Article

A New Key Agreement Mechanism Based on Lattice

Yuxia Qian , Yiwen Liang , Lei Shang , Xinqi Dong , Yincheng Liang *

Posted Date: 15 April 2026

doi: 10.20944/preprints202604.1123.v1

Keywords: lattice-based cryptography; post-quantum key agreement; identity-based encryption; implicit authentication; version control

 doi:10.20944/preprints202604.1123.v1

Article

A New Key Agreement Mechanism Based on Lattice

Yuxia Qian [1], Yiwen Liang [1], Lei Shang [2], Xinqi Dong [3] and Yincheng Liang [1,3,*]

[1] Department of Information Engineering, Shandong Water Conservancy Vocational College, Shandong Province, China

[2] School of Cyberspace Security, Shandong University of political Science and Law, Shandong Province, China

[3] School of Computer Science, Qufu Normal University, Shandong Province, China

* Correspondence: liangyincheng111@163.com

Abstract

Network access control and identity legitimacy verification have been implemented by establishing a secure foundation for the trusted establishment of communication entities. However, successful identity authentication alone does not guarantee secure communication. In open-network environments, it remains essential to establish a secure session key via a robust key agreement mechanism—one that prevents explicit disclosure of identity information while ensuring post-quantum security. To address these requirements, we propose a lattice-based key agreement protocol. The protocol integrates identity binding, implicit authentication, and session key establishment into a single ciphertext exchange. Furthermore, it supports secure key evolution and revocation verification through a version-control mechanism and a blockchain-maintained revocation list—thus realizing a comprehensive, post-quantum-secure key agreement scheme under reasonable computational and communication overhead.

Keywords: lattice-based cryptography; post-quantum key agreement; identity-based encryption; implicit authentication; version control

1. Introduction

With the rapid proliferation of open network paradigms—such as the Internet of Things (IoT) and the Internet of Vehicles (IoV)—terminal security communication imposes increasingly stringent requirements on out-of-the-box key agreement protocols. Such protocols must not only guarantee the confidentiality of session keys and the trusted binding between negotiating parties but also exhibit robustness against active adversaries [1–3]. Moreover, the advent of quantum computing poses a fundamental threat to classical public-key cryptosystems rooted in the hardness of discrete logarithms and large-integer factorization [4–6]; consequently, post-quantum cryptography (PQC) has become an indispensable foundation for secure protocol design [7,8].

Although existing key agreement frameworks offer distinct advantages, they face inherent structural limitations when deployed in open-network environments: certificate-based or explicitly signed schemes introduce substantial maintenance overhead and interactive complexity due to authentication chain management; certificateless and identity-based approaches often necessitate explicit identity disclosure or reliance on a centralized authority. Furthermore, revocation and rekeying mechanisms remain largely ad hoc and poorly integrated into the core protocol logic, thereby introducing unnecessary latency and operational complexity [9–12]. At its core, this challenge stems from the difficulty of simultaneously embedding multiple cryptographic semantics—including identity binding, implicit authentication, session key establishment, and revocation-aware key updates—within a single, lightweight negotiation round, while satisfying both post-quantum security guarantees and minimal interaction requirements [13]. Thus, designing a post-quantum key agreement mechanism that harmonizes strong security properties with practical deployability has emerged as a critical and urgent research problem [14–16].

 doi:10.20944/preprints202604.1123.v1

To address the aforementioned challenges, this paper proposes a key agreement protocol built upon a post-quantum identity-based matching encryption framework. The core idea is to tightly bind session key derivation material to the peer's identity and integrate ciphertext exchange directly into the protocol flow—thereby enabling the receiver to simultaneously achieve identity binding and implicit authentication during decryption. Furthermore, a two-way confirmation mechanism is employed to mitigate the risk of unknown key-share attacks. By virtue of this design, the protocol unifies three critical functionalities—post-quantum-secure identity binding, implicit authentication, and update/revocation management—into a single, coherent negotiation process, thereby striking an effective balance between security guarantees and practical deployability. At the theoretical level, the protocol provides a unified security model capturing multiple security goals—including forward secrecy, mutual authentication, and resistance to unknown key-share attacks—under standard post-quantum assumptions. At the application level, its computational cost is dominated by parallelizable linear-algebraic operations, making it well-suited for a broad range of general-purpose deployment scenarios [17,18].

The main contributions of this chapter are as follows:

- **Identity-binding implicit authentication negotiation mechanism.** We propose a key agreement protocol grounded in a post-quantum identity-matching encryption framework. This design tightly couples session key establishment with peer identity binding within a single ciphertext exchange, thereby eliminating the need for separate authentication steps. Crucially, a mutual confirmation mechanism mitigates risks associated with negotiation deviations—such as unknown key-share attacks—by ensuring both parties cryptographically verify each other's identities and contributions to the shared key.
- **Version control update and blockchain revocation mechanism.** We introduce a version-driven key evolution strategy that synchronizes cryptographic material updates across distributed endpoints without requiring additional protocol rounds. Furthermore, we design a decentralized revocation infrastructure wherein a tamper-evident revocation list is maintained on a permissioned blockchain. This enables transparent, auditable, and trust-minimized verification of key status, supporting secure and scalable key update and revocation management.
- **Security and performance demonstration.** Under a rigorously defined security model—including formal treatment of forward secrecy, mutual authentication, and resistance to post-quantum adversaries—we provide a comprehensive security proof. Performance is quantitatively assessed in terms of computational latency and communication bandwidth. Experimental results show an average computational overhead of less than 100μs per protocol execution and a total communication cost of approximately 17.1 KB—demonstrating that the protocol achieves full security functionality with only moderate, practically viable resource consumption.

2. System Model and Security Objectives

2.1. System Model

This section examines a generic two-party key agreement protocol involving a protocol initiator, denoted as party A, and a responder, denoted as party B. The two parties exchange messages over a public channel and—despite operating independently—ultimately derive an identical session key K. The public channel is assumed to be fully adversarial: all transmitted messages may be subject to eavesdropping, replay, tampering, arbitrary delay, or injection of forged messages [19]. The underlying system model is illustrated in Figure 1.

To achieve key agreement without the constraints of a traditional certificate system while maintaining identity binding, the system also introduces a Key Generation Center (KGC) as an entity for initialization and key derivation [20–24]. The KGC's responsibilities are limited to performing system initialization, publishing the system's public parameters mpk, and holding the master secret

 doi:10.20944/preprints202604.1123.v1

key *msk*. It also distributes identity-related key material based on *msk* for user identities, which is only held by the KGC [12]. The KGC does not participate in the online interaction process between the two parties, does not come into contact with session keys, and does not need to be available online during the session.

Each user holds two types of key materials in the system:

- System public parameters, which are the public keys *mpk* generated by the KGC.
- Identity-related key material (ek_{id}, dk_{id}), derived by the KGC according to the user's identity and (*mpk*,*msk*), for use by the user in the session.

During the session execution, both parties will also sample temporary random quantities and derive the final session key by *KDF* derivation. Temporary quantities are used to ensure forward security and session uniqueness. In order to support natural key rotation and state synchronization, this protocol introduced version vector $V \in Z_q^n$, and injected version vector and identity into the underlying lattice structure, so that the version information really participated in the matrix operation rather than staying in the additional field of the protocol layer.

