, Maheshwari R, Shetty S, Das AK, Lorenz P. Blockchain-based efficient access control with handover policy in IoV-enabled intelligent transportation system. IEEE Trans Veh Technol. 2023;73(3):3009–24. doi:10.1109/tvt.2023.3322637.
40. Saha S, Bera B, Das AK, Kumar N, Islam SH, Park Y. Private blockchain envisioned access control system for securing industrial IoT-based pervasive edge computing. IEEE Access. 2023;11(2):130206–29. doi:10.1109/access.2023.3333441.
41. Eddine MS, Ferrag MA, Friha O, Maglaras L. EASBF: an efficient authentication scheme over blockchain for fog computing-enabled internet of vehicles. J Inf Secur Appl. 2021;59(3):102802. doi:10.1016/j.jisa.2021.102802.
42. Zhang H, Lai Y, Chen Y. Authentication methods for internet of vehicles based on trusted connection architecture. Simul Model Pract Theory. 2023;122:102681. doi:10.1016/j.simpat.2022.102681.
43. Dwivedi SK, Amin R, Vollala S, Khan MK. B-HAS: blockchain-assisted efficient handover authentication and secure communication protocol in VANETs. IEEE Trans Netw Sci Eng. 2023;10(6):3491–504. doi:10.1109/tnse.2023.3264829.