Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.087148

ARTICLE

A Lightweight Quantum-Secure Authentication and Key Agreement Protocol for Vehicular Ad-Hoc Networks

Lasseni Coulibaly[1,*], **Damien Hanyurwimfura**[1], **Evariste Twahirwa**[1] **and Abubakar Diwani**[2]

[1]African Center of Excellence in Internet of Things (ACEIoT), University of Rwanda, Kigali, Rwanda
[2]Department of Computer Science and IT, The State University of Zanzibar, Zanzibar, Tanzania

*Corresponding Author: Lasseni Coulibaly. Email: lassenicoulibaly@gmail.com

Received: 12 June 2026; Accepted: 21 July 2026; Published: 15 September 2026

ABSTRACT: Intelligent transportation systems, a critical pillar for smart cities, enable real-time vehicular communications to prevent human errors and improve traffic safety and efficiency. However, the open and highly dynamic nature of vehicular networks exposes them to various security threats, including message tampering, impersonation, and privacy violations. Several authentication and key agreement (AKA) protocols have been proposed to mitigate these risks, but often fail to maintain future-proof security against emerging quantum threats or introduce significant latency that affects real-time applications by relying on computationally expensive public-key cryptography, blockchain or centralized architectures. This paper proposes a new AKA protocol that combines hash-based operations with the Blom's key predistribution scheme and the standardized Module-Lattice-Based Key Encapsulation Mechanism (ML-KEM) to achieve lightweight and quantum-secure mutual vehicular authentications in a decentralized architecture. Formal security verifications together with informal security analysis demonstrate resistance against common network attacks, while performance evaluations confirm reduced computational costs up to 99.95% and communication overhead reductions from 8.46% to 87.08% compared with related methods. Moreover, NS-3 simulations under varying densities and mobility conditions demonstrate high packet delivery reliability, low end-to-end authentication latency, and excellent scalability, making the proposed protocol a practical solution for next-generation quantum-secure vehicular networks.

KEYWORDS: Authentication; key agreement; security and privacy; vehicular networks

1 Introduction

The rapid advancement of the Internet of Things has accelerated the integration of physical and digital systems, enabling intelligent transportation systems as a cornerstone of smart cities [1]. Within this ecosystem, vehicular ad hoc networks (VANETs) facilitate real-time communication among vehicles and roadside units (RSUs) to improve traffic safety, driving efficiency, and autonomous decision making. Vehicles equipped with onboard units (OBUs) continuously exchange safety-critical messages through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications to support cooperative awareness and proactive hazard avoidance. Despite these advantages, vehicular communications operate over an open and highly dynamic environment, making them vulnerable to numerous security and privacy threats as adversaries can launch message forgery, impersonation, replay, tracking, or denial-of-service (DoS) attacks that compromise communication reliability and road safety [2]. Consequently, authentication and key agreement (AKA) protocols are indispensable for ensuring that only legitimate entities participate in communication while preserving message authenticity, integrity, confidentiality, and user privacy.

Existing solutions predominantly rely on public-key cryptography (PKC), particularly elliptic curve cryptography (ECC), which provides strong classical security but incurs considerable computational overhead due to heavy operations, such as frequent signature generation, verification, and certificate management [3]. Blockchain-based authentication frameworks further improve trust decentralization and auditability [4,5] often using PKC primitives, but their consensus protocols and transaction validation introduce additional communication and computational latency that limits their applicability in real-time vehicular environments. Moreover, classical PKC schemes are fundamentally vulnerable to quantum attacks enabled by Shor's algorithm [6], motivating the transition toward post-quantum authentication mechanisms. Recent lightweight authentication schemes reduce computational complexity by replacing expensive public-key operations with symmetric cryptographic primitives, offering reduced computational complexity with quantum resistance [7]. However, many of these approaches often depend on centralized architectures or static shared secrets, creating scalability bottlenecks and increasing the impact of long-term key compromise. In particular, reliance on a central authority during real-time authentication introduces single points of failure and increases susceptibility to black-hole and DoS attacks. In light of these limitations, achieving a balanced design that simultaneously ensures security, privacy, scalability, and efficiency, while remaining resilient to emerging quantum threats remains an open research problem.

To overcome the above research challenge, this paper proposes the first decentralized and lightweight post-quantum hash-based AKA protocol that combines Blom's key pre-distribution scheme [8] with the Module-Lattice-Based Key Encapsulation Mechanism (ML-KEM) standardized by the National Institute of Standards and Technology (NIST) [9], for VANETs and Internet of Vehicles (IoV) environments. Unlike conventional approaches, the proposed protocol eliminates real-time dependence on a central authority during authentication, thereby reducing latency and removing critical bottlenecks. The Blom's scheme enables efficient pairwise key establishment through lightweight polynomial evaluations during routine authentication, whereas ML-KEM provides quantum-resistant protection for credential distribution and infrastructure-assisted key establishment. In addition, temporary pseudonymous identities protect user privacy through anonymity and unlinkability. To safeguard cryptographic credentials against physical compromise, vehicles and RSUs are assumed to be equipped with tamper-proof devices (TPDs).

The key contributions of this work are summarized below:

1. A lightweight and decentralized post-quantum AKA protocol supporting secure mutual authentication across V2V, vehicle-to-RSU (V2R), and RSU-to-RSU (R2R) communications without requiring real-time involvement of the trusted authority (TA), which is also adaptable to various networking applications.
2. A hybrid key establishment framework that combines Blom's symmetric scheme with ML-KEM to achieve efficient pairwise authentication while providing quantum-resistant credential distribution.
3. A forward secure and privacy-preserving authentication mechanism based on temporary pseudonym identities and session-specific keys to limit long-term credential exposure, while supporting a periodic security parameter refresh strategy and conditional traceability.
4. Formal security verifications, complemented by hardware computation benchmarking and NS-3 simulations demonstrating strong resistance against classical and quantum attacks while achieving low computation and communication costs with high scalability over existing schemes.

The remainder of this paper is organized as follows. Section 2 introduces the background and system models. Section 3 reviews the related literature. Section 4 presents the proposed protocol phases. Section 5 provides formal and informal security analyses. Section 6 evaluates cryptographic performance, while Section 7 presents the NS-3 simulation results before concluding the paper in Section 8.

2 Preliminaries

We describe the preliminaries for system and threat models, security goals, and computation tools.

2.1 System Model

The proposed system architecture considers a typical VANET environment primarily with the TA, RSU, and vehicles that interact to enable vehicular safety communications.

TA: A centralized authority that manages the network with sufficient capacities to register all vehicles and RSUs in a given location (e.g., country or state) and provide security updates to them. TA is the secured trust anchor assumed to be trustworthy and it uses secure channels to interact with vehicles and RSUs. Furthermore, TA has the ability to trace any misbehaving entity and revoke its credentials when needed.

RSU: Fixed infrastructure devices positioned along the roadway, RSUs monitor traffic conditions and disseminate safety information to vehicles within their communication ranges. RSUs can also communicate among themselves to exchange long-range traffic data with each other, and act as intermediaries to facilitate communication between the TA and vehicles, particularly during security updates and misbehavior reports.

Vehicle: Mobile entities equipped with OBUs, vehicles communicate with one another and with RSUs. Vehicles process exchanged safety information to maintain a cooperative driving environment.

2.2 Threat Model

Vehicles and RSUs communicate through insecure wireless channels and they remain vulnerable to various attacks. Therefore, an adversary ($\mathcal{A}$) has the following capabilities:

- $\mathcal{A}$ can observe vehicular communications to track vehicle behaviors or identities (privacy violation).
- $\mathcal{A}$ can intercept transmitted messages, drop, or modify them to mislead receivers (Man-in-the-Middle).
- $\mathcal{A}$ can generate fake messages and broadcast them to attempt unauthorized authentications (forgery).
- $\mathcal{A}$ can retransmit previously captured valid messages to attempt unauthorized authentications (replay).
- $\mathcal{A}$ can attempt to extract stored secret keys from a legitimate entity to use them (physical compromise).
- $\mathcal{A}$ can launch resource-exhaustion attacks by flooding with excessive messages (denial-of-service–DoS).

Assumption 1: *TA is the trust anchor and its database is protected against attack capabilities of $\mathcal{A}$. TPDs are secured and $\mathcal{A}$ cannot extract any useful information from a TPD in a realistic time. All entities use a synchronized clock to enable message freshness and other timely operations.*

2.3 Security Objectives

To counter the aforementioned threats, the proposed protocol aims to achieve the following objectives:

- *Mutual Authentication*: Entities verify each other's identity before establishing communication sessions.
- *Message Integrity*: A receiver verifies whether the content of a transmitted message has been modified or not, and rejects manipulated messages. Therefore, $\mathcal{A}$ should not be able to forge any valid message.
- *Privacy Preservation*: Vehicles and RSUs communicate without exposing their real identities. Therefore, $\mathcal{A}$ should not be able to derive a real identity or link different messages to the same sender to trace it.
- *Session Key Agreement*: Vehicles and RSUs establish unique session keys per communication instance to ensure confidentiality and forward secrecy. Thus, $\mathcal{A}$ should not be able to derive a valid session key.
- *Resistance to Common Attacks*: The protocol should resist common attacks such as replay attacks, storage compromise attacks, and brute force attacks under both classical and quantum computers.

2.4 Security Primer for Computation Tools

The proposed protocol employs Blom's scheme, ML-KEM together with an authenticated encryption with associated data (AEAD) mechanism, and lightweight cryptographic operations based on the secure hash algorithm (SHA-256) complemented by the bitwise exclusive-OR (XOR). These primitives collectively provide efficient mutual authentication, secure credential distribution, and low-latency session key establishment.

Definition 1 (ML-KEM): *Let* $(EK, DK) = \mathsf{MLKEM.KGen}()$ *be a valid ML-KEM key pair. For any encapsulated secret K as* $(C, K) = \mathsf{MLKEM.Encaps}(EK)$*, the decapsulation algorithm correctly recovers the same shared secret,* $K = \mathsf{MLKEM.Decaps}(DK, C)$*, except with negligible probability. This correctness follows the deterministic design of ML-KEM. Moreover, the ML-KEM security relies on reduction to the hardness of the Module Learning With Errors (M-LWE) problem, proving that no probabilistic polynomial-time adversary can distinguish the encapsulated shared secret from a uniformly random value, even under adaptive chosen-ciphertext attacks. Consequently, the ML-KEM is assumed fully quantum secure under the hardness of M-LWE.*

Definition 2 (AEAD): *Given a symmetric key K, the AEAD encryption algorithm protects the confidentiality and integrity of a plaintext M by producing a ciphertext* $C = \mathsf{AEAD.Enc}_K(M)$*. The corresponding decryption algorithm can efficiently recover the original message as* $M = \mathsf{AEAD.Dec}_K(C)$ *or output* $\perp$ *if the ciphertext has been modified. In the proposed protocol, AEAD and ML-KEM are complementary such that ML-KEM establishes temporary symmetric keys that are immediately used by AEAD to protect secret credentials and domain keys.*

Definition 3 (Blom): *Let* $f(x, y)$ *be a chosen bivariate symmetric polynomial of degree d over a finite field* $\mathbb{F}_p$ *with prime order p, such that* $f(x, y) = f(y, x)$ *for all* $x, y \in \mathbb{F}_p$*. By definition,* $f(x, y)$ *can be expressed as:*

$$f(x, y) = \sum_{i=0}^{d} \sum_{j=0}^{d} a_{ij} x^i y^j \pmod{p} \tag{1}$$

where $a_{ij} \in \mathbb{F}_p$ *is the coefficient matrix satisfying* $a_{ij} = a_{ji}$ *for all* $0 \le i, j \le d$*. Therefore,* $f(x, y)$ *allows any two distinct nodes to independently compute an identical algebraic factor using their local polynomial slices.*

Let two legitimate nodes with identifiers ID_i *and* ID_j *possess the polynomial shares* $g_i(y) = f(ID_i, y)$ *and* $g_j(y) = f(ID_j, y)$*, respectively given by TA. To verify each other's identifier, the first node computes* $K_{ij} = g_i(ID_j) = f(ID_i, ID_j)$*, while the second computes* $K_{ji} = g_j(ID_i) = f(ID_j, ID_i)$*. Since* $f(x, y)$ *is symmetric, any two legitimate nodes can independently derive the same algebraic factor such that* $f(ID_i, ID_j) = f(ID_j, ID_i)$ *and* $K_{ij} = K_{ji}$ *if and only if* $g_i(y)$ *and* $g_j(y)$ *were provided by the same TA. As long as the coefficient matrix* a_{ij} *remains secret, the Blom's scheme inherits the information-theoretic security: any adversary possessing at most d polynomial shares* $(g_1(y), g_2(y), \cdots, g_d(y))$ *gains no information about uncompromised pairwise keys. Moreover, the Blom's construction does not rely on algebraic problems vulnerable to Shor's algorithm, making the derived shared factor secure against currently known quantum attacks with sufficiently large prime order p.*

Definition 4 (SHA-256): *The hash function, denoted* $\mathcal{H}(.)$*, performs one-way mathematical operations on any arbitrary input to produce a fixed-length output that is hard to reverse or collide. Given two inputs* x_1 *and* x_2*:*

One way: Given a hash value $y_1 = \mathcal{H}(x_1)$*, reversing this function to recover* x_1 *such that* $x_1 = \mathcal{H}^{-1}(y_1)$ *is computationally hard. Likewise, for* $Y = \mathcal{H}(x_1 \parallel x_2)$*, knowing both Y and* x_1 *does not reveal the unknown* x_2 *(*$\parallel$ *denotes concatenation). Consequently, finding a preimage of an n-bit hash requires classically testing* 2^n

operations, while the best known quantum attacks (e.g., Grover's algorithm) reduce this complexity to about $2^{n/2}$. *Therefore, SHA-256 offers a classical and quantum resistance of* 2^{256} *and* 2^{128} *operations, respectively.*

Collision resistance: with distinct inputs $x_1 \neq x_2$, *it is computationally hard to find* $\mathcal{H}(x_1) = \mathcal{H}(x_2)$. *Due to the Birthday search across an n-bit output space, an adversary can evaluate approximately* $2^{n/2}$ *random inputs to achieve a* 50% *probability of finding a hash collision. Thus SHA-256 bounds a* 2^{128} *collision complexity.*

Avalanche effect: A small change in the input produces a statistically independent output. Hence, knowing $\mathcal{H}(x_1)$ *does not provide meaningful information about* $\mathcal{H}(x_2)$ *even when* x_1 *and* x_2 *differ by only one bit.*

Definition 5 (XOR): *Let* x_1 *and* x_2 *be two binary strings of equal length k. The bitwise XOR operation produces an output string denoted* $y = x_1 \oplus x_2$, *where the i-th bit* $y[i] = x_1[i] \oplus x_2[i]$, *satisfying self-invertibility and commutativity such that* $x_1 = y \oplus x_2$ *and* $x_2 = y \oplus x_1$. *While XOR itself does not provide cryptographic security, combining it with secure hash outputs and secret keys prevents unauthorized recovery of protected data.*

3 Related Works

In this section, we review related AKA protocols for vehicular networks, classifying them into three categories: PKC-based, blockchain-assisted, and lightweight symmetric-key-based approaches.