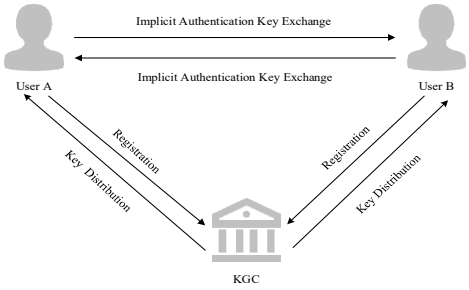

Figure 1. System model.

2.2. Security Objectives

Under the aforementioned system model and the potential threats to the network, the key agreement protocol proposed in this section must satisfy the following security requirements:

- **Session key confidentiality.** For any *PPT* adversary *Adv*, the advantage of distinguishing a real session key from a random string on a fresh session is negligible.
- **Implicit authentication.** If one party outputs a session key *K*, only entities that match the identity tag of the peer and hold the corresponding key material may output the same one *K*. In other words, key consistency on its own constitutes authentication, without the need for an additional signature or certificate.
- **Forward security.** After the session is finished, an adversary can obtain the key of each participant without recovering the previous session key. This property is guaranteed by the binding of a session temporary random quantity *KDF* to a function.
- **Key Separation and Version Evolution Consistency.** Key materials associated with different protocol versions are strictly isolated from one another in the key agreement process. Consequently, compromise of an older version's keying material must not enable derivation of session keys for newer versions [25]. Furthermore, version identifiers are explicitly incorporated into both the key derivation function and the ciphertext structure, enabling both parties to transition synchronously to a new version within the same time window—without requiring additional protocol interactions.

2.3. Security Model

Security proof parties include challenger *C* and adversary $\mathcal{Adv}$. The challenger *C* simulates system initialization, key generation and protocol execution, maintains the system master key *msk* and provides public parameters *mpk* to the adversary [26–28]. The adversary $\mathcal{Adv}$ controls the public channel, and can passively eavesdrop, actively tamper with and replay protocol messages,

 doi:10.20944/preprints202604.1123.v1

and query several keys or session information according to the model. In order to characterize the adversary capability, the oracle is defined as follows:

- **Setup(1^λ):** Challenger C runs system initialization, generates (mpk,msk), returns mpk to Adv, and saves msk to itself. Setup only needs to be used once in the same system.
- **EKGen(id):** Encryption key query, the challenger C gives the id decryption key ek_{id} to Adv.
- **DKGen(id):** Decryption key query, the challenger C gives the id decryption key dk_{id} to Adv.
- **Send(U_i,msg):** In active interactive query, the adversary sends a message msg to user U_i, and the challenger returns the response information as stipulated by the protocol.
- **Execute(A_i,B_j):** Passive eavesdropping query, which returns the complete data of an hone st execution, is equivalent to passive eavesdropping by the adversary.
- **Reveal(U_i):** Returns the session key if the session instance has accepted and output the se ssion key, otherwise returns $\perp$.
- **Test(U_i):** If U_i is fresh, the challenger takes bits randomly $b\in\{0, 1\}$. $b=1$: return the real sess ion key K; b=0: return a uniformly random string of equal length. The adversary outputs a guess, b^* and if $b^*=b$ it does, it wins.

Define a session instance U_i as fresh if and only if:

- Not U_i initiated by a $Reveal(U_i)$ pair;
- Not initiated V_j for its matching session $Reveal(V_j)$;
- The adversary Adv is not allowed to simultaneously obtain all the long-term or tempora ry secrets necessary to lead to the direct computation of the session key.

3. Concrete Schemes

Before giving the calculation process of each stage of the protocol, the main symbols and meanings used in this section should be agreed upon (refer Table 1.) to avoid repeated explanations in different stages. The procedures of initialization, registration, negotiation, and revocation are then described in chronological order.

Table 1. Model dataset.

Symbols	Meaning	Symbol	Meaning
ek_i	The user's encryption	$dk_{i,v}$	The decryption key for the user under the versioniv
V	key i	$H_1()$	$\mathbb{Z}_q^n\rightarrow\mathbb{Z}_q^n$
$H_2()$	Version number	$H_v()$	$\{0,1\}^*\rightarrow\mathbb{Z}_q^n$
	$\mathbb{Z}_q^{2n}\rightarrow\mathbb{Z}_q^{n\times m}$		

3.1. Initialization Phase

The initialization phase is performed by KGC and is used to generate the public parameters as well as the master key (mpk, msk).

- KGC selects parameters (q, m, n) with noise distribution χ, norm bound and function $W(V)$, and selects hash function H_1, H_2, H_v, as well as KDF function (HKDF-SHA-256).
- KGC calls the trapdoor generation function $LTrapGen(q, n)$ to generate (A_0, T_{A0}), (A,T_A), where A_0,$A\in\mathbb{Z}_q^{n\times m}$, T_{A0},$T_A\in\mathbb{Z}_q^{n\times m}$are the corresponding trapdoors.
- KGC randomly samples A_1,$B\in\mathbb{Z}_q^{n\times m}$ and vectors $u_0\in\mathbb{Z}_q^n$.
- Output mpk=(A_0,A_1,A,B,u_0,H_1,H_2,H_v,q,n,m,$W(V)$), msk=(T_{A0},T_A). mpk is the shared public par ameters uploaded to the block chain and only saved by KGC.

 doi:10.20944/preprints202604.1123.v1

3.2. Registration Phase

The registration phase takes place in a secure channel. The registration phase takes place in the secure channel. Taking the user with the identity id_i as an example, it is described as follows.

- The user sends its identity id_i to KGC to initiate a registration request, and KGC verifies the legitimacy $SamplePre(T_A, H_1(id_i))$ of the user's identity and invokes the generation of the user's public key ek_{i_i}.
- Where $v = \lfloor t/\Delta \rfloor$, is the current Unix time (s), Δ is the length of time window (24h), calculate the current version number $V = H_v(v)$, construct $\tau = id_i \| V$ and set clock tolerance $W(V) = \{V-1, V, V+1\}$ to avoid failure caused by clock drift.
- Calculate $F_\tau = [A_0 | A_1 + H_2(\tau) \cdot B] \in \mathbb{Z}_q^{n \times 2m}$
- Generate $dk_{i,v}$ by $SampleLeft(A_0, A_1, T_{A0}, u_0, id_i)$ with trapdoor T_{A0} and make $dk_{i,v}$ satisfies

$$F_\tau \cdot dk_{i,v} \equiv u_0 (mod\ q) \quad (1)$$

- The KGC returns $(ek_i, dk_{i,v})$ to the user through a secure channel, and registers the registration commitment information (such as identity commitment and version commitment) related to the user on the blockchain for subsequent revocation verification.

3.3. Implicit Authentication Negotiation Phase

This phase is conducted in the public channel, and any adversary can monitor, intercept and other attacks on the communication between users [29]. The description of this phase takes users A and B as an example, as shown in Figure 2. In this phase, we assume that both users A and B have completed registration, user A holds (id_A, ek_A, dk_A), user B holds (id_B, ek_B, dk_B), and both parties can query the revocation list maintained by the blockchain.

- **Step 1 (A→B): User A initiates a session with user B and does the following:**

(1) User A computes $v = \lfloor t/\Delta \rfloor$, $V = H_v(v)$. If A queries that the current version of the revocation list exists, the communication is terminated.

(2) A sampling $M_A \leftarrow \{0, 1\}^l$, $s_A \in \mathbb{Z}_q^n$, noise value $x_A \in \chi$, noise vector $y_A \in \chi^m$, random matrix $R_A \in \{-1, 1\}^{m \times m}$, construction $\tau_A = id_B \| V$.

