future internet

MDPI

Article

Security Analysis and Designing Advanced Two-Party Lattice-Based Authenticated Key Establishment and Key Transport Protocols for Mobile Communication

Mani Rajendran [1], Dharminder Chaudhary [2], S. A. Lakshmanan [3,*] and Cheng-Chi Lee [4,5,*]

[1] Department of Mathematics, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Chennai 601103, India
[2] Department of Artificial Intelligence, Amrita School of Artificial Intelligence, Amrita Vishwa Vidyapeetham, Faridabad 121002, India
[3] Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Chennai 601103, India
[4] Department of Library and Information Science, Fu Jen Catholic University, New Taipei City 24205, Taiwan
[5] Department of Computer Science and Information Engineering, Fintech and Blockchain Research Center, Asia University, Taichung City 41354, Taiwan
* Correspondence: sa_lakshman@ch.amrita.edu (S.A.L.); cclee@mail.fju.edu.tw (C.-C.L.)

Abstract

In this paper, we have proposed a two-party authenticated key establishment (AKE), and authenticated key transport protocols based on lattice-based cryptography, aiming to provide security against quantum attacks for secure communication. This protocol enables two parties, who may share long-term public keys, to securely establish a shared session key, and transportation of the session key from the server while achieving mutual authentication. Our construction leverages the hardness of lattice problems Ring Learning With Errors (Ring-LWE), ensuring robustness against quantum and classical adversaries. Unlike traditional schemes whose security depends upon number-theoretic assumptions being vulnerable to quantum attacks, our protocol ensures security in the post-quantum era. The proposed protocol ensures forward secrecy, and provides security even if the long-term key is compromised. This protocol also provides essential property key freshness and resistance against man-in-the-middle attacks, impersonation attacks, replay attacks, and key mismatch attacks. On the other hand, the proposed key transport protocol provides essential property key freshness, anonymity, and resistance against man-in-the-middle attacks, impersonation attacks, replay attacks, and key mismatch attacks. A two-party key transport protocol is a cryptographic protocol in which one party (typically a trusted key distribution center or sender) securely generates and sends a session key to another party. Unlike key exchange protocols (where both parties contribute to key generation), key transport protocols rely on one party to generate the key and deliver it securely. The protocol possesses a minimum number of exchanged messages and can reduce the number of communication rounds to help minimize the communication overhead.

Keywords: data security; network security; mobile communications; wireless communications security; key establishment; cryptography

Academic Editor: Gianluigi Ferrari

Received: 22 July 2025
Revised: 24 September 2025
Accepted: 29 September 2025
Published: 16 October 2025

Citation: Rajendran, M.; Chaudhary, D.; Lakshmanan, S.A.; Lee, C.-C. Security Analysis and Designing Advanced Two-Party Lattice-Based Authenticated Key Establishment and Key Transport Protocols for Mobile Communication. *Future Internet* **2025**, *17*, 472. https://doi.org/10.3390/fi17100472

1. Introduction

In the rapidly evolving landscape of mobile communications, ensuring both secure key exchange and user anonymity has become increasingly critical. Traditional cryptographic

primitives, largely based on number-theoretic assumptions such as RSA or ECC, are vulnerable to quantum adversaries due to the development of quantum algorithms like Shor's algorithm [1]. This emerging threat necessitates the exploration of post-quantum secure cryptographic schemes. Among the promising candidates, lattice-based cryptography, especially based on the hardness of problems like Ring Learning with Errors (RLWE), has gained significant attention for its strong security guarantees and computational efficiency. RLWE is an extension of the LWE problem [2], which is widely used as the basis for many cryptographic primitives in lattice-based cryptography. In the Ring-LWE problem [3], the underlying structure is a polynomial ring rather than a Euclidean lattice. The problem involves deducing the coefficients of a random polynomial having small coefficients when given noisy evaluations of the polynomial at various points in the ring.