PKC-based approaches have been widely adopted due to their ability to provide public verifiability. In particular, ECC is predominantly used through digital signatures with or without certificate-based authentication. For instance, AKA protocols in [10–12] rely on ECC signatures and bilinear pairings, often supporting signature aggregation to enable batch authentication and group session key establishment. Although these techniques reduce authentication delay in dense vehicular environments, they introduce practical limitations where a single invalid signature may invalidate the entire aggregated verification process or, if verification succeeds, allow a malicious participant to obtain the group session key. Alternatively, Hoque et al. [13], Raghav et al. [14], and Yang et al. [15] have proposed individual AKA protocols that combine ECC with hash functions to reduce authentication latency while supporting V2V, V2R/V2I, and/or R2R communications. Jayashree and Santhosh Kumar [16] further combine ECC with bilinear pairing, providing enhanced security at the expense of significantly higher computational overhead. Despite the classical security guarantees, PKC-based schemes incur substantial computational overhead due to repeated heavy public-key operations and their mathematical problems remain fundamentally vulnerable to quantum attacks based on Shor's algorithm, limiting their long-term applicability.

Blockchain-assisted authentication has emerged as an alternative for decentralized trust management. Dwivedi et al. [17] and Lin and Jhuang [18] integrate blockchain with ECC to realize batch and certificateless authentication, respectively. Noh et al. [19] propose a non-interactive blockchain-based authentication framework for one-time entity verification, whereas Zhao et al. [20] combine zero-knowledge proofs, smart contracts, and Merkle trees for cross-domain authentication. Zhao et al. [21] further employ homomorphic encryption and ECC for anonymous authentication with dynamic pseudonym updates. More recently, Yan et al. [22] adopt lattice-based cryptography to achieve quantum resistance within a blockchain architecture. Lu et al. [23] integrate blockchain with Merkle Patricia Trees to support privacy-preserving authentication and certificate revocation. Although blockchain improves trust decentralization and auditability, these schemes suffer from transaction latency, consensus overhead, scalability limitations, and dependence on reliable connectivity, making them less suitable for latency-sensitive vehicular communications.

Lightweight symmetric-key AKA protocols replace expensive public-key operations to reduce computational complexity. Wazid et al. [24] proposed a cluster-based framework that reduces repeated authentication but introduces communication overhead and potential single points of failure through cluster-head management. Aman et al. [25] and Kim et al. [26] improve authentication efficiency using RSU-assisted

and two-factor authentication mechanisms, respectively, whereas decentralized protocols have also been proposed to improve scalability [27–30], with several incorporating biometric authentication to strengthen user verification. Although these approaches significantly reduce computational overhead and naturally resist Shor's algorithm, many continue to depend on centralized infrastructures, static pre-shared symmetric keys, limiting scalability, forward secrecy, and resilience against long-term key compromise.

Overall, existing AKA protocols exhibit complementary strengths and limitations. PKC-based schemes provide strong classical security but incur high computational overhead and remain vulnerable to quantum attacks, while blockchain-assisted approaches offer decentralized trust at the cost of significant communication and processing latency. Lightweight hash-based protocols improve authentication efficiency but often depend on centralized infrastructures or static shared secrets, limiting scalability and long-term security. Table 1 summarizes the security and privacy comparison of representative schemes.

Table 1: Comparative security and privacy features.

Security Features	[13]	[14]	[21]	[22]	[26]	[29]	Proposed
V2V Authentication	✓	×	✓	×	×	×	✓
V2R (V2I) Authentication	✓	✓	✓	✓	✓	✓	✓
R2R Authentication	✓	×	×	×	×	×	✓
Message integrity	✓	✓	✓	✓	✓	✓	✓
Identity Anonymity	✓	×	✓	×	✓	✓	✓
External Unlinkability	✓	✓	×	×	×	×	✓
Formal proof	✓	✓	✓	✓	✓	✓	✓
Resist impersonation	✓	✓	✓	✓	✓	✓	✓
Resist replay attack	✓	✓	×	×	✓	✓	✓
Resist black-hole attack	✓	✓	×	×	×	✓	✓
Resist session key attack	✓	✓	✓	✓	✓	✓	✓
Resist storage attack	✓	×	✓	×	×	✓	✓
Quantum resistance	×	×	×	✓	✓	✓	✓

Note: ✓: Feature is supported; ×: Feature is not supported.

Although most existing methods support V2R/V2I authentication, secure session key establishment, message integrity, impersonation resistance, and formal security verification, many do not natively support V2V and R2R communications. While anonymity is generally preserved, the schemes in [14,22] expose real identities to communicating peers, and several methods fail to ensure unlinkability by transmitting a static identifier across multiple authentication sessions. Moreover, the protocols in [22,26] continuously rely on an intermediate authority to achieve V2R authentication, while [21] depends on RSU availability to achieve V2R authentication before enabling V2V authentications. These central entity-dependent architectures create a risk of single points of failure. Although replay protection is commonly achieved using timestamps, methods like [21,22] do not bind timestamps to message authentication, allowing replay with refreshed timestamps. In addition, the schemes in [14,22,26] provide no explicit protection against storage compromise. Finally, ECC-based methods like [13,14,21] remain vulnerable to quantum attacks. In sum, the proposed protocol offers a more robust and balanced security profile by combining hash functions, Blom's scheme, and ML-KEM to provide decentralized, low-latency, and quantum-secure authentication with dynamic privacy preservation, periodic credential updates, and support for V2V, V2R, and R2R communications.

4 Proposed AKA Design

Fig. 1 illustrates the vehicular network architecture, in which the proposed AKA protocol enables post-quantum and low-latency mutual authentication and session key establishment for V2V, V2R, and R2R communications using ML-KEM for secure key encapsulation and Blom's key pre-distribution scheme for efficient pairwise key derivation. The proposed AKA protocol consists of six phases. First, the TA initializes the system by generating the global cryptographic parameters. Second, it registers vehicles and RSUs by assigning pseudonymous identities, generating their Blom polynomial shares, and securely distributing these credentials using ML-KEM. Third, neighboring entities perform mutual authentication and establish a pairwise session key using their polynomial shares and fresh timestamps, which is then used to protect subsequent unicast safety messages, while temporary pseudonyms preserve user privacy. Fourth, each RSU securely distributes its domain key to vehicles within its coverage area through ML-KEM encapsulation, enabling secure RSU-domain broadcast communications. Fifth, the protocol periodically renews security credentials to mitigate long-term key compromise. Finally, it preserves network integrity by identifying and revoking malicious nodes. The notations used throughout the protocol are summarized in Table 2.

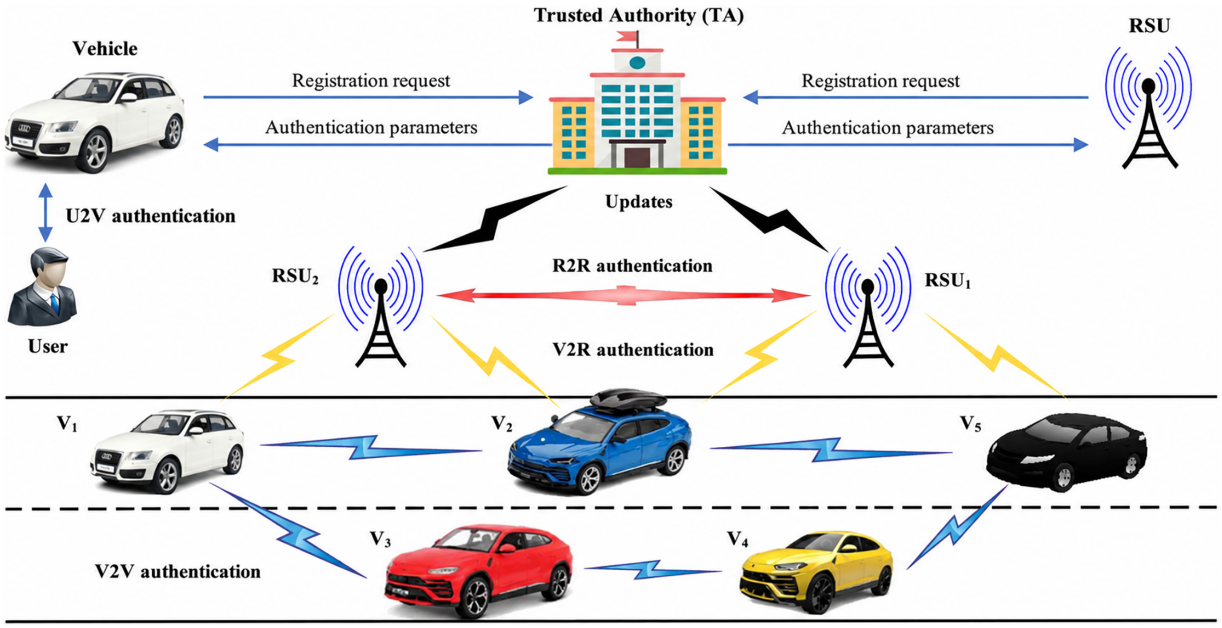

Figure 1: Network model for vehicular authentication.

Table 2: Important notations.

Symbols	Description
ID_{V_i}; ID_{R_k}; ID_{TA}	Real identities of a vehicle V_i, RSU R_k, and Trusted Authority (TA).
PID_{V_i}; PID_{R_k}	Pseudonymous identities of a vehicle V_i and RSU R_k.
TID_{V_i}; TID_{R_k}	Temporary identities of a vehicle V_i and RSU R_k.
(EK_{V_i}, DK_{V_i})	ML-KEM encapsulation/decapsulation key pair of a vehicle V_i.
K_{TA}	Master secret key of the TA.
K^{g}_{auth}, K^{ij}_{auth}	Global authorization key and Pairwise authentication key between entities i and j.
$f(x,y)$ and $A = (a_{ij})_{d\times d}$	Symmetric bivariate polynomial of degree d and its secret coefficient matrix.
$g_{V_i}(y)$	Blom's polynomial share assigned to a vehicle V_i.

(Continued)

Table 2 (continued)

Symbols	Description
$Auth_1, Auth_3, Auth_5$	Authentication tokens.
$K_{ij}; SK_{ij}$	Blom's pairwise secret key and Session key established between entities i and j.
$K^D_{R_k}$	Broadcast domain key of RSU R_k.
T_1; T_2; T_3 and ΔT	Authentication timestamps and their freshness time interval.
$\mathcal{H}(\cdot)$	SHA-256 one-way hash function.
$\oplus$; $\|$	XOR and concatenation operations.

4.1 System Initialization Phase

Before network deployment, the TA initializes the system parameters. It first selects a unique identity ID_{TA} and randomly generates two secrets (X, Y) to derive the master secret key $K_{TA} = \mathcal{H}(ID_{TA} \| X)$ and a global authorization key $K^g_{auth} = \mathcal{H}(ID_{TA} \| Y)$. The TA then selects a large prime p defining the finite field $\mathbb{F}_p$ and generates a symmetric coefficient matrix $A = (a_{ij})_{d\times d}$ to construct the Blom's symmetric polynomial $f(x, y)$ as defined in Eq. (1). It also specifies the message freshness threshold ΔT and security update interval T_{up}. The TA securely stores K_{TA}, A, and $f(x, y)$, while distributing K^g_{auth} to registered vehicles and RSUs for lightweight message filtering before full authentication. Finally, the TA selects the ML-KEM and AEAD schemes (e.g., the Advanced Encryption Standard in Galois/Counter Mode (AES-GCM)).

4.2 Registration Phase

The TA registers all vehicles and RSUs in offline or via secure channels following the same procedure but with their respective credentials as shown in Fig. 2. The registration steps are detailed as follows:

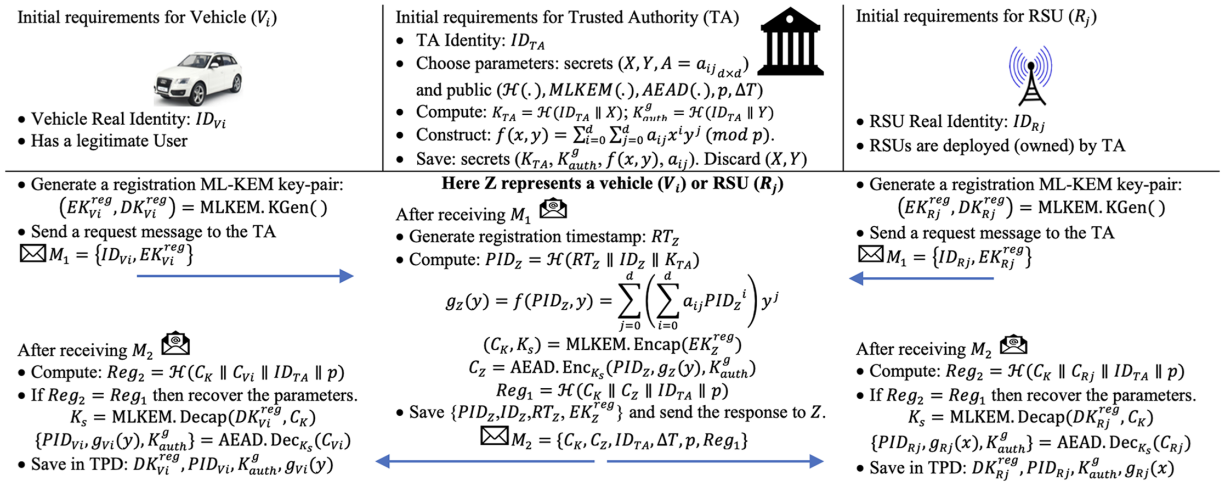

Figure 2: Vehicle and RSU registration.