(3) Calculate

$$F_{\tau A} = [A_0 | A_1 + H_2(\tau_A) \cdot B] \quad (2)$$

$$c_{1A} = F_{\tau A}^T \cdot s_A + \begin{bmatrix} y_A \\ R_A^T \cdot y_A \end{bmatrix} \in \mathbb{Z}_q^{2m} \quad (3)$$

(4) Let $\Delta_m = \lfloor q/2^l \rfloor$, calculate

$$c_{0A} = u_0^T \cdot s_A + x_A + \Delta_m \cdot EncInt(M_A) + \| A \cdot ek_A \|_2 \in \mathbb{Z}_q \quad (4)$$

Where $EncInt(M) \in \{0, \ldots, 2^{l-1}\}$ is an encoding that interprets l bits as integers.

(5) Witness $w_A = (id_A, ek_A)$, generate the consistency proof $\pi_A = P(y_A, w_A)$ and $ct_A = (v, c_{0A}, c_{1A}, \pi_A)$.

(6) $msg_1 = ct_A$ will be sent to B.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 April 2026 doi:10.20944/preprints202604.1123.v1

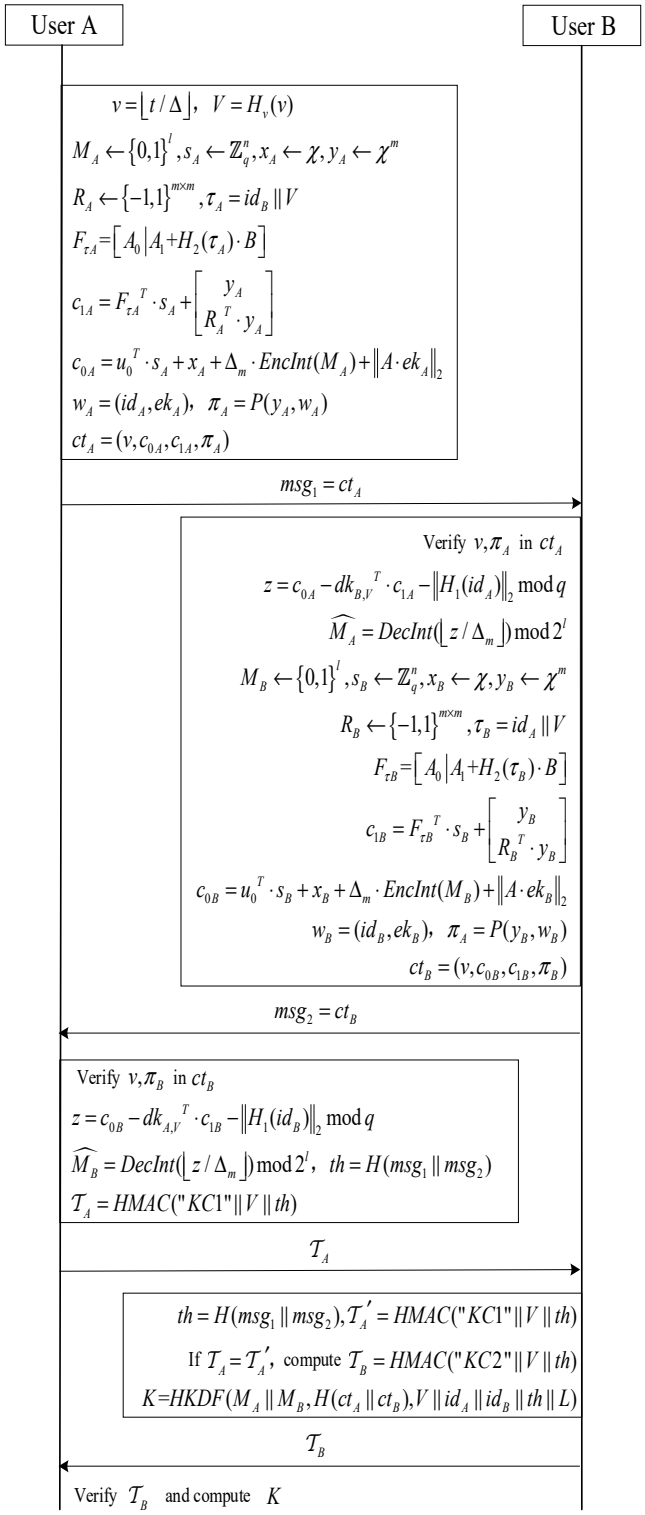

Figure 2. Flow chart of implicit authentication negotiation.

- **Step two (B→A): User B decrypts and responds after receiving the message msg_1.**

(1) Parse $ct_A=(v, c_{0A}, c_{1A}, \pi_A)$, verify v and π_A, terminate if verification fails.

(2) B Select $V' \in W(V)$, first $V'=V$, calculate

$$z = c_{0A} - dk_{B,V}^T \cdot c_{1A} - \|H_1(id_A)\|_2 \bmod q \quad (5)$$

(3) Order $\Delta_m=\lfloor q/2^l \rfloor$, resume $\widehat{M_A}=DecInt(\lfloor z/\Delta_m \rceil) \bmod 2^l$. If $\widehat{M_A}$ so $\perp$, V' replace repeat (2) up to three times and terminate if it still fails.

(4) B sampling $M_B \leftarrow \{0, 1\}^l$, $s_B \in \mathbb{Z}_q^n$, noise value $x_B \in \chi$, noise vector $y_B \in \chi^m$, random matrix $R_B \in \{-1, 1\}^{m\times m}$, construction $\tau_B=id_A \| V$, calculation $F_{\tau B}=[A_0 | A_1+H_2(\tau_B) \cdot B]$.

 doi:10.20944/preprints202604.1123.v1

(5) Computation

$$c_{1B}=F_{\tau B}^{T}\cdot s_B+\begin{bmatrix} y_B \\ R_B^{T}\cdot y_B \end{bmatrix} \quad (5\text{-}6)$$

$$c_{0B}=u_0^{T}\cdot s_B+x_B+\Delta_m\cdot EncInt(M_B)+\|A\cdot ek_B\|_2 \in \mathbb{Z}_q \quad (5\text{-}7)$$

(6) Witness $w_B=(id_B, ek_B)$, generate proof $\pi_B=P(y_B, w_B)ct_B=(v, c_{0B}, c_{1B}, \pi_B)$ of consistency.

(7) $msg_2=ct_B$ will be sent to A.

- **Step 3 (A→B): User A decrypts the message and generates a key confirmation to sen d to B.**

(1) Parse $ct_B=(v, c_{0B}, c_{1B}, \pi_B)$, verify v and π_B, terminate if verification fails.

(2) A Select $V'' \in W(V)$, $V''=V$, calculate

$$z=c_{0B}-dk_{A,V}^{T}\cdot c_{1B}-\|H_1(id_B)\|_2 \bmod q \quad (5\text{-}8)$$

(3) Recovery$\widehat{M_B}=DecInt(\lfloor z/\Delta_m \rceil) \bmod 2^l$.If $\widehat{M_B}$ is $\perp$ replaceable V'' repeat (2), up to three times, still fail to terminate.

(4) Computed $th=H(msg_1||msg_2)$ and $T_A=HMAC("KC1"||V||th)$ will be sent to B.