This paper presents a novel two-party authenticated key establishment protocol that leverages ideal-lattice constructions to ensure post-quantum security while simultaneously preserving user anonymity, a critical requirement in privacy-sensitive mobile communication environments. Our protocol ensures mutual authentication between communicating parties and derives a shared session key resistant to quantum and classical attacks. Furthermore, the protocol incorporates lightweight mechanisms to provide identity concealment, making it suitable for deployment in resource-constrained mobile devices. We provide a detailed formal security analysis [4] of the proposed protocol under the standard security model and demonstrate that it satisfies crucial security properties including resistance against man-in-the-middle, impersonation, and key compromise impersonation attacks. In addition, we prove the correctness and performance efficiency of the protocol through both theoretical and experimental evaluations. Our work contributes to the growing body of post-quantum secure mobile communication protocols by offering a practical and secure solution that meets the dual requirements of authenticated key exchange and anonymous communication, paving the way for future mobile systems resilient to quantum-era threats.

2. Related Work

Authenticated Key Exchange (AKE) and transport protocols are cryptographic protocols that allow two parties to establish a session key over a public channel, while ensuring mutual authentication in the presence of active adversaries. In the era of quantum computers, traditional AKE protocols, whose security relies upon RSA, ECC, or discrete logarithms, are no longer considered secure. Ideal lattice-based assumptions such as Learning With Errors (LWE), and its advanced version Ring Learning With Errors (Ring-LWE) provide a promising foundation for post-quantum AKE protocols, offering both efficiency and security against quantum computers. Using these assumptions, several novel key exchange techniques (KETs) have been put forward, see [5–19]. Furthermore, these techniques are broadly classified into two categories (Table 1):

- A key transport protocol is a mechanism in which the sender generates a secret session key and securely transfers it to the receiver using a public key encryption scheme or a Key Encapsulation Mechanism (KEM).
- A key agreement protocol enables both the sender and receiver to actively contribute to the generation of a shared session key, typically through an interactive message exchange.

Table 1. Comparison of lattice-based key transport and key agreement protocols.

Aspect	Key Transport Protocols	Key Agreement Protocols
Interaction	One-way (non-interactive)	Two-way (interactive)
Key Generation	Performed by one party	Jointly derived by both
Forward Secrecy	Not automatic	Naturally supported
Efficiency	Often more compact and faster	Slightly higher overhead
Authentication	Needs explicit digital signatures	Can support implicit/explicit
Lattice Basis	LWE/Module-LWE + KEM	Ring-LWE + reconciliation
Example	Kyber, FrodoKEM	NewHope, BCNS, Lizard

The first ideal lattice-based key exchange protocol using the reconciliation technique was proposed by Ding et al. [5]. In 2015 Zhang et al. proposed a method [20] that avoids reliance on additional cryptographic primitives, particularly digital signatures, which helps to streamline the protocol and allows the security to be grounded directly in the hardness of the Ring Learning With Errors (Ring-LWE) problem. The security is formally established within the Bellare–Rogaway model, offering weak perfect forward secrecy under the random oracle assumption. In 2016, Fluhrer et al. [21] demonstrated that several Ring-LWE-based key exchange protocols can be compromised when the same key share is reused across multiple sessions. The findings highlight the critical need for generating fresh key shares for each individual exchange to maintain security. However, this attack strategy proves to be ineffective against several other key agreement protocols that utilize the least significant bits (LSBs) of intermediate shared secrets to derive the final session keys between the two communicating parties. By extracting keys from LSBs, these protocols inherently introduce a non-linearity that reduces the exploitable structure available to adversaries, thereby enhancing resistance to key recovery and mismatch-based attacks. For example, this attacking strategy would not work on the scheme proposed by Ding et al. [6]. They designed a new attacking technique on Ring-LWE-based key exchange protocols using reconciliation approaches, and they named it the signal leakage attack (SLA). In an SLA, an adversary can exploit the reuse of key pairs to compromise the security of the cryptographic scheme. Reusing private or public keys across multiple sessions introduces correlation in computations, which can be leveraged through leakage analysis to recover secret information, thereby undermining the protocol's confidentiality guarantees. Feng et al. [7] proposed an ideal lattice-based anonymous authentication protocol for a mobile client–server communication system. A comprehensive security proof illustrates that the protocol is provably secure in the random oracle model, relying on the hardness of the Ring Learning With Errors (Ring-LWE) assumption. However, the Feng et al. scheme [7] was found to be susceptible to various attacks like spoofing, manipulation, and anonymity violation. In 2020, Dharminder et al. [8] introduced a new Ring-LWE-based quantum-safe key agreement protocol. They also carried out implementation in Lattice crypto, MIRACL lib, and proved its security in the ROM model. In the year 2020, Dabra et al. proposed a new scheme [9] which is anonymous, and they claimed it to be secure against signal leakage attacks. In the scheme in [9], Chaudhary et al. [22] identified that instead of engaging the master key pw, a user server utilizes a pseudo identity $psid_i$ and pseudo secret s^*, and adds $\{psid_i, s^*, pw\}$ into the repository. This is problematic as any dishonest party $\mathfrak{D}$ can easily create a session key for any user once the party gets access to the repository [22]. In the year 2020, Islam et al. proposed a protocol [23] based on CBi-ISIS that is short integer solution assumption secure against quantum attacks. In 2020, Rana et al. proposed a scheme [24] to mitigate the security vulnerabilities introduced by Shor's algorithm.