Step 1: Each vehicle (V_i) has a unique identity (ID_{Vi}) and is expected to authenticate its user (e.g., using the method in [31]) to prevent illegal use. Before joining the network, V_i generates a registration ML-KEM encapsulation/decapsulation key pair as $(EK^{reg}_{Vi}, DK^{reg}_{Vi})$ = MLKEM.KGen(), and requests registration by sending its real identity and its public key to the TA as $M_1 = \{ID_{Vi}, EK_{Vi}\}$ (same for RSUs).

Step 2: TA receives the registration request M_1 and verifies the vehicle's real identity (ID_{V_i}). It then generates a registration timestamp (RT_{V_i}) and derives a pseudonym identity $PID_{Vi} = \mathcal{H}(RT_{Vi} \| ID_{Vi} \|$

K_{TA}) for the vehicle to enable anonymous communications while preserving conditional traceability by TA. Next, the TA evaluates the symmetric bivariate polynomial at $x = PID_{V_i}$ to generate the vehicle's identity-bound polynomial share $g_{V_i}(y)$ as defined in Eq. (2). To securely distribute the registration credentials, the TA uses the vehicle's encapsulation key $EK_{V_i}^{reg}$ to generate a shared secret K_s protected into a ciphertext C_K via the ML-KEM algorithm, which is then used to secure the registration tuple $\{PID_{V_i}, g_{V_i}(y), K_{auth}^{g}\}$ to produce a ciphertext C_{Vi} via a quantum-secure symmetric encryption algorithm as in Eq. (3). Subsequently, TA derives a verification code $Reg_1 = \mathcal{H}(C_K \parallel C_{Vi} \parallel ID_{TA} \parallel p)$, stores $(ID_{V_i}, RT_{V_i}, PID_{V_i}, EK_{V_i}^{reg})$ in its database and finally returns the registration response $M_2 = \{C_K, C_{Vi}, ID_{TA}, \Delta T, p, Reg_1\}$.

$$g_{Vi}(y) = f(PID_{Vi}, y) = \sum_{j=0}^{d}\left(\sum_{i=0}^{d} a_{ij}(PID_{Vi})^i\right) y^j \pmod{p} \tag{2}$$

$$(C_K, K_s) = \mathsf{MLKEM.Encaps}(EK_{V_i}^{reg}); \quad C_{Vi} = \mathsf{AEAD.Enc}_{K_s}\left(PID_{V_i} \parallel g_{V_i}(y) \parallel K_{auth}^{g}\right) \tag{3}$$

Step 3: Upon receiving M_2, the vehicle verifies the verification code Reg_1 and recovers the shared secret through ML-KEM decapsulation to decrypts C_{Vi} as in Eq. (4). The vehicle then stores sensitive registration data $\{DK_{V_i}^{reg}, PID_{V_i}, g_{V_i}(y), K_{auth}^{g}\}$ in its TPD and terminates the registration process.

$$K_s = \mathsf{MLKEM.Decaps}(DK_{V_i}^{reg}, C_K); \quad \{PID_{V_i}, g_{V_i}(y), K_{auth}^{g}\} = \mathsf{AEAD.Dec}_{K_s}(C_{Vi}) \tag{4}$$

4.3 Mutual Authentication and Session Key Establishment

After completing the registration phase, any two neighboring vehicles V_i and V_j can mutually authenticate each other and establish a shared session key without requiring further assistance from the TA. As illustrated in Fig. 3a, each vehicle stores its pseudonymous identity PID, the TA identity ID_{TA}, the global authorization key K_{auth}^{g}, the authentication validity interval ΔT, and the update period T_{up} in its TPD. The authentication protocol combines temporary identities, timestamp verification, and Blom's key pre-distribution to provide anonymous mutual authentication while establishing a common session key.

Step 1: When vehicle V_i intends to communicate with vehicle V_j, it first generates a fresh timestamp T_1 and computes a temporary identity $TID_{V_i} = PID_{V_i} \oplus \mathcal{H}(K_{auth}^{g} \parallel T_1)$, which conceals its real pseudonym from external observers. It then uses the global authorization key to compute the authentication token $Auth_1 = \mathcal{H}(TID_{V_i} \parallel T_1 \parallel K_{auth}^{g})$, and transmits the authentication request to vehicle V_j as $M_1 = \{TID_{V_i}, T_1, Auth_1\}$.

Step 2: Upon receiving M_1, vehicle V_j records the current timestamp T_2 and verifies whether the received message satisfies the freshness condition $|T_2 - T_1| \le \Delta T$. If valid, it recomputes the authentication token $Auth_2 = \mathcal{H}(TID_{V_i} \parallel T_1 \parallel K_{auth}^{g})$, and compares it with the received value. Successful verification indicates the sender is authorized allowing V_j to proceed and recover the pseudonym $PID_{V_i} = TID_{V_i} \oplus \mathcal{H}(K_{auth}^{g} \parallel T_1)$. Using its stored Blom polynomial share, V_j computes the pairwise secret $K_{ji} = g_{V_j}(PID_{V_i}) \pmod{p}$, which is further used to derive the temporary authentication key $K_{auth}^{ji} = \mathcal{H}(K_{ji} \parallel K_{auth}^{g} \parallel T_2)$. Vehicle V_j then generates its own temporary identity $TID_{V_j} = PID_{V_j} \oplus \mathcal{H}(K_{auth}^{g} \parallel T_2)$, computes the authentication token $Auth_3 = \mathcal{H}(TID_{V_j} \parallel T_2 \parallel K_{auth}^{ji})$, and replies with message $M_2 = \{TID_{V_j}, T_2, Auth_3\}$.

Step 3: After receiving M_2, vehicle V_i first verifies the timestamp freshness by checking whether $|T_3 - T_2| \le \Delta T$. It subsequently reconstructs the responder's pseudonym and also computes the pairwise secret $K_{ij} = g_{V_i}(PID_{V_j}) \pmod{p}$. According to the properties of Blom's key pre-distribution scheme, both vehicles independently obtain the same pairwise secret, such that $K_{ij} = K_{ji}$. Therefore, vehicle V_i derives the temporary authentication key $K_{auth}^{ij} = \mathcal{H}(K_{ij} \parallel K_{auth}^{g} \parallel T_2)$, which is unique and only known to the two vehicles (V_i and V_j) preventing session impersonation by other insider entities. The vehicle then

computes $Auth_4 = \mathcal{H}(TID_{V_j} \parallel T_2 \parallel K_{auth}^{ij})$ and verifies the received authentication token by comparing $Auth_4$ with $Auth_3$. If vehicle V_j is successfully authenticated, vehicle V_i computes the confirmation token as $Auth_5 = \mathcal{H}(K_{ij} \parallel T_3)$ and sends the message $M_3 = \{T_3, Auth_5\}$ to vehicle V_j.

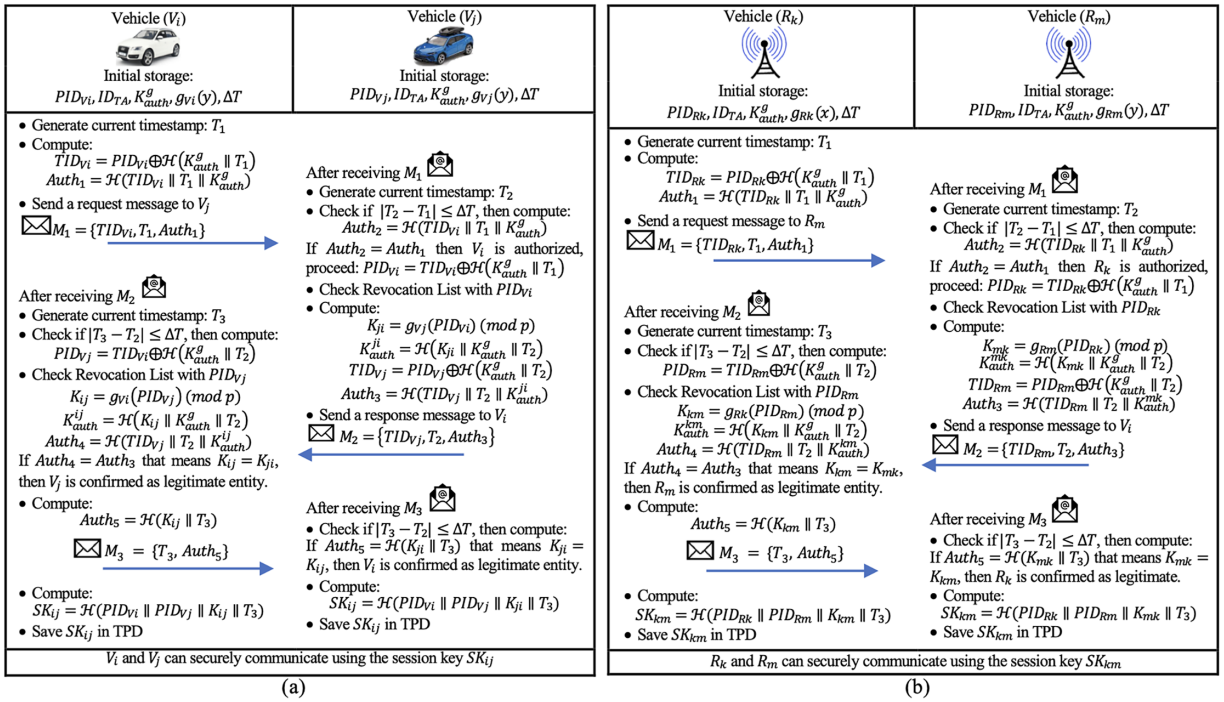

Figure 3: Mutual authentication and key agreement phase: (**a**) vehicle-to-vehicle and (**b**) RSU-to-RSU.

Step 4: Finally, vehicle V_j verifies the received confirmation by checking whether $Auth_5 = \mathcal{H}(K_{ji} \parallel T_3)$. Since $K_{ij} = K_{ji}$, successful verification confirms that both vehicles are effectively registered by the same TA and the exchanged pseudonym identities are legitimate leading to the same pairwise secret. Therefore, they have mutually authenticated each other and both entities can independently derive the same session key as $SK_{ij} = \mathcal{H}(PID_{V_i} \parallel PID_{V_j} \parallel K_{ij} \parallel T_3)$, which is securely stored in their respective TPDs and subsequently used to protect their V2V communications in the active session.

Note that this mutual authentication and session key establishment procedure is applicable for all direct vehicular communication modes, including the V2V (Fig. 3a), R2R (Fig. 3b), and V2R (Fig. 4a) interactions. Each session key is valid only for one active session due to the inclusion of a fresh session timestamp, which prevents key reuse across different communication sessions and mitigates session key capture attacks.

4.4 V2R Mutual Authentication and Domain Key Distribution

When a vehicle enters the communication range of an RSU, they execute the mutual V2R authentication process where the RSU uses the ML-KEM encapsulation to securely distribute its domain broadcast key to the vehicle as illustrated in Fig. 4b. Consequently, authenticated vehicles can securely receive broadcast traffic messages within the RSU coverage area. In addition, vehicles and RSUs can achieve unicast communications when necessary by executing a mutual authentication and session key establishment as shown in Fig. 4a.

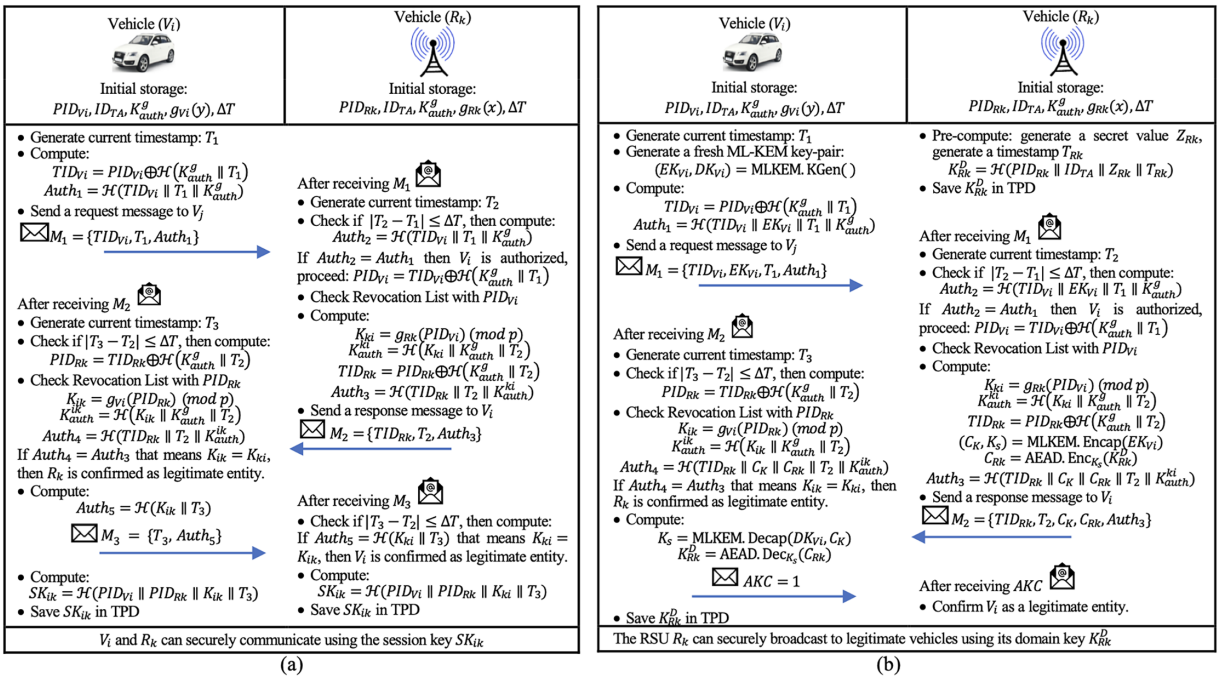

Figure 4: Vehicle-to-RSU authentication phase: (**a**) session key agreement and (**b**) domain key sharing.