- **Step 4 (B→A): User B verifies the key confirmation and sends back the confirmation.**

(1) B similarly calculates $th=H(msg_1||msg_2)$, $T_A'=HMAC("KC1"||V||th)$, and verifies T_A' and T_A are equal. If not, terminate. $T_B=HMAC("KC2"||V||th)$ will be sent to A after succes sful verification.

(2) Calculate the session key with the key derivation function

$$K=HKDF(M_A||M_B, H(ct_A||ct_B), V||id_A||id_B||th||L) \quad (9)$$

- **Step 5 (A) : A calculates the session key K after successful verification T_B. After the verification passes, both parties confirm that they share the same session key, and th e session is established.**

3.4. Cancellation Phase

The revocation phase is used to invalidate the user's identity after the specified version in time, so as to prevent the user from continuing to participate in the negotiation when there is a violation or key material disclosure. Once a certain status in version V_r has been revoked, the identity from V_r within (including V_r) in the subsequent session shall be rejected. This phase is illustrated by an example of the revoked user identity id_i.

- KGC retrieves the pertinent registration records from the blockchain using the id_i, appen ds the identified revoked version V_r to the revocation list RL, and then propagates the updated RL to the blockchain—thereby ensuring that all participants can consistently ac cess the most current revocation status.
- Once the revocation record (id_i, V_r) has been committed to the blockchain, any honest user querying this record at version $V \geq V_r$ must refuse to establish a session with entity id_i. If t he peer's identity is discovered to have been revoked during the session negotiation proce ss, the current session must be immediately terminated, and no further message exchange s or key derivations shall proceed.

3.5. Correctness Proof and Noise Bound

Let $z=c_0-dk_V^{T}\cdot c_1-\|H_1(id)\|_2 \ (mod\ q)$, c_0, c_1 will be expanded to be obtained

$$z \equiv \Delta_m\cdot M+\varepsilon \ (mod\ q) \quad (10)$$

Where, the noise direction is

$$\varepsilon=x-\langle dk_v, \begin{bmatrix} y \\ R^{T}\cdot y \end{bmatrix}\rangle+\|A\cdot ek\|_2-\|H_1(id)\|_2 \quad (11)$$

Since decryption relies on rounding $\widehat{M}=\lfloor z/\Delta_m \rceil$, a sufficient condition for correctness is $|\varepsilon|<(\Delta_m/2)$, where $\Delta_m=\lfloor q/2^l \rfloor$. Equivalently, in the modulo q sense, the perturbation introduced into $\Delta_m\cdot M$ must remain strictly within half a quantization step to ensure correct recovery of M.

 doi:10.20944/preprints202604.1123.v1

In order to give operational parameter constraints, $|\varepsilon|$ upper bounds can be estimated for. $|x| \le B_x$ A noise meet, and a short vector boundary, is $\|y\|_\infty \le B_y \|dk_v\|_1 \le B_e$, then

$$\left|\langle dk_v, \begin{bmatrix} y \\ R^T \cdot y \end{bmatrix}\rangle\right| \le \|dk_v\|_1 \cdot \left\|\begin{bmatrix} y \\ R^T \cdot y \end{bmatrix}\right\|_\infty \quad (12)$$

Since $R \in \{-1, 1\}^{m \times m}$, there are $\|R^T \cdot y\|_\infty \le \|y\|_1 \le m\|y\|_\infty \le mB_y$, therefore,

$$\left\|\begin{bmatrix} y \\ R^T \cdot y \end{bmatrix}\right\|_\infty \le mB_y \Rightarrow \left|\langle dk_v, \begin{bmatrix} y \\ R^T \cdot y \end{bmatrix}\rangle\right| \le B_e \cdot mB_y \quad (13)$$

In addition, there are

$$\|A \cdot ek\|_2 - \|H_1(id)\|_2 \le \|A \cdot ek\|_2 + \|H_1(id)\|_2 \le B_{AE} + B_H \quad (14)$$

Where $B_{AE} \ge \|A \cdot ek\|_2$, $B_H \ge \|H_1(id)\|_2$ is the bound that can be given by the parameter and the hash output range. In conclusion

$$|\varepsilon| \le B_x + (B_e m B_y) + (B_{AE} + B_H) \quad (15)$$

So as long as the choice parameters meet

$$B_x + B_e m B_y + B_{AE} + B_H < \frac{1}{2}\lfloor q/2^l \rfloor \quad (16)$$

And can ensure the correct M and ensure the consistency M_A, M_B of the agreement the two sides of consistency with the session key.

4. Security Proof and Analysis

4.1. Security Proof

Theorem 1: Under the random oracle model, the hardness assumption underlying the lattice-based difficulty hypothesis implies that any probabilistic polynomial-time (PPT) adversary's advantage in attacking the protocol proposed in this paper is negligible.

$$Adv(A) \le Adv^{IND}(\mathcal{B}_1) + Adv^{Forge}(\mathcal{B}_2) + negl(\lambda) \quad (17)$$

Among them, $Adv^{IND}(\mathcal{B}_1)$ denotes the adversary's advantage in distinguishing the challenge ciphertext—encrypted under the real session key—from a uniformly random string of the same length. $Adv^{Forge}(\mathcal{B}_2)$ denotes the adversary's advantage in generating a valid ciphertext that decrypts and verifies correctly under the receiver's decryption key, despite being injected adversarially. The following use Game-hopping to prove.

Game G_0: A real-world round-optimal (RoR) game, in which a challenge session is executed on both sides via ciphertext exchange (recall the ciphertexts M_A, M_B). The session key is ultimately derived through mutual agreement, computed as a deterministic function of the concatenation $M_A||M_B$. *Let* $Succ_i$ denote the event that the adversary correctly guesses the test bit in game G_i.

Game G_1: Only the challenge phase is modified in Game G_1. Specifically, the plaintext contribution M_A of user A—embedded in ciphertext ct_A—is replaced with a uniformly random string U_A of identical length; additionally, ct_A must remain a valid encryption of this new plaintext. All other phases are executed identically to the original construction.

Lemma 1 (G_0 and 5.1 G_1 indistinguishable) : existence PPT adversaries $\mathcal{B}_1$, makes

$$|Pr[Succ_0] - Pr[Succ_1]| \le Adv^{IND}(B_1) \quad (18)$$

Proof: $\mathcal{B}_1$ in the face of a hidden challenger C_{IND}. C_{IND} Gives the public parameters and provides a challenge encryption interface, where the inputs are two equal length messages (m_0, m_1) with the target identity (receiver identity). Return the challenge ciphertext ct^* for one of them. In the following, $\mathcal{B}_1$ runs A and simulates G_0/G_1.

Semi-selective Phase: The adversary A declares the identities of the two communicating parties (id_A^*, id_B^*) for the challenge session prior to the Setup phase. $\mathcal{B}_1$ designates id_B^* as the target challenge recipient in this scenario.

 doi:10.20944/preprints202604.1123.v1

Setup: $\mathcal{B}_1$ obtains the public parameters (denoted as mpk) from the challenger C_{IND} and forwards them to A. The mpk is held by the challenger, and $\mathcal{B}_1$ is not required to have knowledge of it.

EKGen Query: When A initiates an $EKGen(id)$ query, $\mathcal{B}_1$ forwards it to the encryption key query oracle of C_{IND}, retrieves ek_{id} and returns it to A.

DKGen Query: When A initiates a $DKGen(id)$ query, if $id=id_B^*$ (a query that would violate the constraints of the challenge game), the query is rejected in accordance with the game rules; otherwise, the oracle of C_{IND}returns dk_{id}.