They proposed a lattice-based key agreement protocol based on the Ring Learning With Errors (RLWE) assumption. However, their scheme [24] suffers key mismatch and some mathematical errors while establishing the session key. In 2022, Akleylek et al. proposed a protocol [25] based on short integer solution assumption, and this scheme supports tag, server, reader, and informal security analysis. In the year 2021, Wang et al. proposed a protocol [26], and they took an initial step toward addressing this challenge by presenting Quantum2FA—a practical, quantum-resistant, smart card-based password authentication scheme. The design integrates Alkim et al.'s lattice-based key exchange protocol [27] with Wang and Wang's "fuzzy-verifier plus honeywords" technique [28]. Quantum2FA effectively mitigates the recently discovered key reuse attacks (ACISP 2018; CT-RSA 2019) targeting lattice-based key exchange protocols, specifically by addressing two major threats: signal leakage attacks and key mismatch attacks. Although this protocol needs the exchange of only three messages to establish a session key, it suffers from computation cost overhead. In 2020, Islam et al. [10] proposed a two-party quantum-safe authenticated key agreement protocol, and proved their protocol is provably secure. In 2023, Kumar et al. [12] studied many protocols [5–9,12,13], and they found these protocols are susceptible to signal leakage attacks due to their reuse of the long-term key. In the year 2023, Mishra et al. [29] proposed a quantum enhanced authentication and key agreement protocol for autonomous vehicles, but this protocol does not support unlinkability. In this scheme, the vehicle user sends the message $< PID_i, x_i, \Sigma_i >$ which contains non-repeating PID_i, and it helps to link older messages with the new coming message. In the same yearg, Moony et al. [13] introduced a post-quantum two-party authentication protocol for mobile devices. This protocol lacks key control, as the user is able to compute the session key independently, without any direct involvement or contribution from the server. Moreover, the overall security of the protocol heavily relies on password-based authentication, yet it fails to incorporate robust countermeasures against offline password guessing attacks, leaving it vulnerable to brute-force exploitation. Recently, Pursharthi et al. [15] proposed a quantum-safe authenticated key agreement scheme for mobile devices, but their protocol does not provide unlinkability. One can observe that user U_i sends information $\{PID_i,\ x_i,\ T_i,\ \sum_i\}$ to the server over a public channel. In the message $\{PID_i,\ x_i,\ T_i,\ \sum_i\}$, the value PID_i is chosen by the server, and it is a fixed value for each of the sessions; therefore, this protocol cannot provide unlinkability. Moreover, this protocol uses learning with error multiple times on the same component $k_j = (x_i.s_j + 2c)d + 2e'$ that will increase the error added from the Gaussian; therefore, this is mathematically incorrect. These findings underscore the urgent need for carefully designed lattice-based protocols that offer strong security properties such as unlinkability, key control, and resilience against side-channel and guessing attacks, while maintaining correctness and efficiency under practical constraints.

3. Authenticated Key Establishment and Key Transport Protocols

In modern cryptographic systems, establishing a secure communication channel is foundational to ensuring the confidentiality, integrity, and authenticity of exchanged information. Two primary classes of cryptographic protocols—Authenticated Key Establishment (AKE) and key transport protocols—address this need by enabling entities to securely agree on or distribute cryptographic keys.