Step 0: Each RSU R_k independently generates its domain key $K^D_{R_k} = \mathcal{H}(PID_{R_k} \| ID_{TA} \| Z_{R_k} \| T_{R_k})$ using a fresh random secret Z_{R_k} and current timestamp T_{R_k}, which can be periodically updated for security.

Step 1: To initiate the authentication process, vehicle V_i generates the current timestamp T_1 and a fresh ML-KEM encapsulation/decapsulation key pair (EK_{V_i}, DK_{V_i}). It then computes the temporary identity $TID_{V_i} = PID_{V_i} \oplus \mathcal{H}(K^g_{auth} \| T_1)$ and the authentication token $Auth_1 = \mathcal{H}(TID_{V_i} \| EK_{V_i} \| T_1 \| K^g_{auth})$. The authentication request $M_1 = \{TID_{V_i}, EK_{V_i}, T_1, Auth_1\}$ is subsequently transmitted to the RSU R_k.

Step 2: After receiving M_1, the RSU verifies the freshness of the received timestamp and authenticates the vehicle V_i by recomputing the received authentication token. Upon successful verification, the RSU R_k reconstructs the vehicle pseudonym, computes the Blom pairwise secret $K_{ki} = g_{R_k}(PID_{V_i}) \pmod p$, and derives the temporary authentication key $K^{ki}_{auth} = \mathcal{H}(K_{ki} \| K^g_{auth} \| T_2)$. The RSU R_k then generates its temporary identity $TID_{R_k} = PID_{R_k} \oplus \mathcal{H}(K^g_{auth} \| T_2)$, uses the vehicle's ML-KEM encapsulation key to generate a shared secret as $(C_K, K_s) = \mathsf{MLKEM.Encaps}(EK_{V_i})$ and encrypts its current domain broadcast key $C_{R_k} = \mathsf{AEAD.Enc}_{K_s}(K^D_{R_k})$. Finally, the RSU computes the authentication token $Auth_2 = \mathcal{H}(TID_{R_k} \| T_2 \| C_K \| C_{R_k} \| K^{ki}_{auth})$ and sends the response $M_2 = \{TID_{R_k}, T_2, C_K, C_{R_k}, Auth_2\}$ to the requesting vehicle.

Step 3: Upon receiving the response M_2, vehicle V_i verifies the freshness of the received timestamp and reconstructs the RSU pseudonym. It then computes the Blom pairwise secret $K_{ik} = g_{V_i}(PID_{R_k}) \pmod p$. Due to the symmetry property of Blom's polynomial, both entities obtain the same secret $K_{ik} = K_{ki}$. The vehicle derives $K^{ki}_{auth} = \mathcal{H}(K_{ik} \| K^g_{auth} \| T_2)$ and verifies $Auth_2$ to confirm the authenticity of the RSU R_k. It then decapsulates the received ML-KEM ciphertext using its private decapsulation key $K_s = \mathsf{MLKEM.Decaps}(DK_{V_i}, C_K)$ to recover the RSU domain broadcast key $K^D_{R_k} = \mathsf{AEAD.Dec}_{K_s}(C_{R_k})$. The vehicle securely stores the recovered domain key $K^D_{R_k}$ in its TPD and optionally sends an acknowledgement confirming successful key reception. After completion of this phase, V_i becomes an authenticated member of the RSU domain and can securely receive subsequent broadcast traffic messages protected by $K^D_{R_k}$.

4.5 Security Update Phase

To limit the long-term exposure of cryptographic material, the proposed scheme incorporates a periodic security update mechanism. This phase primarily refreshes the global authorization key K_{auth}^{g}, which is shared among registered vehicles and RSUs for lightweight authorization check before further verification.

Re-initialization: The TA achieves an offline re-initialization before any update period following the steps in Section 4.1. It refreshes the global authorization key $K_{auth}^{g,new}$ for each update interval and reconstructs the symmetric bivariate polynomial $f^{new}(x, y)$ to renew entities' Blom polynomial shares $g_{V_i}^{new}(y)$ only when required. The updated parameters replace the previous ones after the current T_{up} expires.

Distribution: TA announces the security update period T_{up} to allow credential refresh in advance. Each entity can send an update request $U_{req} = \{TID_{V_i}, T_1\}$ before T_{up} ends, where $TID_{V_i} = PID_{V_i} \oplus \mathcal{H}(K_{auth}^{g} \parallel T_1)$. Upon receiving the request, the TA verifies the entity's legitimacy, protects the refreshed credentials using ML-KEM and AEAD with the entity's registered encapsulation key $EK_{V_i}^{reg}$, and returns the response $U_{resp} = \{C_K, C_{Vi}, Auth_1\}$, where $(C_K, K_s) = \mathsf{MLKEM.Encaps}(EK_{V_i}^{reg})$, $C_{Vi} = \mathsf{AEAD.Enc}_{K_s}\left(g_{V_i}^{new}(y) \parallel K_{auth}^{g,new}\right)$, and $Auth_1 = \mathcal{H}(C_K \parallel C_{Vi})$. Since $EK_{V_i}^{reg}$ belongs to a legitimate entity, only the intended receiver possessing the corresponding decapsulation key $DK_{V_i}^{reg}$ can recover the protected credentials, preventing unauthorized entities from requesting updates on behalf of legitimate participants. During the transition period, entities continue using their current credentials until T_{up} expires. After the update deadline, all legitimate vehicles and RSUs switch to the refreshed parameters for subsequent authentication operations. This coordinated update strategy prevents credential inconsistencies and ensures continuous network availability.

4.6 Revocation Phase

The proposed protocol provides conditional traceability, allowing the TA to identify and revoke malicious or compromised entities while preserving the anonymity of legitimate users.

Misbehavior Reporting: When a vehicle or RSU detects invalid behavior or suspicious activity, it forwards the corresponding authentication messages as evidence to the TA for verification.

Identity Tracing and Revocation: Upon receiving a valid misbehavior report, the TA reconstructs the reported pseudonym as $PID = TID \oplus \mathcal{H}(K_{auth}^{g} \parallel T)$ using the corresponding timestamp and authorization key. TA then maps the identified pseudonym to the associated real identity from its registration database. If the reported entity is confirmed as malicious, its pseudonym PID is added to the revocation list (RL).

Revocation Dissemination: To enable a dynamic revocation enforcement, the TA immediately publishes the RL after every update to be accessible by RSUs. Each RSU frequently broadcasts the latest RL to vehicles within its communication range, enabling them to verify whether the received pseudonym belongs to the current revocation list before initiating mutual authentication. Since vehicles and RSUs cannot forge valid pseudonyms, any entity associated with a revoked pseudonym is immediately rejected by legitimate peers even if its previous credentials remain valid within the current update interval T_{up}. Therefore, revoked entities are excluded from new session key establishments and subsequent security updates.

5 Security Analysis

In this section, we analyze the security guarantees of the proposed AKA protocol through an informal analysis and a formal correctness analysis and a formal security verification.

5.1 Informal Security Analysis

5.1.1 Mutual Authentication and Message Integrity

The proposed protocol provides mutual authentication between communicating entities through a challenge-response mechanism based on hash verification and Blom's pairwise secret computation. During authentication, each entity validates the received authentication token using the derived pairwise secret and the global authorization key. Since only legitimate nodes possess valid polynomial shares and authentication parameters, an adversary cannot generate a valid authentication response. Consequently, both communicating parties are assured of each other's legitimacy before establishing a session key. Moreover, the computed hash-based authentication token assures that any modification of an intercepted message changes the hash input and causes the receiver's verification to fail. Therefore, forged or modified messages are detected and discarded to preserve message integrity against man-in-the-middle attacks.

5.1.2 Privacy (Anonymity and Unlinkability)

In the proposed protocol, the real identity ID of a communicating entity is never transmitted directly during authentication. Instead, the TA assigns each legitimate vehicle and RSU a unique pseudonymous identity (PID) during registration. Since recovering ID from PID requires knowledge of the TA master key K_{TA} and the one-way property of the hash function, an adversary cannot infer the real identity from intercepted messages. Moreover, every authentication session generates a temporary identity $TID = PID \oplus \mathcal{H}(K_{auth}^{g} \parallel T)$, using the global authorization key and a fresh timestamp, which produces different authentication messages in different sessions to ensure that intercepted messages cannot be correlated by an external attacker to identify or track a particular vehicle over time. Therefore, the proposed protocol preserves both anonymity and unlinkability of legitimate vehicles and RSUs.

5.1.3 Replay Attacks Resistance

Every authentication message includes a fresh timestamp that is verified upon reception using the condition $|T_r - T_s| \leq \Delta T$, where T_s and T_r denote the sending and receiving timestamps, respectively. Any replayed message outside the valid time interval is immediately rejected. Even if an adversary modifies the timestamp, the associated authentication token becomes invalid because it is computed over the original timestamp. Consequently, the proposed protocol effectively prevents replay attacks.

5.1.4 Impersonation (Even from Insider) Attack Resistance

An adversary attempting to impersonate a legitimate vehicle or RSU must generate valid authentication messages and establish a valid session key. External adversaries cannot accomplish this because they lack the required authorization key K_{auth}^{g} and legitimate Blom's polynomial shares $g_i(y)$. However, compromise of K_{auth}^{g} alone may allow an external adversary to bypass the initial authorization verification and initiate the pairwise authentication process. Nevertheless, possessing K_{auth}^{g} is insufficient even for insiders to fully impersonate another legitimate entity or establish a valid session key, which requires the entity-specific $g_i(y)$ to compute the Blom pairwise secret $K_{ij} = g_i(PID_j) \pmod{p}$. Therefore, the proposed scheme resists complete impersonation attacks under the assumption that legitimate entities' TA-issued polynomial shares remain protected. Thus, a compromise of K_{auth}^{g} is considered a partial credential exposure that affects preliminary message filtering only but does not reveal pairwise authentication secrets or session keys.

5.1.5 Resistance to Node Capture and Storage Intrusion

The proposed protocol protects sensitive authentication parameters, including pseudonymous identities, polynomial shares, authorization keys, and session keys, by storing them in trusted TPDs. These tamper-resistant hardware modules significantly reduce the risk of unauthorized extraction of cryptographic material through physical attacks. Furthermore, the protocol inherits the threshold security property of Blom's key pre-distribution scheme. Specifically, compromising up to d legitimate vehicles or RSUs reveals only their individual polynomial shares and provides no information about the secret polynomial or uncompromised pairwise secrets. An adversary must compromise at least $d + 1$ distinct nodes to reconstruct the secret polynomial. Therefore, the combined use of secure hardware and Blom's threshold property provides strong resistance against storage intrusion and node capture attacks.

5.1.6 Resistance to Session Key Capture Attacks

In the proposed protocol, the session key is never transmitted over the communication channel. Instead, both communicating entities independently derive the identical fresh session key from their polynomial shares, pseudonym identities, and fresh timestamps after successful mutual authentication. Consequently, each session key is unique and independent of previously established session keys, ensuring that even if an adversary compromises a session key, it cannot derive past or future session keys without compromising the Blom's polynomial share. Moreover, a fresh ML-KEM key pair is generated whenever it is used during credential or domain key distribution, preventing the reuse of cryptographic material across sessions and further limiting the impact of key compromise. Therefore, the protocol is secure against session key attacks.

5.1.7 Brute Force Attack Resistance

The proposed protocol employs SHA-256 and ML-KEM to protect its authentication parameters, requiring approximately 2^{256} operations under classical computation to recover secret values through exhaustive search. Even under Grover's quantum search algorithm, the effective complexity remains 2^{128}, providing a sufficient post-quantum security margin. Hence, the protocol is resistant to brute-force attacks.

5.1.8 Resistance to Flooding Attacks

The protocol minimizes the computational impact of flooding attacks through early message validation. Received messages first undergo timestamp verification to immediately discard stale messages before any expensive cryptographic operation is performed, while authentication relies primarily on lightweight hash computations and Blom polynomial evaluations. Furthermore, the decentralized authentication process eliminates continuous dependence on the TA, reducing the risk of bottlenecks and distributed DoS attacks.

5.2 Formal Security Verification Using BAN Logic

To prove the structural correctness and cryptographic soundness of the proposed mutual authentication and key agreement mechanism, a formal deductive analysis is conducted using Burrows-Abadi-Needham (BAN) Logic [32]. This analysis verifies that the participating nodes can securely establish a common session key (SK_{ij}) while ensuring mutual authentication, freshness, and protection against entity impersonation.