Challenge Session Embedding: When A drives the protocol to the challenge session via the Send/Execute oracles, and the user pair $A \rightarrow B$ in the challenge session needs to generate ct_A, set $m_0=M_A$ and $m_1=U_A$. $\mathcal{B}_1$ submits two messages of equal length (m_0,m_1) to the challenger C_{IND} and requests a challenge ciphertext. C_{IND} returns ct^*, whose plaintext is either M_A or U_A depending on the hidden bit b. $\mathcal{B}_1$ embeds $ct_A=ct^*$ into the protocol message msg_1 and continues to simulate subsequent protocol messages as specified.

Determination: If A can distinguish between Game G_0 (with the real plaintext M_A and Game G_1 (with the random string U_A, then $\mathcal{B}_1$ uses A's output as its guess for the challenge bit b of the IB-ME privacy challenge, thereby breaking the security of the underlying lattice-based cryptosystem. Q.E.D.

Game G_2: In G_2, building on G_1 a symmetric modification is made: the plaintext M_B of user B is replaced with a random string U_B of the same length, while ensuring ct_B remains a valid encryption output.

Lemma 2 ((Indistinguishability of G_1 and G_2): There exists a probabilistic polynomial-time (PPT) adversary $\mathcal{B}_1^*$ such that

$$|Pr[Succ_1]-Pr[Succ_2]| \leq Adv^{IND}(\mathcal{B}_1^*) \quad (19)$$

The proof of Lemma 2 is symmetric to that of Lemma 1 and is therefore omitted here.

Game G_3: In G_3, both M_A and M_B in the challenge session have been replaced with uniformly random strings. Thus, $M_A \| M_B$ is information-theoretically hidden from the adversary. Since the session key is derived deterministically from $M_A \| M_B$ the adversary's maximum success probability is 1/2, i.e., $Pr[Succ_2]=1/2$.

Summing the inequalities corresponding to the above games yields the advantage bound for Theorem 1, thereby establishing the session key confidentiality.

The proof proceeds as follows.

Lemma 3 (Indistinguishability of If an adversary causes an honest instance U to accept without a matching session, then within the data relied upon for its acceptance, there must exist a ciphertext ct' injected by the adversary via an empty direct channel, satisfying the following:

- ct' passes the verification steps specified by the protocol (otherwise, the instance would not enter the Accept state).
- Under the long-term decryption key dk_U of the accepting party, ct' can be correctly decrypted to a value other than $\perp$ (otherwise, a consistent session key cannot be derived, and the Accept state cannot be reached).

Proof: The necessary conditions for an honest instance to enter the Accept state directly imply the two points above. If the ciphertext originated from an honest peer with a matching session, this would contradict the premise of "acceptance without a matching session." Thus, the ciphertext must have been injected by the adversary.

 doi:10.20944/preprints202604.1123.v1

Lemma 4: If the probability of the event in Lemma 5.3 is non-negligible, then we can construct an adversary $\mathcal{B}_2$ that outputs a successful forgery in the forgery challenge of the underlying lattice-based cryptosystem IB-ME with non-negligible probability—a contradiction to the underlying hardness assumption.

Proof: Adversary $\mathcal{B}_2$ interacts with the challenger C_{For} of the underlying IB-ME scheme. C_{For} provides the master public key mpk and permits $\mathcal{B}_2$ to issue encryption key queries and decryption key queries (with restrictions on target identity pairs as per the challenge rules). The goal is to output a valid ciphertext that is decryptable under the target identity, satisfies the verification relation, and is not generated via the legitimate methods allowed by the challenger. Let $\mathcal{B}_2$ execute adversary A, and extract the forged ciphertext that causes the honest party's first acceptance in A's interaction as the forgery output (ct^*, snd^*, rcv^*) in the IB-ME authenticity game—where (snd^*, rcv^*) denote the sender and receiver identities, respectively. If, when this acceptance event occurs, the encryption key generation query $EKGen(snd^*)$ (corresponding to snd^* has never been made, then $\mathcal{B}_2$ obtains a forgery that satisfies the winning condition of the IB-ME authenticity game. $\mathcal{B}_2$ proceeds as follows:

(1) Initialize the system using the master public key mpk provided by challenger C_{For}, and forward mpk to adversary Adv.

(2) Forward and simulate's $EKGen$ and $DKGen$ queries to the query interfaces permitted by one, ensuring the simulation is indistinguishable from the real execution.

(3) Execute the Send/Execute oracles, and record all interaction data.

(4) Upon A inducing a non-matching acceptance, $\mathcal{B}_2$ extracts the ciphertext ct' from the recorded data as the forgery output and submits it to accordance with Lemma 3.

If the success probability of adversary Adv is non-negligible, then the forgery success probability of $\mathcal{B}_2$ is non-negligible and of the same order, which contradicts the underlying forgery infeasibility assumption. Additionally, the HMAC/HKDF/ZK primitives employed in the protocol are standard cryptographic components—their security is treated as negligible terms and incorporated into the overall security bound, and no separate proofs will be provided hereafter.

In summary, the security of the proposed protocol can be decomposed into two core components: Session Key Confidentiality: The semi-selective privacy of the underlying IB-ME scheme ensures that the embedded string $M_A /\!/ M_B$ is hidden from the adversary, from which session key confidentiality is derived. Implicit Authentication & Impersonation Resistance: The authenticity of the underlying IB-ME scheme prevents the adversary from injecting verifiable pseudo-ciphertexts, thereby guaranteeing implicit authentication and resistance to impersonation attacks. Combined with the security of the standard cryptographic components, the conclusion stated in Theorem 1 follows.

4.2. Safety Analysis

(1) **Resistance to Man-in-the-Middle (MitM) Attacks:** The protocol implements a bidirectional confirmation mechanism, where both parties generate ciphertexts bound to the counterparty's identity and embed identity information for verification during message exchange. Even if an adversary intercepts and tampers with msg_1,msg_2, the inclusion of identity tags and encrypted session-related materials in the messages prevents the adversary from forging valid ciphertexts or verification tags. Thus, MitM adversaries cannot successfully hijack or tamper with the key agreement process.

(2) **Resistance to Replay Attacks:** The protocol incorporates a version V and a timestamp tag in each message, binding session key generation to a time window. If an adversary attempts to replay outdated messages, the version number or timestamp will trigger verification failure, rendering the replay attack ineffective.

(3) **Resistance to Impersonation Attacks:** Through identity-bound ciphertexts and bidirectional authentication, the protocol ensures that both parties can only establish session keys with legitimate counterparts. Any impersonation attempt will be detected via identity verification or confirmation tag failure, leading to the rejection of forged messages [30].

 doi:10.20944/preprints202604.1123.v1

(4) **Resistance to Unknown Key-Share (UKS) Attacks:** The key derivation function KDF is bound to the version V and identity information, ensuring the uniqueness and security of session keys. Even if an adversary attempts to reconstruct session keys from known ciphertexts, the per-session binding of unique V and identities prevents cross-session key sharing or impersonation-based session establishment [31].

(5) **Resistance to Revocation Bypass Attacks:** A revocation list mechanism and blockchain-based maintenance enable distributed storage and verification of revocation information. Prior to session initiation, the protocol checks if participants are on the revocation list—revoked identities are denied session negotiation, thus preventing revocation bypass [32].