3.1. Authenticated Key Establishment

Authenticated Key Establishment (AKE) protocols enable two or more parties to mutually derive a shared secret key over an insecure channel, ensuring that the key is known only to the legitimate participants. Unlike unauthenticated key exchange mechanisms that are susceptible to man-in-the-middle (MitM) attacks, AKE protocols incorporate authenti-

cation mechanisms such as digital signatures, certificates, or message authentication codes (MACs) to confirm the identity of communicating parties. These protocols are widely used in internet security protocols such as TLS, IPsec, and SSH, and are especially important in scenarios requiring forward secrecy, where the compromising of long-term keys does not affect past session keys.

With the advent of quantum computing, classical AKE protocols based on discrete logarithms and elliptic curves are being reconsidered in favor of post-quantum AKE schemes, particularly those based on lattice problems, code-based, or hash-based cryptography.

3.2. Key Transport Protocols

In contrast, key transport protocols involve a single party generating and securely transmitting a key to another party. This model is often simpler and more efficient than key agreement protocols but typically lacks forward secrecy. Key transport mechanisms usually rely on asymmetric encryption: a session key is generated by the sender and encrypted with the recipient's public key. Upon receiving, the recipient uses their private key to decrypt the session key. Such protocols are widely used in email encryption (e.g., S/MIME), digital envelopes, and centralized systems like Kerberos, where a trusted authority distributes session keys.

3.3. Importance and Applications

Both AKE and key transport protocols serve as the backbone of secure communication protocols, enabling encrypted messaging, secure file transfer, protected authentication, and more. Choosing between them depends on the security requirements of the application—whether mutual contribution to the key and forward secrecy are critical, or centralized distribution suffices.

In the evolving landscape of cybersecurity, designing efficient, scalable, and quantum-resistant key establishment and transport mechanisms remains a significant research challenge, particularly for applications such as Internet of Things (IoT), vehicular networks, mobile communication, and cloud services.

4. Motivation and Contribution

With the increasing reliance on digital communication across domains such as finance, healthcare, critical infrastructure, and defense, secure and authenticated key establishment has become a cornerstone of modern cryptographic systems. Traditional public key primitives like RSA, Diffie–Hellman, and Elliptic Curve Cryptography (ECC), while widely used, are vulnerable to quantum algorithms such as Shor's algorithm, which threatens to break their underlying hardness assumptions. This looming threat has spurred significant interest in post-quantum cryptography, with lattice-based cryptography emerging as one of the most promising alternatives. Lattice-based cryptography offers quantum-resistant alternatives, providing security even in the presence of powerful quantum adversaries. There are many advantage of the proposed protocol compared with existing protocols.

- An authenticated key transport protocol is a cryptographic mechanism that securely delivers a secret key from a sender to a receiver over an insecure channel while ensuring the authenticity of both the key and the communicating parties. The proposed authenticated key transport protocol (see section 7.4) is an anonymous two-party lattice-based key protocol that enables two parties to communicate without revealing their identities to the adversary.
- These protocols ensure that forward secrecy enables two parties to establish a secret key securely while ensuring that compromise of long-term secret keys does not compromise the confidentiality of past communication. Even if an attacker gains access

to a party's long-term secret key (e.g., a private signing or decryption key), they cannot retroactively decrypt previously recorded communications or derive previously established session keys.

- To ensure real-world security of post-quantum communication protocols, especially on untrusted platforms, resisting signal leakage attacks is as essential as mathematical soundness. This protocol resists well-known signal leakage attacks, and is efficient in both communication and computation overheads.

5. Preliminaries

Preliminaries and notations (see Table 2) for lattice-based key exchange protocols provide the foundational elements and symbols necessary for understanding, analyzing, and implementing these cryptographic protocols. The function f(x) can also be defined as cyclotomic polynomial $x^n + 1$. Let us take a finite ring $R = \frac{\mathbb{Z}[x]}{<f(x)>}$, and the finite ring $R_q = \frac{\mathbb{Z}_q[x]}{<f(x)>}$ with integer coefficients. Here R_q is interpretable as a polynomial ring over $\mathbb{Z}_q$, where $a = a_0 + a_1.x + ... + a_{n-1}.x^{n-1}$. L_2 and L_∞ standard can also be described as $\| a \| = \sqrt{a_0^2 + a_1^2 + ... + a_{n-1}^2}$ and $\| a \|_\infty = max\{|a_i|\}$.