Notation 1: *Let P and Q be two communicating entities; X and Y be information; and K be a security key. The standard syntactic operators of BAN logic are defined as follows:*

- $P \overset{K}{\longleftrightarrow} Q$: *K is a secret key shared between P and Q.*
- $\{X\}_K$: *The information X is encrypted using the key K.*
- $P \Rightarrow X$: *P controls or has jurisdiction over the information X.*
- $\#(X)$: *The information X is fresh.*
- $P \mid\sim X$: *P once said the information X.*
- $Q \triangleleft X$: *Q sees the information X.*
- $Q \mid\equiv X$: *Q believes the information X is true.*
- $P \overset{Y}{\rightleftharpoons} Q$: *Y is a shared secret known only to principals P and Q.*
- $\langle X \rangle_Y$: *Formula X is combined with the secret parameter Y.*
- (X, Y): *X and Y form a compound message.*

The foundational inference rules of BAN logic applied in this proof include:

R1: *The message meaning rule:* $\dfrac{\textit{If } P \mid\equiv (P \overset{K}{\longleftrightarrow} Q) \textit{ and } P \triangleleft \{X\}_K}{\textit{then } P \mid\equiv Q \mid\sim X}$

R2: *The nonce-verification rule:* $\dfrac{\text{If } P \mid\equiv \#(X) \text{ and } P \mid\equiv Q \mid\sim X}{\text{then } P \mid\equiv Q \mid\equiv X}$

R3: *The jurisdiction rule:* $\dfrac{\text{If } P \mid\equiv (Q \Rightarrow X) \text{ and } P \mid\equiv Q \mid\equiv X}{\text{then } P \mid\equiv X}$

R4: *The freshness rule:* $\dfrac{\text{If } P \mid\equiv \#(X)}{\text{then } P \mid\equiv \#(X, Y)}$

R5: *The belief rule:* $\dfrac{\text{If } P \mid\equiv X \text{ and } P \mid\equiv Y}{\text{then } P \mid\equiv (X, Y)}$

Assumption 2: *The verification proof is evaluated under the following initial beliefs and baseline assumptions regarding parameter distribution and clock synchronization:*

A1 (the sender can verify timestamp freshness): $V_i \mid\equiv \#(T_1), \quad V_i \mid\equiv \#(T_2), \quad V_i \mid\equiv \#(T_3)$

A2 (the receiver can verify timestamp freshness): $V_j \mid\equiv \#(T_1), \quad V_j \mid\equiv \#(T_2), \quad V_j \mid\equiv \#(T_3)$

A3 (each entity received the global authorization key): $V_i \mid\equiv V_i \overset{K^g_{\text{auth}}}{\longleftrightarrow} V_j$ ***and*** $V_j \mid\equiv V_i \overset{K^g_{\text{auth}}}{\longleftrightarrow} V_j$

A4 (each entity received its secret polynomial slice): $V_i \mid\equiv V_i \overset{g_{V_i}(y)}{\rightleftharpoons} \text{TA}$ ***and*** $V_j \mid\equiv V_j \overset{g_{V_j}(y)}{\rightleftharpoons} \text{TA}$

By the Blom's mathematical structure in Definition 3, evaluating local secret polynomial slices creates a unique, mutually verifiable pairwise secret $K_{ij} = K_{ji}$. This yields the following critical jurisdiction assumptions:

A5 (Pairwise Secret Ownership Authority): $V_i \mid\equiv (V_j \Rightarrow V_i \overset{K_{ij}}{\longleftrightarrow} V_j)$ ***and*** $V_j \mid\equiv (V_i \Rightarrow V_i \overset{K_{ji}}{\longleftrightarrow} V_j)$

Assumption 3: *The actual over-the-air protocol packets (M_1, M_2, M_3) are converted into their idealized logical forms to capture the exact cryptographic assertions transmitted between vehicle V_i and vehicle V_j:*

Idealized $M_1\ (V_i \to V_j) :\ \{PID_{V_i}, T_1\}_{K^g_{auth}}$

Idealized $M_2\ (V_j \to V_i) :\ \{PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j\}_{K^{ji}_{\text{auth}}} \quad$ where $K^{ji}_{\text{auth}} = \mathcal{H}(K_{ji} \parallel K^g_{\text{auth}} \parallel T_2)$

Idealized $M_3\ (V_i \to V_j) :\ \{T_3\}_{K_{ij}}$

Theorem 1: *To confirm that the session key configuration is fully secure and mutually authenticated, the deductive analysis must satisfy the following four fundamental security goals:*

G1: $V_j \mid\equiv V_i \mid\equiv (PID_{V_i})$: *The receiver* V_j *authenticates the origin of the request message* M_1.

G2: $V_i \mid\equiv V_j \mid\equiv (PID_{V_j})$: *The sender* V_i *authenticates the origin of the response message* M_2.

G3: $V_i \mid\equiv V_i \overset{SK_{ij}}{\longleftrightarrow} V_j$: V_i *confirms that it successfully establishes a session key* SK_{ij} *with* V_j.

G4: $V_j \mid\equiv V_i \overset{SK_{ij}}{\longleftrightarrow} V_j$: V_j *confirms that it successfully establishes a session key* SK_{ij} *with* V_i.

Proof of Theorem 1: The step-by-step application of BAN logic inference rules proceeds as follows:

Analysis of Message M_1: Vehicle V_j intercepts the initiation request message M_1 from the open channel:

S1: $V_j \lhd \{PID_{V_i}, T_1\}_{K^g_{\text{auth}}}$

By applying the *Message Meaning Rule* to **S1** combined with the global initialization in **A3**, we derive:

S2: $$\frac{V_j \mid\equiv V_i \overset{K^g_{\text{auth}}}{\longleftrightarrow} V_j,\ V_j \lhd \{PID_{V_i}, T_1\}_{K^g_{\text{auth}}}}{V_j \mid\equiv V_i \mid\sim (PID_{V_i}, T_1)}$$

Because vehicle V_j independently confirms the clock freshness of timestamp T_1 via **A2**, the *Freshness Rule* implies that the entire composite statement is fresh: if $V_j \mid\equiv \#(T_1)$ then $V_j \mid\equiv \#(PID_{V_i}, T_1)$. Applying the *Nonce Verification Rule* to **S2** combined with the message freshness yields:

S3: $$\frac{V_j \mid\equiv \#(PID_{V_i}, T_1),\ V_j \mid\equiv V_i \mid\sim (PID_{V_i}, T_1)}{V_j \mid\equiv V_i \mid\equiv (PID_{V_i}, T_1)}$$

Based on the *Belief Rule*, we can decompose the belief components from **S3** to directly achieves **Goal 1**:

S4: $V_j \mid\equiv V_i \mid\equiv (PID_{V_i})$ (Goal 1 Attained)

Analysis of Message M_2: Once V_i is successfully authenticated, V_j extracts PID_{V_i} and computes the temporary key K^{ji}_{auth} to build message M_2. Vehicle V_i intercepts M_2:

S5: $V_i \lhd \{PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j\}_{K^{ji}_{\text{auth}}}$

Since K^{ji}_{auth} is directly derived using the symmetric pairwise factor $K_{ij} = K_{ji}$ known only to V_i and V_j, V_i believes that K^{ji}_{auth} is a valid channel key. Applying the *Message Meaning Rule* with **S5** gives:

S6: $$\frac{V_i \mid\equiv V_i \overset{K^{ji}_{\text{auth}}}{\longleftrightarrow} V_j,\ V_i \lhd \{PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j\}_{K^{ji}_{\text{auth}}}}{V_i \mid\equiv V_j \mid\sim (PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j)}$$

By referencing the timestamp freshness of T_2 via **A1**, applying the *Nonce Verification Rule* to **S6** yields:

S7: $$\frac{V_i \mid\equiv \#(PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j),\ V_i \mid\equiv V_j \mid\sim (PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j)}{V_i \mid\equiv V_j \mid\equiv (PID_{V_j}, T_2, V_i \overset{K_{ji}}{\longleftrightarrow} V_j)}$$

Breaking down the conjunction vector in **S7** based on the *Belief Rule* achieves **Goal 2**:

S8: $V_i \mid\equiv V_j \mid\equiv (PID_{V_j})$ (Goal 2 Attained)

Concurrently, parsing the remaining element from **S7** yields:

S9: $V_i \mid\equiv V_j \mid\equiv (V_i \overset{K_{ji}}{\longleftrightarrow} V_j)$

Applying the *Jurisdiction Rule* to **S9** using the initial network authority setting in **A5** establishes:

S10: $$\frac{V_i \mid\equiv (V_j \Rightarrow V_i \overset{K_{ij}}{\longleftrightarrow} V_j),\ V_i \mid\equiv V_j \mid\equiv (V_i \overset{K_{ji}}{\longleftrightarrow} V_j)}{V_i \mid\equiv V_i \overset{K_{ij}}{\longleftrightarrow} V_j}$$

Analysis of Message M_3: Vehicle V_i returns the confirmation message, which V_j captures from the channel:

S11: $V_j \lhd \{T_3\}_{K_{ij}}$

Because V_j has already proven the origin belief of PID_{V_i} in **S4**, it relies on the polynomial properties in **A4** that govern the pairwise key derivation from local parameters, leading to the key boundary belief:

S12: $V_j \mid\equiv V_i \overset{K_{ji}}{\longleftrightarrow} V_j$

By applying the *Message Meaning Rule* to **S11** and **S12** yields: **S13:** $\dfrac{V_j \mid\equiv V_i \overset{K_{ji}}{\longleftrightarrow} V_j,\ V_j \lhd \{T_3\}_{K_{ij}}}{V_j \mid\equiv V_i \mid\sim T_3}$.

By integrating the freshness check T_3 via **A2**, the *Nonce Verification Rule* verifies V_i's active state:

S14: $$\frac{V_j \mid\equiv \#(T_3),\ V_j \mid\equiv V_i \mid\sim T_3}{V_j \mid\equiv V_i \mid\equiv T_3}$$

Session Key Establishment: The final session key is computed as $SK_{ij} = \mathcal{H}(PID_{V_i} \parallel PID_{V_j} \parallel K_{ij} \parallel T_3)$. Since vehicle V_i has verified the identity and active presence of V_j via **S8**, and holds a secure belief regarding the symmetric key component K_{ij} from **S10**, it can securely assert belief over the derived session key:

S15: $V_i \mid\equiv V_i \overset{SK_{ij}}{\longleftrightarrow} V_j$ (Goal 3 Attained)

Similarly, because vehicle V_j has verified the identity and active presence of V_i via **S4**, and holds a secure belief regarding the symmetric key component K_{ji} from **S12**, it asserts identical belief over the derived session key structure, satisfying **Goal 4:**

S16: $V_j \mid\equiv V_i \overset{SK_{ij}}{\longleftrightarrow} V_j$ (Goal 4 Attained)

Conclusion: The BAN logic verification confirms mutual authentication, message freshness, and session key establishment under the assumed trust conditions. The use of timestamps ensures protection against replay attacks, while the derivation of identity-bound temporary keys prevents adversarial impersonation and message forgery, validating the correctness of the protocol in satisfying its intended security properties. □

5.3 Formal Security Verification via AVISPA

To rigorously validate the security strength of the proposed AKA protocol against active malicious exploitation, a formal simulation is conducted using the industry-standard Automated Validation of Internet Protocols and Applications (AVISPA) tool [33]. Implementing a Dolev-Yao intruder model, AVISPA evaluates cryptographic transactions over an insecure channel where an adversary can intercept, modify, delete, or inject arbitrary messages. The architecture is formally modeled using the High-Level Protocol Specification Language (HLPSL), defining explicit state transitions, parameter spaces, and foundational network goals.

5.3.1 HLPSL Role Specifications

The proposed protocol is modeled in HLPSL using two primary roles: the initiator (V_i) and the responder (V_j/R_k). The initiator generates a fresh timestamp, constructs a temporary identity and authentication token, transmits the authentication request, verifies the responder's reply, computes the pairwise secret using its Blom polynomial share, and finally derives the session key after successful mutual authentication.