(6) **Forward Security:** A key version control mechanism ensures that the leakage of long-term keys does not compromise historical session keys. Each key update/derivation relies on a new version number and latest session information, making historical keys unrecoverable even if subsequent keys are leaked [33].

(7) **Resistance to Denial-of-Service (DoS) Attacks:** Verification of version numbers and revocation status filters invalid/revoked identities, preventing adversaries from consuming resources via forged requests. Further DoS risk mitigation can be achieved via optimized query and verification processes in practical implementations [34].

(8) **Resistance to Key Compromise Impersonation (KCI) Attacks:** Even if an adversary compromises the static private key dk_A, they cannot impersonate the legitimate party. While dk_A allows calculation of M_A, the adversary lacks the recipient's private key dk_B to compute M_B, thus gaining no information about the session key.

In conclusion, the proposed protocol comprehensively mitigates common active and passive attacks via a suite of mechanisms, including identity-bound encryption, bidirectional confirmation, version control, and revocation lists. These security mechanisms ensure that the protocol achieves provably secure key agreement and identity authentication even in adversarial environments, while avoiding unnecessary communication and computational overhead—rendering it highly deployable and practical for real-world applications.

5. Performance Evaluation

This section evaluates the performance of the proposed solutions. Since protocol security depends on the underlying computational problems and the choice of zero-knowledge proof system—and since different hardware platforms and implementation approaches can lead to significant variations in execution time—we adopt a parametric analysis method. Specifically, we analyze the inherent computational complexity and communication overhead (i.e., traffic volume) of the protocol using asymptotic formulae, thereby avoiding platform-specific dependencies on absolute runtime or resource measurements. Compared with similar theoretical solutions, this analysis clarifies the protocol's functional completeness as well as the trade-offs between performance efficiency and implementation cost.

5.1. Computing Cost Analysis

As specified in the agreement, Parties A and B decompose the protocol into several elementary cryptographic operations. Table 2 defines these basic operations and their corresponding notations. The total computational time required by both parties can thus be expressed as:

$$T_A = 2T_{samp} + 2T_{mult} + 3T_{hash} + T_{\pi G} + T_{\pi V} + 2T_{sym} \quad (20)$$

$$T_B = 2T_{samp} + 2T_{mult} + 3T_{hash} + T_{\pi G} + T_{\pi V} + T_{sym} \quad (21)$$

 doi:10.20944/preprints202604.1123.v1

Table 2. basic operation symbols and definitions.

Operation symbols	define	Operation symbol	Definition
T_{samp}	Discrete Gaussian sampling	T_{mult}	Matrix multiplication
T_{hash}	The hash operation	$T_{\pi G}$	Zero-knowledge proofs
$T_{\pi V}$	Zero knowledge proof	T_{sym}	generate HMAC/HKDF symmetry operations

In practical deployment, the latency of each operation is determined by the selected hardware platform and zero-knowledge proof (ZKP) instantiation scheme. For example, on an Intel i7 platform, matrix multiplication T_{mult} takes approximately 12.5μs , Gaussian sampling ~0.8μs, and hashing ~0.2μs. Notably, ZKP latency is system-dependent: zk-STARK generation and verification take ~40μs and ~25μs, respectively . The proposed protocol treats ZKPs as pluggable components, enabling flexible selection based on security-efficiency tradeoffs in deployment. With an efficient zk-STARK, single-party computation latency can be constrained to under 100μs—comparable to state-of-the-art post-quantum protocols.

Table 3 compares the computational complexity (measured by operation counts) of the proposed protocol with related schemes. The primary overhead of the proposed protocol stems from matrix multiplication and ZKPs, whereas bilinear pairing-based schemes rely on pairing and exponentiation operations (typically more latency-intensive). Compared to Wei et al.'s lattice-based AKA protocol, the proposed protocol exhibits a similar operation count but additionally supports revocation and version evolution functionalities.

Table 3. computational overhead.

Protocols	Major operation counts	Functionality completeness
Ours	$4T_{mult}+2T_{\pi}+6T_{\text{hash}}$	Identity binding + implicit authentication + revocation + version evolution
Bos et al. [35]	$2T_{mult}+2T_{\text{hash}}$	KEM only
Wei et al. [12]	$4T_{mult}+2T_{pairing}$	Identity binding + Implicit authentication
Ateniese et al.[23]	$2T_{pairing}+2T_{exp}$	Identity binding + implicit authentication

To provide an intuitive comparison of the computational complexity across protocols, Figure 3 illustrates the count of core cryptographic operations—including matrix multiplication, hash operations, Gaussian sampling, and zero-knowledge proof (ZKP) generation/verification—between the proposed protocol and three benchmark schemes: Kyber [35], Wei et al.'s lattice-based AKA [12], and Ateniese et al.'s bilinear pairing-based protocol [23]. Here, the vertical axis denotes the number of operations, while the horizontal axis categorizes the operation types.

 doi:10.20944/preprints202604.1123.v1

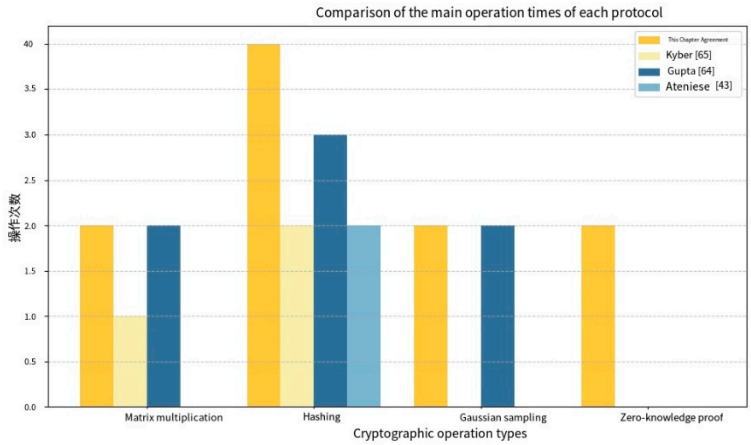

Figure 3. Comparison of the number of main cryptographic operations of each protocol.

It can be observed from the figure that the protocol's primary computational operations consist of two matrix multiplications and one zero-knowledge proof each for generation and verification. This operational structure arises directly from the protocol's design objective: to embed identity binding and implicit authentication into ciphertext-based interaction. Compared with Kyber—which provides only a key encapsulation mechanism (KEM)—our protocol introduces zero-knowledge proofs while maintaining a comparable number of matrix multiplications. Relative to Wei et al.'s lattice-based authenticated key agreement (AKA) protocol, our scheme exhibits a similar overall operational count but further supports revocation and version evolution—functionalities absent in their design. In contrast, Ateniese et al.'s bilinear pairing–based scheme avoids both matrix multiplication and zero-knowledge proofs; however, its pairing operations are typically more computationally expensive (≈12.5 μs per operation versus ≈1.2 μs for a matrix multiplication) and it lacks support for revocation. Consequently, the proposed protocol incurs no significant increase in computational overhead relative to comparable schemes, while delivering enhanced security functionality. As corroborated by the theoretical analysis presented in Table 3, the protocol's computational complexity is predominantly governed by highly parallelizable linear algebra operations; the zero-knowledge proof component, being modular and pluggable, can be flexibly adapted to specific deployment constraints. Overall, the computational cost remains well within acceptable bounds for post-quantum cryptographic protocols.