Similarly, the responder validates the received authentication request, verifies its freshness, computes the corresponding pairwise secret from its polynomial share, returns the authentication response, and independently derives the same session key upon validating the final confirmation message. Both roles are integrated into a *session* module and executed within an *environment* role that instantiates multiple concurrent sessions under the Dolev–Yao adversarial model. The intruder is assumed to possess public and compromised system parameters but has no access to entity-specific secrets, including the TA secrets and valid polynomial shares. Finally, the HLPSL specification defines the authentication and session key secrecy goals determined by polynomial shares, mutual identity and fresh timestamps verifications, and agreement on the established session key. The HLPSL specifications are provided in Fig. 5 to enable reproducibility.

```
%% 1. Role Specification for Initiator (Vehicle Vi)
role vehicle_Vi(Vi, Vj: agent, H,G: hash_func,
Pid_Vi, Slice_Vi: text, K_auth_g: symmetric_key,
SND, RCV: channel (dy))
played_by Vi
def= local
State: nat,
Pid_Vj, Tid_Vi, Tid_Vj, T1, T3, T2: text,
Auth1, Auth3, Auth5: text,
K_ij, K_auth_ij, Sk_ij: symmetric_key
init State := 0
transition
% Step 1: Vi initiates by sending M1
1. State = 0 /\ RCV(start) =|> State' := 1 /\ T1':= new()
/\ Tid_Vi' := xor(Pid_Vi, H(K_auth_g.T1'))
/\ Auth1' := H(Tid_Vi'.T1'.K_auth_g)
/\ SND(Tid_Vi'.T1'.Auth1')
/\ witness(Vi, Vj, vehicle_vi_vj_Auth1, T1')
% Step 3: Vi receives response M2, and sends M3
2. State = 1 /\ RCV({Tid_Vj'.T2'.Auth3'}_K_auth_ij)
=|> State' := 2 /\ Auth3' := H(Tid_Vj'.T2'.K_auth_ij')
/\ Pid_Vj':= xor(Tid_Vj', H(K_auth_g.T2'))
/\ K_ij' := G(Slice_Vi.Pid_Vj')
/\ K_auth_ij' := H(K_ij'.K_auth_g.T2') / \ T3' := new()
/\ Auth5':=H(K_ij'.T3')
/\ SND({T3'.Auth5'}_K_auth_ij')
/\ Sk_ij' := H(Pid_Vi.Pid_Vj'.K_ij'.T3')
/\ secret({K_ij',Sk_ij',K_auth_ij'},secret_sk_ij,{Vi,Vj})
/\ witness(Vi, Vj, vehicle_vi_vj_Auth5, Sk_ij')
/\ request(Vi, Vj, vehicle_vj_vi_Auth3, {T2', Auth3'})
end role
```

```
% 2. Role Specification for Responder (Vehicle Vj)
role vehicle_Vj(Vi, Vj: agent, H, G: hash_func,
Pid_Vj, Slice_Vj: text, K_auth_g: symmetric_key,
SND, RCV: channel (dy))
played_by Vj
def= local
State: nat,
Pid_Vi, Tid_Vi, Tid_Vj, T1, T3, T2_fresh: text,
Auth1, Auth3, Auth5: text,
K_ji, K_auth_ji, Sk_ij: symmetric_key
init State := 0
transition
% Step 2: Vj captures M1, verifies, and sends M2
1. State = 0 /\ RCV({Tid_Vi'.T1'.Auth1'}_K_auth_g)
=|> State' := 1 /\ Auth1' := H(Tid_Vi'.T1'.K_auth_g)
/\ Pid_Vi' := xor(Tid_Vi', H(K_auth_g.T1'))
/\ K_ji' := G(Slice_Vj.Pid_Vi') / \ T2_fresh' := new()
/\ K_auth_ji' := H(K_ji'.K_auth_g.T2_fresh')
/\ Tid_Vj' := xor(Pid_Vj, H(K_auth_g.T2_fresh'))
/\ Auth3' := H(Tid_Vj'.T2_fresh'.K_auth_ji')
/\ SND({Tid_Vj'.T2_fresh'.Auth3'}_K_auth_ji)
/\ secret({K_ji', K_auth_ji'}, secret_sk_ij, {Vi, Vj})
/\ witness(Vj,Vi,vehicle_vj_vi_Auth3,{T2_fresh',Auth3'})
/\ request(Vj, Vi, vehicle_vi_vj_Auth1, {T1',Auth1'})
% Step 4: Vj validates M3 and derives session key
2. State = 1 /\ RCV({T3'.Auth5'}_K_auth_ji')
=|> State' := 2 /\ Auth5' := H(K_ji.T3')
/\ Sk_ij' := H(Pid_Vi.Pid_Vj.K_ji.T3')
/\ secret(Sk_ij', secret_sk_ij, {Vi, Vj})
/\ request(Vj, Vi, vehicle_vi_vj_Auth5, Sk_ij')
end role
```

```
%% 3. Role Specification for the Session
role session(Vi, Vj: agent, K_auth_g: symmetric_key,
H, G: hash_func, Pid_Vi, Pid_Vj, Slice_Vi, Slice_Vj: text)
def= local
SND1, RCV1, SND2, RCV2: channel (dy)
composition
vehicle_Vi(Vi,Vj,Pid_Vi,Slice_Vi,K_auth_g,H,G,SND1,RCV1)
/\ vehicle_Vj(Vi,Vj,Pid_Vj,Slice_Vj,K_auth_g,H,G,SND2,RCV2)
end role
%% 4. Environment Role & Security Goals
role environment()
def= local
H, G: hash_func,
K_auth_g: symmetric_key,
Vi, Vj: agent,
Pid_Vi, Pid_Vj, Pid_I, Slice_Vi, Slice_Vj, Slice_I: text
const secret_sk_ij, vehicle_vi_vj_Auth1, vehicle_vj_vi_Auth3,
vehicle_vi_vj_Auth5: protocol_id
intruder_knowledge = {Vi, Vj, H, G}
composition
session(Vi,Vj,Pid_Vi,Pid_Vj,Slice_Vi,Slice_Vj,K_auth_g,H,G)
/\ session(Vi, i, Pid_Vi, Pid_I,Slice_Vi, Slice_I, K_auth_g,H,G)
/\ session(i, Vj,Pid_I, Pid_Vj, Slice_I, Slice_Vj, K_auth_g,H,G)
end role
goal
authentication_on vehicle_vi_vj_Auth1
authentication_on vehicle_vj_vi_Auth3
authentication_on vehicle_vi_vj_Auth5
secrecy_of secret_sk_ij
end goal
environment()
```

Figure 5: HLPSL role specifications.

5.3.2 AVISPA Simulation Results and Analysis

The HLPSL specification is formally verified using the AVISPA framework with its two principal verification backends: the On-the-Fly Model Checker (OFMC) and Constraint-Logic-Based Attack Searcher (CL-AtSe), which systematically explore the protocol state space to identify potential attacks, including replay, man-in-the-middle, and impersonation attacks. As shown in Fig. 6a,b, both analyses return a `SAFE` verdict, indicating that no attack trace violates the specified secrecy and authentication goals. Furthermore, an insider attack was evaluated by assuming that the adversary compromises the global authorization key K^{g}_{auth}. Under this assumption, AVISPA correctly reports an `UNSAFE` verdict, as illustrated in Fig. 6c. The generated attack trace shows that possession of K^{g}_{auth} only allows the adversary to bypass the first authorization check, but subsequent pairwise verifications fail. These results confirm that the protocol preserves session-key secrecy and mutual authentication as long as the Blom polynomial shares remain unavailable to the attacker, while effectively limiting the impact of partial compromised global credentials.

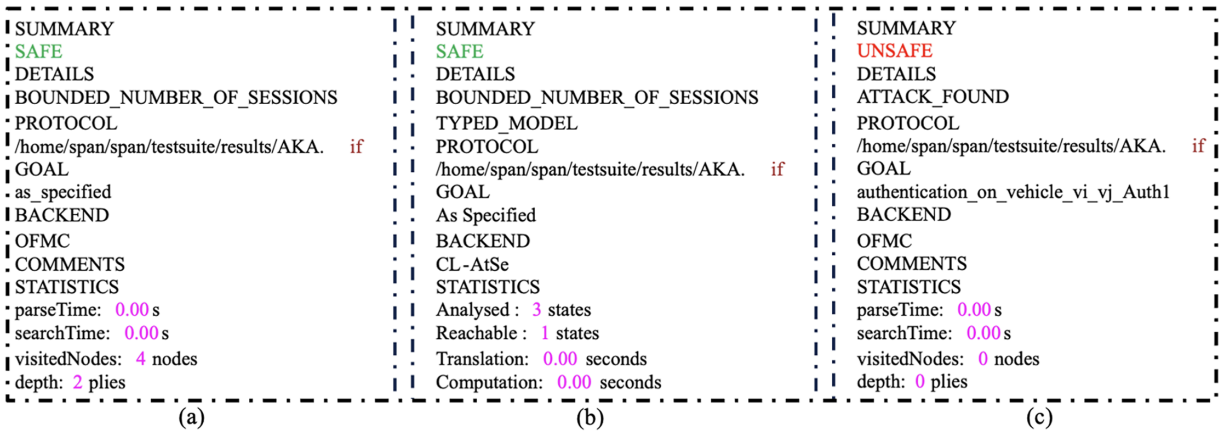

Figure 6: AVISPA results: (**a**) normal under OFMC, (**b**) normal under CL-AtSe, and (**c**) compromise under OFMC.

6 Performance Evaluation

In this section, we discuss the computational and communication performances of the proposed protocol phases, while comparing it against existing representative AKA schemes (referenced as [13,14,21,22,26,29]).

6.1 Computational Overhead

To evaluate the computational overhead, cryptographic benchmarks were conducted on an Apple M2 processor (10 cores, 16 GB RAM) running Python 3.13, as summarized in Table 3. ECC, AES, SHA-256, and the pseudo-random function (PRF) were implemented using the *cryptography* (OpenSSL) library, while ML-KEM was evaluated using the *pqcrypto* library [34]. Additional benchmarks, including bitwise XOR, ECC modular arithmetic, Blom polynomial evaluation, Paillier homomorphic encryption, Arbiter Physical Unclonable Function (PUF) delay-path simulation, and Fuzzy Extractor operations, were implemented using native Python arithmetic. Execution times were measured with Python's high-resolution $time.perf\_counter\_ns()$ timer, and the mean, standard deviation, minimum, and maximum values performed on a baseline 32 bytes input data over 1000 executions are reported to ensure reproducible measurements.

Table 3: Empirical benchmark of cryptographic operation times.

Operation Paradigm & Notations (ms)	Mean	Std	Min	Max
T_{xor}: Bitwise XOR Operation	0.001723	3.2e−05	0.001625	0.001958
T_{sha}: Hash Function (SHA-256)	0.001601	6e−05	0.001458	0.001958
$T_{E/D}$: AES-GCM Symmetric Encryption/Decryption	0.007727	0.000987	0.007333	0.022875
T_{prf}: Pseudo-Random Function (using HMAC-SHA256)	0.002868	0.000436	0.002667	0.009834
T_{puf}: Physical Unclonable Function (PUF)	0.015692	0.000943	0.015083	0.027667
T_{fe}: Fuzzy Extractor Generation or Reproduction operations	0.003368	0.000598	0.003208	0.014167
T_{blom}: Blom's Polynomial Evaluation	0.010471	0.000866	0.009958	0.029875
T_{mod}^{mul}: ECC Modular Multiplication	0.000693	0.000261	0.000625	0.008875
T_{mod}^{exp}: ECC Modular Exponentiation	0.175582	0.004626	0.171541	0.205458
T_{ecc}^{pa}: ECC Point Addition	0.041966	0.001674	0.040292	0.055625

(Continued)

Table 3 (continued)

Operation Paradigm & Notations (ms)	Mean	Std	Min	Max
T_{ecc}^{sm}: ECC Scalar Multiplication	18.393814	0.505108	17.206042	24.52975
T_{pai}^{enc}: Paillier Homomorphic Encryption	3.200536	0.026435	3.18125	3.358042
T_{ecc}^{sig}: ECC signature generation	0.035303	0.001891	0.034417	0.05275
T_{ecc}^{ver}: ECC signature verification	0.082326	0.002493	0.081042	0.099375
T_{mlkem}^{KGen}: ML-KEM-768 KeyGen	0.041838	0.001144	0.040833	0.057209
T_{mlkem}^{Encap}: ML-KEM-768 Encapsulation	0.045178	0.000965	0.04475	0.054084
T_{mlkem}^{Decap}: ML-KEM-768 Decapsulation	0.056954	0.001036	0.056416	0.070584

Table 4 gives the estimated computational complexity, communication overhead, and authentication latency of the proposed protocol for each operational phase. During registration, the TA computes the entity's credentials and securely delivers them using ML-KEM and AEAD security mechanisms, resulting in a one-time latency of approximately 0.175 ms and a communication overhead of 2456 bytes for registering a single entity. The session key establishment phase (in V2V, V2R, and R2R operations) is the most efficient, requiring only one Blom polynomial evaluation and lightweight hash and XOR operations at each vehicle, thereby achieving a latency of approximately 0.05 ms with only 184 bytes exchanged per authentication. In contrast, the domain key sharing in V2R authentication incurs higher latency of approximately 0.205 ms and communication overhead of 2448 bytes mainly due to the ML-KEM high footprint, which achieves a quantum-secure key sharing at the expense of communication overhead. However, the domain key sharing is less frequent compared to the normal V2V, V2R, and R2R communications, which can be achieved without high network congestion impact. The periodic security update phase is similar to the registration mechanism incurring a latency of 0.161 ms but with reduced communication cost to 1224 bytes depending on the credentials needed for renewal, wich has minimal impact on routine communications. Finally, the revocation phase introduces only a small communication overhead of 72 bytes to forward a malicious message, while its computational cost mainly depends on the investigation time at the TA end. Overall, the proposed design achieves an effective balance between post-quantum security, low authentication latency, and communication efficiency by reserving ML-KEM operations for infrequent management procedures and relying primarily on Blom's key pre-distribution and hash computations during regular authentication.

Table 4: Estimated computational and communication costs of protocol phases.

Protocol Phases	TA	Sender (V_i)	Receiver (V_j/R_k)	Latency (ms)	Overhead (byte)
Registration	$T_{blom} + T_{mlkem}^{Encap} + 2T_{sha} + T_{E/D}$	$T_{mlkem}^{KGen} + 1T_{mlkem}^{Decap} + T_{sha} + T_{E/D}$	–	$0.175 \times N$	$2456 \times N$
Session Key Establishment	–	$T_{blom} + 7T_{sha} + 2T_{xor}$	$T_{blom} + 7T_{sha} + 2T_{xor}$	$0.05 \times N$	$184 \times N$
V2R Domain Key Sharing	–	$T_{mlkem}^{KGen} + T_{blom} + T_{mlkem}^{Decap} + T_{E/D} + 5T_{sha} + 2T_{xor}$	$T_{blom} + T_{mlkem}^{Encap} + T_{E/D} + 6T_{sha} + 2T_{xor}$	$0.205 \times N$	$2448 \times N$

(Continued)

Table 4 (continued)

Protocol Phases	TA	Sender (V_i)	Receiver (V_j/R_k)	Latency (ms)	Overhead (byte)
Security Update	$T_{\text{mlkem}}^{\text{Encap}} + T_{\text{sha}} + T_{E/D}$	$T_{\text{mlkem}}^{\text{KGen}} + T_{\text{mlkem}}^{\text{Decap}} + T_{E/D}$	–	$0.161 \times N$	$1224 \times N$
Revocation	Variable	Forwarding	–	Variable	$72 \times N$

6.2 Comparative Computational Scalability

Based on the estimated computational costs in Table 5, the proposed protocol (denoted by X) is compared with each related scheme (denoted by Y) using the percentage improvement $100(Y - X)/Y$. As illustrated in Fig. 7a, the proposed protocol reduces the V2V computation time by 99.95% and 99.93% compared with the methods in [13] and [21], respectively, while achieving a 99.93% reduction in R2R computation compared with [13]. For V2R authentication, the proposed protocol achieves computation-time reductions of 99.72%, 99.90%, and 99.86% compared with [13,14], and [21], respectively. However, the methods in [26,29] achieve 59.02% and 69.75% lower computation in V2R mainly because of the ML-KEM integration in the proposed protocol. The computational cost was not evaluated for the protocol in [22] because its implementation relies on custom lattice-based operation that were not measured in our benchmarking. The significant reduction in computational cost in our protocol is achieved by confining the relatively expensive ML-KEM operations to registration, security updates, and V2R domain-key sharing, while frequent authentication relies primarily on lightweight Blom polynomial evaluations and SHA-256 computations. Consequently, the proposed protocol can theoretically support approximately 20,000 V2V and 4878 V2R authentications per second, demonstrating excellent scalability for dense vehicular environments, as shown in Fig. 7b, while improving resilience against network flooding and resource-exhaustion attacks.

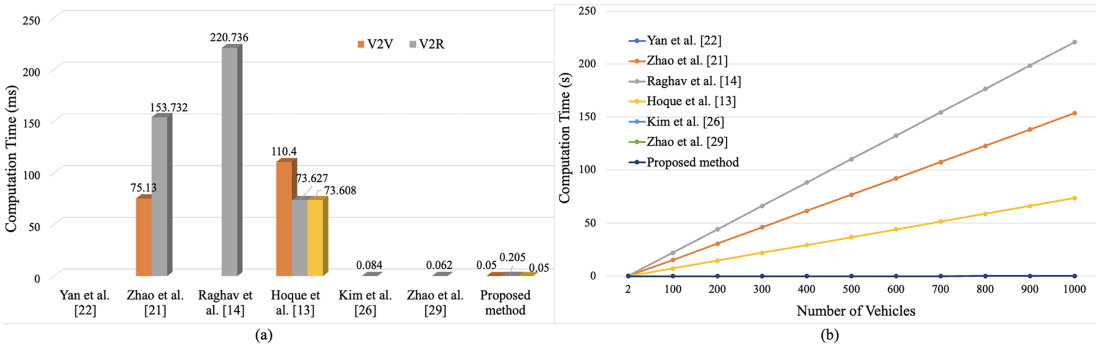

Figure 7: Computation cost: (**a**) comparative overhead and (**b**) comparative scalability.

Table 5: Performance comparison of different methods.

Methods	Computation Time (ms)			Communication Cost (bytes)		
	V2V	V2R	R2R	V2V	V2R	R2R
[13]	$6T_{\text{ecc}}^{\text{sm}} + 2T_{E/D} + 10T_{\text{sha}} + 4T_{\text{xor}} = 110.4$	$4T_{\text{ecc}}^{\text{sm}} + 2T_{E/D} + 14T_{\text{sha}} + 8T_{\text{xor}} = 73.627$	$4T_{\text{ecc}}^{\text{sm}} + 14T_{\text{sha}} + 6T_{\text{xor}} = 73.608$	201	240	204

(Continued)

Table 5 (continued)

Methods	Computation Time (ms)			Communication Cost (bytes)		
	V2V	V2R	R2R	V2V	V2R	R2R
[14]	N/A	$12T_{ecc}^{sm} + 4T_{sha} + 2T_{xor} = 220.736$	N/A	N/A	188	N/A
[21]	$4T_{ecc}^{sm} + 2T_{mod}^{mul} + 8T_{mod}^{exp} + T_{ecc}^{sig} + T_{ecc}^{ver} + 4T_{E/D} = 75.13$	$8T_{ecc}^{sm} + 2T_{mod}^{mul} + T_{mod}^{exp} + 2T_{pai}^{enc} + 2T_{xor} = 153.732$	N/A	1424	484	N/A
[22]	N/A	–	N/A	N/A	1695	N/A
[26]	N/A	$28T_{sha} + 23T_{xor} = 0.084$	N/A	N/A	288	N/A
[29]	N/A	$13T_{sha} + T_{puf} + T_{fe} + 6T_{prf} + 3T_{xor} = 0.062$	N/A	N/A	264	N/A
Proposed	$2T_{blom} + 14T_{sha} + 4T_{xor} = 0.05$	$T_{mlkem}^{KGen} + 2T_{blom} + T_{mlkem}^{Encap} + 2T_{E/D} + T_{mlkem}^{Decap} + 11T_{sha} + 4T_{xor} = 0.205$	$2T_{blom} + 14T_{sha} + 4T_{xor} = 0.05$	184	2448	184

6.3 Comparative Communication Scalability

The communication cost is evaluated based on the total size of the exchanged authentication messages. During mutual authentication and session key establishment (V2V, V2R, and, R2R), the protocol exchanges three messages containing temporary identities (32 bytes), timestamps (8 bytes), and authentication tokens (32 bytes), resulting in a total communication overhead of 184 bytes. In contrast, the V2R domain key sharing phase additionally carries the ML-KEM public key (1184 bytes), the ML-KEM encapsulated ciphertext (1088 bytes), and the AEAD encrypted domain key (32 bytes), increasing the communication overhead to 2448 bytes. Table 5 compares the communication costs of different protocols. The proposed protocol reduces the V2V communication overhead by 8.46% and 87.08% compared with [13] and [21], respectively as visible in Fig. 8a, while achieving a 9.80% reduction in R2R compared with [13]. However, the V2R communication overhead is higher than that of existing classical schemes because of the transmission of ML-KEM public keys and ciphertexts required to provide quantum-resistant key sharing. This additional overhead represents a deliberate trade-off for post-quantum security and occurs only during infrastructure-assisted authentication, whereas the more frequent mutual authentication and session key establishment phases remain lightweight. Furthermore, the decentralized authentication architecture eliminates the need for blockchain interactions or frequent TA involvement, reducing the number of exchanged messages and allowing most cryptographic operations to be performed locally. Consequently, despite the increased V2R message size, the proposed protocol maintains low authentication latency while providing scalable and quantum-secure authentication for dense vehicular networks as shown in Fig. 8b.

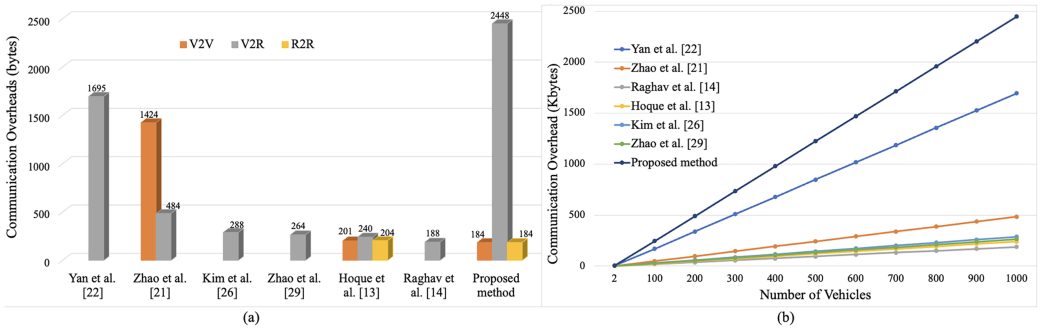

Figure 8: Communication cost: (**a**) comparative overhead and (**b**) comparative scalability.

7 Network-Scale Simulation

7.1 Simulation Environment

The proposed protocol is evaluated using the NS-3 network simulator on macOS, with NetAnim employed for network visualization. The simulation models a dynamic V2X environment consisting of four static RSUs spaced 500 m apart and $N \in [2, 1000]$ vehicles randomly distributed over a 1 km^2 area, moving at speeds between 5 and 25 m/s. Vehicles and RSUs perform frequent mutual authentication and session key establishment over Dedicated Short-Range Communication (DSRC) protocol (IEEE 802.11bd [35]) with a communication range of 500 m, capturing the dynamic formation and disruption of V2V and V2R links as vehicles move through RSU coverage areas. The simulation parameters are summarized in Table 6.

Table 6: Simulation parameters.

Parameters	Setup
Platform	Mac Os (Chip: M2, RAM: 16 GB)
Tool used	NS-3.47 and NetAnim
Implementation Language	C++
Simulation Area	1000 m × 1000 m
Number of Vehicles	2 to 1000 moving at 5 to 25 m/s
Number of RSUs	4 (500 m distance)
Maximum communication range	500 m for both vehicles and RSUs
Wireless protocol	DSRC (802.11bd) with a max of 2304 bytes per packet frame
Message Freshness Interval	ΔT = 100 ms
Simulation Run Time	100 s

7.2 Simulation Metrics

The proposed protocol is evaluated using two performance metrics: the packet delivery ratio (PDR) and the end-to-end authentication latency. The PDR, defined in Eq. (5), measures the percentage of successfully received packets (N_{rx}) relative to the total transmitted packets (N_{tx}). The end-to-end authentication latency, given in Eq. (6), denotes the elapsed time from transmitting an authentication request to the successful establishment of a session key. It comprises the cryptographic processing delay (T_{proc}) and the network transmission delays for the request (T_{tx}) and response (T_{rx}). The processing delay incorporates the benchmarked execution times of SHA-256, XOR, Blom's polynomial evaluation, and ML-KEM operations, which are injected into the NS-3 event scheduler to emulate realistic cryptographic processing, while the

transmission delays are directly measured from the simulated wireless channel. Together, these metrics assess the protocol's reliability and real-time performance under varying vehicular network conditions.

$$PDR = (N_{rx}/N_{tx}) \times 100 \tag{5}$$

$$L = T_{tx} + T_{proc} + T_{rx} \tag{6}$$

7.3 Simulation Results

We first simulate the network by randomly distributing vehicles within a 1000 m × 1000 m free space and observed that the PDR score can degrade even in low-density conditions when some senders are placed outside of the receiver's communication range as illustrated in Fig. 9a,b. Therefore, we reconfigured vehicles distribution such that senders are initially placed within the receiver's communication range as illustrated in Fig. 9c, and observed that the PDR score became stable throughout topology variations. The impact of density and speed variations on the PDR and Latency are discussed below:

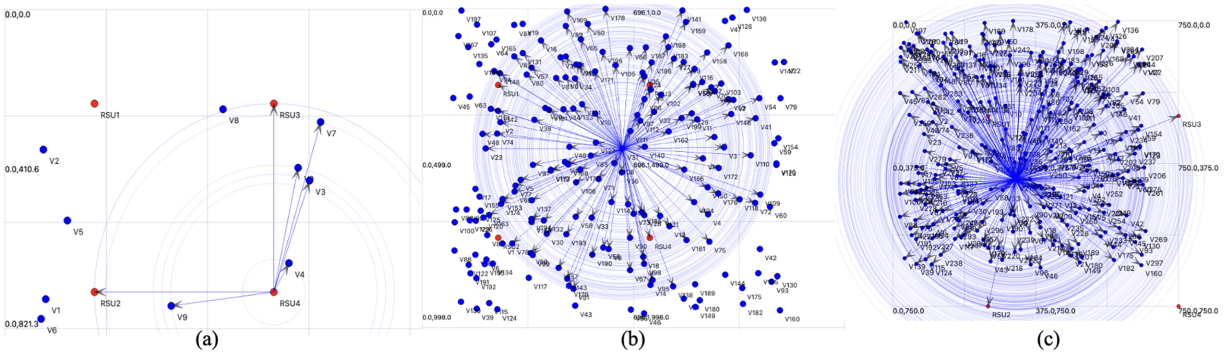

Figure 9: Visualization of the vehicular network simulation in the NetAnim environment: (**a**) low density with out-coverage senders, (**b**) high density with out-coverage senders, and (**c**) high density with full-coverage senders.

7.3.1 Impact of Vehicle Density Variation

This scenario tests the scalability constraints on a single vehicle and a single RSU when receiving authentication requests from multiple neighboring vehicles. Therefore, we maintain the vehicle speed to 20 m/s and vary the vehicle density from 2 to 1000, while initially distributing them within the maximum 500 m range. Fig. 10a shows that the proposed V2V authentication consistently achieves a PDR of nearly 100% across all network densities, demonstrating reliable message delivery between neighboring vehicles even under high densities because they move at the same speed and remain in the communication range of one another throughout the simulation. In contrast, the V2R PDR gradually decreases from 100% in sparse networks to approximately 67% at 1000 vehicles. This degradation can be expected because vehicles are moving away from static RSUs and may lose connectivity beyond the allowable range. Moreover, the increasing communication contention from simultaneous connected vehicles can also create channel congestion reduce communication performance. Nevertheless, the V2R communication maintains a satisfactory delivery performance even under highly dense traffic conditions. Fig. 10b illustrates the authentication latency under different vehicle densities. As expected, the authentication delay increases with the number of vehicles because of higher channel contention, packet queuing, and processing overhead. The V2V latency rises from approximately 0.5 ms with two vehicles to about 84 ms at 1000 vehicles, while the V2R latency increases from about 1.5 to 87 ms over the same range. Although authentication latency grows with network density due to long message queues, both communication modes remain within practical limits for vehicular authentication. Furthermore, the consistently higher PDR achieved by V2V and the moderate latency increase demonstrate

that the proposed protocol scales efficiently with increasing network density while preserving reliable and timely authentication.

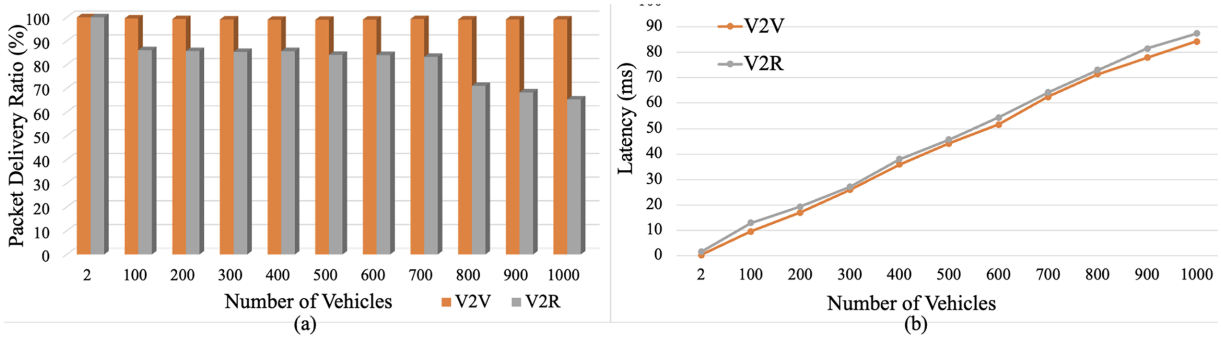

Figure 10: Density variations: (**a**) packet delivery ratio and (**b**) average end-to-end latency.