5.1. Communication Overhead Analysis

The format and length of messages exchanged in the protocol are parameterized by system configurations. We fix the matrix column count $m = 1024$, matrix element size $\log q = 4Byte (q \approx 2^{32})$, hash value length L_H=32Byte bytes, and version identifier length L_v=4Byte. The zero-knowledge proof (ZKP) length L_π is system-dependent. Below are the message lengths for each of the four rounds in a full key agreement session:

- Round 1 ($A \rightarrow B$): Contains c_0, c_1, π, v, with a total length of $2m \log q + L_\pi + L_v$ bytes.
- Round 2 ($B \rightarrow A$): Identical to Round 1.
- Round 3 ($A \rightarrow B$): Consists solely of h_A, with a length of L_H bytes.
- Round 4 ($B \rightarrow A$): Identical to Round 2.

The total amount of communication is expressed as follows:

$$Comm = 4m \log q + 2L_\pi + 2L_v + 2L_H \quad (5\text{-}22)$$

Plug in parameters to get $Comm = 16392 + 2L_\pi$ bytes.

To further analyze the composition of the protocol's communication overhead, Figure 4 presents a stacked bar chart illustrating the total communication volume across different zero-knowledge proof instantiations. In this figure, the total communication traffic is decomposed into two components: the base ciphertext portion—which encompasses the matrix ciphertext, hash values,

 doi:10.20944/preprints202604.1123.v1

version identifiers, and related metadata—and the zero-knowledge proof portion—arising from the aggregate proof length of the two-round protocol. The base ciphertext portion remains constant at approximately 16.3 KB, as determined by the system parameters; in contrast, the zero-knowledge proof portion varies depending on the specific instantiation employed.

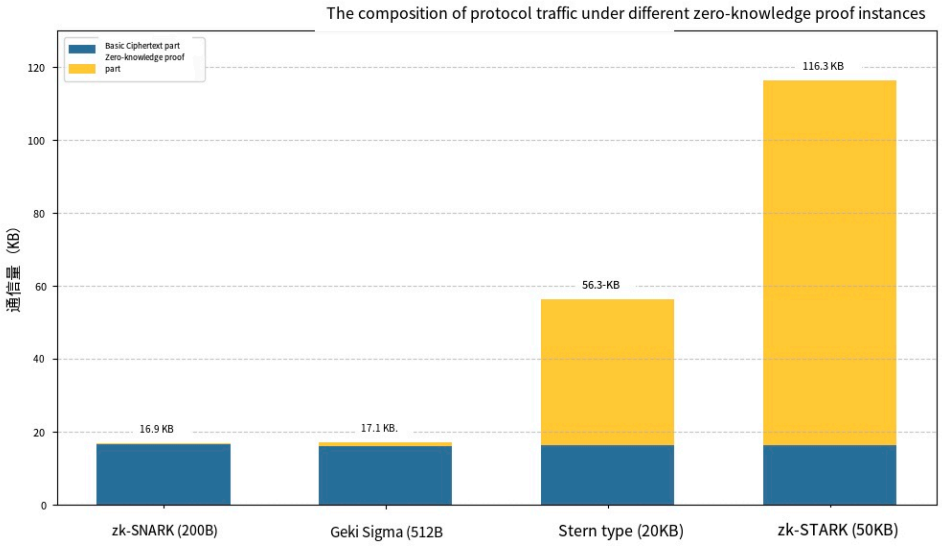

Figure 4. Total communication under different instantiations of zero-knowledge proofs.

If a zk-SNARK—with a typical proof size of approximately 200 bytes—is employed, the zero-knowledge proof contributes only 0.4 KB to the total communication overhead, yielding an overall communication cost of ≈16.9 KB.

If instead a lattice-based Sigma protocol (proof size ≈512 bytes) is adopted, the total communication increases to ≈17.1 KB, with the proof component accounting for 1 KB.

When the Stern protocol —whose typical proof length is 20 KB—is used, the total communication rises to 56.3 KB, of which 40 KB stems from the proof itself.

Finally, deploying zk-STARK—with a typical proof size of 50 KB—results in a total communication volume of 116.3 KB, with the proof constituting 100 KB of that amount.

These results demonstrate that the proportion of the zero-knowledge proof within the total communication traffic is controllable and can be flexibly adjusted by selecting appropriate cryptographic primitives according to practical security requirements: zk-SNARKs minimize communication overhead under bandwidth-constrained conditions, whereas zk-STARKs—though incurring higher communication costs—offer transparency and post-quantum security. Notably, even the largest overhead (116.3 KB) remains well within acceptable limits for modern high-speed networks: over a 5G link, this corresponds to a transmission latency of only ≈9.3 ms.

Table 4 compares the communication cost of the proposed protocol with related schemes. The reported figure of 17.1 KB is fully compatible with contemporary high-speed network environments. For instance, under ideal 5G conditions—with a theoretical peak data rate exceeding 100 Mbps—the transmission of 17.1 KB requires merely ≈1.4 ms. Moreover, this communication cost inherently integrates two critical functionalities: revocation status verification and version synchronization. In contrast, conventional approaches require separate CRL lookups or OCSP requests—entailing 2–3 round-trip interactions and several kilobytes of additional overhead—thereby incurring higher end-to-end latency in practice. Consequently, the observed communication overhead represents a reasonable and justified trade-off for achieving strong, formally grounded security semantics.

 doi:10.20944/preprints202604.1123.v1

Table 4. Comparison of communication costs.

Protocols	Traffic (KB)	Rounds
Chapter Agreement	17.1	4
Bos et al. [35]	1.2	2
Wei et al. [12]	3.2	3
Ateniese et al. [23]	2.5	4

6. Conclusions

Addressing the security requirements for key agreement under post-quantum threats in open network environments, this paper proposes and refines a unified protocol framework that simultaneously realizes identity binding, session key establishment, key update, and revocation management. Centered on the core cryptographic mechanisms of the proposed protocol, we provide a comprehensive exposition—including phase decomposition, message interaction design, formal security analysis, and performance evaluation.

At the protocol level, shared secret materials are exchanged and mutual consistency is established via ciphertexts tightly bound to the peer's identity; a bidirectional confirmation mechanism is further incorporated to mitigate the risk of negotiation divergence. For key lifecycle management, we introduce a version-control tag to orchestrate synchronized key evolution-thereby reducing reliance on external coordination and minimizing auxiliary communication overhead. Regarding revocation management, a blockchain-based distributed architecture is employed to jointly maintain the revocation list, enabling decentralized verification and full auditability of revocation status—thus enhancing both scalability and operational feasibility of the mechanism. Security analysis demonstrates that the protocol preserves session key confidentiality and withstands typical active attacks under the assumed adversary model—including specified adversarial capabilities and query oracles. Performance evaluation indicates that the per-end computational overhead remains below 100 μs, while the total communication overhead is approximately 17.1 KB—validating the protocol's pragmatic trade-off between moderate resource consumption and comprehensive security functionality.

Author Contributions: Conceptualization, Yuxia Qian; Methodology, Yincheng Liang; Software, Xinqi Dong; Validation, Yiwen Liang; Writing – original draft, Xinqi Dong. All authors will be updated at each stage of manuscript processing, including submission, revision, and revision reminder, via emails from our system or the assigned Assistant Editor.

Funding: This study is supported by the Foundation of National Natural Science Foundation of China (Grant No.: 62372266, 62472251); This work is supported by the Open Research Fund of Guangdong Key Laboratory of Blockchain Security, Guangzhou University (No. 2023B1212010013).

Data Availability Statement: The data is available if requested.

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

References

1. F. Barrett-Danes and F. Ahmad. Quantum computing and cybersecurity: a rigorous systematic review of emerging threats, post-quantum solutions, and research directions (2019–2024)[J]. Discover Applied Sciences, 2025, 7(10): 1083.
2. P. Ravi, J. Howe, A. Chattopadhyay, et al. Lattice-based key-sharing schemes: a survey[J]. ACM Computing Surveys (CSUR), 2021, 54(1): 1-39.
3. M. Ajtai. Generating hard instances of lattice problems. Proceedings of the twenty-eighth annual ACM symposium on Theory of computing[C]. New York: ACM, 1996. 99-108.

 doi:10.20944/preprints202604.1123.v1

4. M. Wang and G. L. Long. Lattice-based access authentication scheme for quantum communication networks[J]. Science China Information Sciences, 2024, 67(12): 222501.
5. H. A. Hameed, H. J. S. Alhamedi, W. F. Abbas, et al. PQ-Lattice: a lattice-based post-quantum authentication protocol for decentralized IoT systems[J]. Informatica, 2025, 49(35).
6. Z. Yi, X. Du, Y. Liao, et al. An access authentication algorithm based on a hierarchical identity-based signature over lattice for the space-ground integrated network. 2019 International Conference on Advanced Communication Technologies and Networking (CommNet)[C]. IEEE, 2019. 1-9.
7. S. Wang, C. Xu, G. Zhao, et al. APQA: an anonymous post quantum access authentication scheme based on lattice for space ground integrated network[J]. Computer Networks, 2025, 257: 110979.
8. G. Liu, H. Li, J. Le, et al. Lrcpa: lattice-based robust and conditional privacy-preserving authentication for vanets[J]. IEEE Transactions on Vehicular Technology, 2024, 74(3): 4698-4712.
9. V. Lyubashevsky. Fiat-Shamir with aborts: applications to lattice and factoring-based signatures. International conference on the theory and application of cryptology and information security[C]. Springer, 2009. 598-616.
10. L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. Annual Cryptology Conference [C]. Springer, 2014. 335-352.
11. M. J. Hossain, C. Xu, Y. Zhang, et al. LAMA: a secure lattice-based authentication scheme for cloud storage against misbehaved private key generator[J]. Journal of Ambient Intelligence and Humanized Computing, 2023, 14(7): 8613-8629.
12. G. Wei, K. Fan, K. Zhang, et al. Lower rounds lattice-based anonymous AKA under the seCK model for the IoT[J]. Peer-to-Peer Networking and Applications, 2024, 17(4): 2031-2046.
13. P. Rewal, M. Singh, D. Mishra, et al. Quantum-safe three-party lattice based authenticated key agreement protocol for mobile devices[J]. Journal of Information Security and Applications, 2023, 75: 103505.
14. V. P. Ojha, S. Chauhan, S. Yarahmadian, et al. Unfolding post-quantum cryptosystems: CRYSTALS-Dilithium, McEliece, BIKE, and HQC[J]. Mathematics, 2025, 13(17): 2841.
15. D. Micciancio and O. Regev. Lattice-based cryptography. D. J. Bernstein, J. Buchmann and E. Dahmen. Post-Quantum Cryptography[C]. Berlin, Heidelberg: Springer, 2009. 147-191.
16. A. Pellet-Mary, G. Hanrot and D. Stehlé. Approx-SVP in ideal lattices with pre-processing. Annual international conference on the theory and applications of cryptographic techniques[C]. Springer, 2019. 685-716.
17. Q. Guo and T. Johansson. Faster dual lattice attacks for solving LWE with applications to CRYSTALS. International Conference on the Theory and Application of Cryptology and Information Security[C]. Springer, 2021. 33-62.
18. Y. Du and X. Ma. On the rejection rate of exact sampling algorithm for discrete Gaussian distributions over the integers[J]. Theory of Computing Systems, 2022, 66(6): 1099-1122.
19. D. Micciancio and C. Peikert. Trapdoors for lattices: simpler, tighter, faster, smaller. Advances in Cryptology – EUROCRYPT 2012[C]. Berlin, Heidelberg: Springer, 2012. 700-718.
20. Z. Yang, D. H. Duong, W. Susilo, et al. Hierarchical identity-based signature in polynomial rings[J]. The Computer Journal, 2020, 63(10): 1490-1499.
21. A. Shamir. Identity-based cryptosystems and signature schemes. Workshop on the theory and application of cryptographic techniques[C]. Springer, 1984. 47-53.
22. R. K. Zhao, S. McCarthy, R. Steinfeld, et al. Quantum-safe HIBE: does it cost a Latte?[J]. IEEE Transactions on Information Forensics and Security, 2023, 19: 2680-2695.
23. G. Ateniese, D. Francati, D. Nuñez, et al. Match me if you can: matchmaking encryption and its applications. Annual International Cryptology Conference[C]. Springer, 2019. 701-731.
24. S. Goldwasser, S. Micali and C. Rackoff. The knowledge complexity of interactive proof-systems. Providing sound foundations for cryptography: on the work of shafi goldwasser and silvio micali[C]. 2019. 203-225.
25. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. Conference on the theory and application of cryptographic techniques[C]. Springer, 1986. 186-194.
26. V. Lyubashevsky. Lattice signatures without trapdoors. Annual International Conference on the Theory and Applications of Cryptographic Techniques[C]. Springer, 2012. 738-755.

 doi:10.20944/preprints202604.1123.v1

27. K. Xue, W. Meng, S. Li, et al. A secure and efficient access and handover authentication protocol for Internet of Things in space information networks[J]. IEEE Internet of Things Journal, 2019, 6(3): 5485-5499.
28. R. Ma, J. Cao, D. Feng, et al. LAA: lattice-based access authentication scheme for IoT in space information networks[J]. IEEE Internet of Things Journal, 2019, 7(4): 2791-2805.
29. J. Guo, Y. Du, Y. Zhang, et al. A provably secure ECC-based access and handover authentication protocol for space information networks[J]. Journal of Network and Computer Applications, 2021, 193: 103183.
30. Y. Liu, L. Ni and M. Peng. A secure and efficient authentication protocol for satellite-terrestrial networks[J]. IEEE Internet of Things Journal, 2022, 10(7): 5810-5822.
31. J. Guo, S. Yao, Y. Song, et al. N3PA-STIN: a novel three-party authentication protocol for multiuser access in satellite terrestrial integrated networks[J]. IEEE Internet of Things Journal, 2025, 12(10): 14952-14968.
32. A. A. Almazroi, M. A. Alqarni, M. A. Al-Shareeda, et al. L-CPPA: lattice-based conditional privacy-preserving authentication scheme for fog computing with 5G-enabled vehicular system[J]. PLoS ONE, 2023, 18(10): e0292690.
33. M. El-Hajj and B. Oude Roelink. Evaluating the efficiency of zk-snark, zk-stark, and bulletproof in real-world scenarios: a benchmark study[J]. Information, 2024, 15(8): 463.
34. S. K. H. Islam and S. Zeadally. Provably secure identity-based two-party authenticated key agreement protocol based on CBi-ISIS and Bi-ISIS problems on lattices[J]. Journal of Information Security and Applications, 2020, 54: 102540.
35. J. Bos, L. Ducas, E. Kiltz, et al. CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. 2018 IEEE European symposium on security and privacy (EuroS&P)[C]. IEEE, 2018. 353-367.