7.3.2 Impact of Vehicle Speed Variation

Unlike Scenario 1, this scenario fixes the network topology to two vehicles and one RSU while varying the vehicle speed from 5 to 25 m/s to evaluate the impact of mobility on the proposed authentication protocol. Fig. 11a shows that the V2V communication consistently achieves a PDR of nearly 100% across the entire speed variation, indicating that the proposed authentication protocol maintains reliable direct communications despite increasing vehicle mobility. Since vehicles are moving at the same speed at a time, they remain in the communication range of one another throughout the simulation duration. In contrast, the V2R PDR experiences a slight decrease from nearly 100% at 5 m/s to approximately 91% at 25 m/s. This reduction is primarily attributed to the lack of connectivity as RSUs are fixed while vehicles are moving and may leave the RSU communication coverage before the simulation ends. Nevertheless, the obtained PDR demonstrates that the protocol maintains high communication reliability even under high-mobility conditions. The authentication latency results in Fig. 11b further demonstrate the protocol's robustness against vehicle mobility. As the vehicle speed increases, the authentication delay grows only slightly, with the V2V latency increasing from approximately 0.22 to 0.34 ms and the V2R latency from about 1.05 to 1.6 ms. The consistently lower V2V latency results from the absence of infrastructure involvement, whereas V2R authentication incurs additional communication and processing overhead at the RSU. Overall, the modest latency increase confirms that the proposed protocol maintains efficient real-time authentication performance even in highly dynamic vehicular environments.

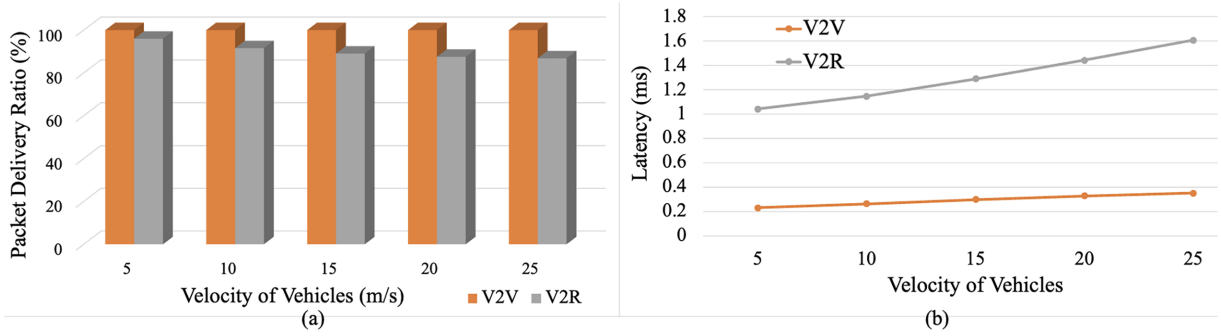

Figure 11: Speed variations: (**a**) packet delivery ratio and (**b**) average end-to-end latency.

8 Conclusion

Vehicular networks are a fundamental component of intelligent transportation systems, enabling secure and reliable communications for future smart cities. This paper presented a lightweight post-quantum authentication and key agreement protocol that combines Blom's key pre-distribution scheme with ML-KEM to achieve efficient and quantum-resistant authentication across V2V, V2R, and R2R communications. By confining computationally intensive ML-KEM operations to registration and infrastructure-assisted key distribution while relying on lightweight Blom polynomial evaluations and SHA-256 hashing during routine authentication, the proposed protocol effectively balances security and efficiency. Security was validated through informal analysis, BAN logic, and AVISPA verification, demonstrating resistance against impersonation, replay, node capture, session key compromise, storage intrusion, and quantum attacks while preserving anonymity, unlinkability, and conditional traceability. Performance evaluation showed substantial reductions in computational cost up to 99.95% and communication overhead (8.46%–87.08%) compared with existing schemes. Although V2R communication incurs additional overhead due to ML-KEM ciphertext transmission, this trade-off enables post-quantum secure key establishment while maintaining low authentication latency. Furthermore, NS-3 simulations confirmed that the proposed protocol provides reliable packet delivery and low end-to-end authentication latency under varying vehicle densities and mobility conditions. Overall, the proposed protocol offers a scalable, secure, and practical post-quantum authentication framework for next-generation vehicular networks and other resource-constrained cyber-physical systems. Future work will investigate broadcast basic safety message authentication protocol using distributed RSU domain keys and implementation on real vehicular hardware platforms for more realistic deployments.

Acknowledgement: The authors acknowledge the support from the PASET-Regional Scholarship and Innovation Funds through the African Center of Excellence in Internet of Things (ACEIoT), the Google Africa PhD Fellowship, and the Carnegie Corporation of New York.

Funding Statement: The APC was funded by Carnegie Corporation of New York through the DOCTAS II Mobility Grant Award-Publication Support.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, methodology, and resources: Lasseni Coulibaly and Damien Hanyurwimfura; investigation, software, and formal analysis, Lasseni Coulibaly; writing—original draft preparation, Lasseni Coulibaly; writing—review and editing and validation: Damien Hanyurwimfura, Evariste Twahirwa, Abubakar Diwani and Lasseni Coulibaly; supervision: Damien Hanyurwimfura, Evariste Twahirwa and Abubakar Diwani; project administration: Damien Hanyurwimfura; funding acquisition: Lasseni Coulibaly and Damien Hanyurwimfura. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available upon request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Muthuramalingam S, Bharathi A, Kumar SR, Gayathri N, Sathiyaraj R, Balamurugan B. IoT based intelligent transportation system (IoT-ITS) for global perspective: a case study. In: Balas VE, Solanki VK, Kumar R, Khari M, editors. Internet of Things and Big Data Analytics for Smart Generation. Cham, Switzerland: Springer; 2018. p. 279–300. doi:10.1007/978-3-030-04203-5_13.
2. Coulibaly L, Hanyurwimfura D, Twahirwa E, Diwani A. A review of cybersecurity challenges and solutions for autonomous vehicles. Int J Adv Comput Sci Appl. 2025;16(2):850–66. doi:10.14569/ijacsa.2025.0160285.

3. Muhammad M, Ali Safdar G. 5G-based V2V broadcast communications: a security perspective. Array. 2021;11(6):100084. doi:10.1016/j.array.2021.100084.
4. Lai J, Zhang X, Liu S, Zhong S, Moshayedi AJ. Blockchain-based VANET edge computing-assisted cross-vehicle enterprise authentication scheme. Comput Commun. 2025;231(6):108040. doi:10.1016/j.comcom.2024.108040.
5. Zhang H, Zhao F. Cross-domain identity authentication scheme based on blockchain and PKI system. High Confid Comput. 2023;3(1):100096. doi:10.1016/j.hcc.2022.100096.
6. Moody D, Robinson A. Cryptographic standards in the post-quantum era. IEEE Secur Privacy. 2022;20(6):66–72. doi:10.1109/msec.2022.3202589.
7. Moody D, Perlner R, Regenscheid A, Robinson A, Cooper D. Transition to post-quantum cryptography standards. Gaithersburg, MD, USA: National Institute of Standards and Technology (NIST); 2024. NIST IR 8547 (IPD). doi:10.6028/NIST.IR.8547.ipd.
8. Antony SNFMA, Bahari MFA. Implementation of elliptic curves in the polynomial Blom key pre-distribution scheme for wireless sensor networks and distributed ledger technology. J Sens Actuator Netw. 2023;12(1):15. doi:10.3390/jsan12010015.
9. NIST.FIPS.203. Module-Lattice-Based Key-Encapsulation Mechanism Standard. Washington, DC, USA: National Institute of Standards and Technology; 2024. doi:10.6028/NIST.FIPS.203.
10. Liu Z, Yao N, Bai S, Mai T. A cooperative ECC-based authentication protocol for VANETs. Sci Rep. 2025;15(1):40837. doi:10.1038/s41598-025-24663-8.
11. Mei Q, Xiong H, Chen J, Yang M, Kumari S, Khan MK. Efficient certificateless aggregate signature with conditional privacy preservation in IoV. IEEE Syst J. 2021;15(1):245–56. doi:10.1109/jsyst.2020.2966526.
12. Jiang H, Hua L, Wahab L. SAES: a self-checking authentication scheme with higher efficiency and security for VANET. Peer Peer Netw Appl. 2021;14(2):528–40. doi:10.1007/s12083-020-00997-0.
13. Hoque A, Amin R, Ali Khan D. L-VAKMC: lightweight authentication and dynamic key agreement for VANET multi-entity communications. IEEE Trans Depend Secur Comput. 2026;23(2):4310–22. doi:10.1109/tdsc.2025.3645431.
14. Raghav, Maurya C, Verma S. Certificateless federated authentication and key agreement in IoV. Comput Netw. 2026;277(4):112017. doi:10.1016/j.comnet.2026.112017.
15. Yang Q, Zhu X, Wang X, Fu J, Zheng J, Liu Y. A novel authentication and key agreement scheme for internet of vehicles. Fut Gener Comput Syst. 2023;145(8):415–28. doi:10.1016/j.future.2023.03.037.
16. Jayashree S, Santhosh Kumar SVN. LAPEP—lightweight authentication protocol with enhanced privacy for effective secured communication in vehicular ad-hoc network. Wirel Netw. 2024;30(1):151–78. doi:10.1007/s11276-023-03459-6.
17. Dwivedi SK, Amin R, Vollala S, Das AK. Design of blockchain and ECC-based robust and efficient batch authentication protocol for vehicular ad-hoc networks. IEEE Trans Intell Transp Syst. 2024;25(1):275–88. doi:10.1109/tits.2023.3310514.
18. Lin HT, Jhuang WL. Blockchain-based lightweight certificateless authenticated key agreement protocol for V2V communications in IoV. IEEE Internet Things J. 2024;11(16):27744–59. doi:10.1109/jiot.2024.3400320.
19. Noh J, Kwon Y, Son J, Cho S. Blockchain-based one-time authentication for secure V2X communication against insiders and authority compromise attacks. IEEE Internet Things J. 2023;10(7):6235–48. doi:10.1109/jiot.2022.3224465.
20. Zhao F, Ding H, Li C, Su Z, Liang G, Yang C. A blockchain-based efficient cross-domain authentication scheme for Internet of vehicles. Comput Mater Contin. 2024;80(1):567–85. doi:10.32604/cmc.2024.052233.
21. Zhao J, Guo Y, Liao L, Wang D. A blockchain-based efficient traceability authentication scheme in VANET. Digit Commun Netw. 2025;11(5):1410–20. doi:10.1016/j.dcan.2025.04.013.
22. Yan M, Bao Z, Li J, Tian H. Quantum-resistant blockchain-assisted certificateless authentication and three-party key agreement scheme for VANET. Expert Syst Appl. 2026;299(5):130163. doi:10.1016/j.eswa.2025.130163.
23. Lu Z, Wang Q, Qu G, Zhang H, Liu Z. A blockchain-based privacy-preserving authentication scheme for VANETs. IEEE Trans VLSI Syst. 2019;27(12):2792–801. doi:10.1109/tvlsi.2019.2929420.

24. Wazid M, Das AK, Kumar N, Odelu V, Goutham Reddy A, Park K, et al. Design of lightweight authentication and key agreement protocol for vehicular ad hoc networks. IEEE Access. 2017;5:14966–80. doi:10.1109/access.2017.2723265.
25. Aman MN, Javaid U, Sikdar B. A privacy-preserving and scalable authentication protocol for the Internet of vehicles. IEEE Internet Things J. 2021;8(2):1123–39. doi:10.1109/jiot.2020.3010893.
26. Kim C, Kwon D, Son S, Yu S, Park Y. An anonymous and efficient authentication scheme with conditional privacy preservation in Internet of vehicles networks. Mathematics. 2024;12(23):3756. doi:10.3390/math12233756.
27. Vasudev H, Das D. An efficient authentication and secure vehicle-to-vehicle communications in an IoV. In: Proceedings of the 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring); 2019 Apr 28–May 1; Kuala Lumpur, Malaysia. p. 1–5. doi:10.1109/vtcspring.2019.8746612.
28. Nandy T, Idris MYI, Noor RM, Das AK, Li X, Ghani NA, et al. An enhanced lightweight and secured authentication protocol for vehicular ad-hoc network. Comput Commun. 2021;177:57–76. doi:10.1016/j.comcom.2021.06.013.
29. Zhao Y, Wu L, Wang J, Li W, Jin J, Zhang Z, et al. LDSAV: lightweight and dynamically scalable authentication key agreement for vehicle-cloud collaboration. IEEE Trans Veh Technol. 2026;1–15. doi:10.1109/tvt.2026.3669949.
30. Li X, Liu T, Obaidat MS, Wu F, Vijayakumar P, Kumar N. A lightweight privacy-preserving authentication protocol for VANETs. IEEE Syst J. 2020;14(3):3547–57. doi:10.1109/jsyst.2020.2991168.
31. Coulibaly L, Hanyurwimfura D, Twahirwa E, Diwani A. A lightweight biometric-based user authentication protocol for self-driving vehicles. In: Proceedings of the 2024 1st International Conference on Cyber Security and Computing (CyberComp); 2024 Nov 6–7; Melaka, Malaysia. p. 144–9. doi:10.1109/cybercomp60759.2024.10913668.
32. Burrows M, Abadi M, Needham R. A logic of authentication. ACM Trans Comput Syst. 1990;8(1):18–36. doi:10.1145/77648.77649.
33. Viganò L. Automated security protocol analysis with the AVISPA tool. Electron Notes Theor Comput Sci. 2006;155(1):61–86. doi:10.1016/j.entcs.2005.11.052.
34. Backbone Authors. Pqcrypto: post-quantum cryptography for python. 2026 [cited 2026 May 1]. Available from: https://pypi.org/project/pqcrypto/.
35. IEEE Std 80211bd-2022. IEEE standard for information technology, specific requirements part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: enhancements for next generation V2X. Piscataway, NJ, USA: IEEE; 2023. p. 1–144. doi:10.1109/IEEESTD.2023.10063942.