Definition 1. *If we are given a basis $B \in \mathbb{Z}^{M \times M}$ of some lattice, then the problem is to find a nonzero lattice point $B \cdot y$ ($y \in \mathbb{Z}^M \setminus \{0\}$) satisfying for all $x \in \mathbb{Z}^M \setminus \{0\}$:*

$$\|B \cdot y\| \leq \|B \cdot x\|. \tag{1}$$

Example 1. *Given*

$$B = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}, \tag{2}$$

one finds the shortest nonzero lattice point by computing the norms:

$$\|B \cdot y\| = \min\{\sqrt{10}, \sqrt{5}, \sqrt{5}\} = \sqrt{5}. \tag{3}$$

Definition 2 (Closest Vector Problem (CVP))**.** *Given a target point $t \in \mathbb{Z}^M$ and a basis $B \in \mathbb{Z}^{M \times M}$ for a lattice, the problem is to find $y \in \mathbb{Z}^M$ satisfying*

$$\|B \cdot y - t\| \leq \|B \cdot x - t\|, \quad \forall x \in \mathbb{Z}^M. \tag{4}$$

Example 2. *Given*

$$t = \begin{bmatrix} 4 \\ 7 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \tag{5}$$

one computes the closest lattice point and finds

$$y = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad B \cdot y = \begin{bmatrix} 5 \\ 4 \end{bmatrix}, \quad \textit{distance } \sqrt{10}. \tag{6}$$

Definition 3 (Ring Learning With Error (RLWE) Assumption)**.** *Given $(a, b) \in R_q \times R_q$, where $a = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \in R_q$ is sampled uniformly and $b = a \cdot s + e$ with $s, e \leftarrow Sample(\mathcal{D}_\beta^n)$, it is computationally hard to recover s given (a, b).*

Definition 4 (Decision Ring Learning With Error (DRLWE) Assumption)**.** *Let $R_q = \frac{\mathbb{Z}_q[x]}{\langle f(x) \rangle}$, where q is prime and $f(x)$ has degree n. Given a pair $(a, b) \in R_q \times R_q$, the task is to distinguish between:*

- *$(a, b = a \cdot s + e)$ for $s, e \leftarrow Sample(\mathcal{D}_\beta^n)$, and*

- *uniformly random* $(a^*, b^*) \in R_q \times R_q$.

Let

$$\mathbb{Z}_q = \left\{-\frac{q-1}{2}, \ldots, \frac{q-1}{2}\right\}, \quad E = \left\{-\left\lfloor\frac{q}{4}\right\rfloor, \ldots, \left\lfloor\frac{q}{4}\right\rfloor\right\}. \tag{7}$$

We define the *signal* function $Cha(\cdot)$ (also called the characteristic function) [20]:

$$Cha(v) = \begin{cases} 0, & v \in E, \\ 1, & \text{otherwise.} \end{cases} \tag{8}$$

Another function $Mod_2 : \mathbb{Z}_q \times \{0,1\} \to \{0,1\}$ is defined as

$$Mod_2(v, w) = v + w \cdot \frac{q-1}{2}. \tag{9}$$

If $v \in \mathbb{Z}_q$ and $w = Cha(v)$, then for $u = v + 2e$, where e is a small error vector,

$$Mod_2(v, w) = Mod_2(u, w). \tag{10}$$

Definition 5. *Consider q an odd prime, and $u, v \in R_q$ with $u = v + 2e$, $e \leftarrow \mathcal{D}_\beta^n$ such that $\|u - v\| < q/8$, and $w \leftarrow Cha(v)$. Then*

$$Mod_2(v, w) = Mod_2(u, w). \tag{11}$$

The $Cha(\cdot)$ function can be extended component-wise to elements in R_q:

$$v = v_0 + v_1 x + \cdots + v_{n-1} x^{n-1} \implies Cha(v) = (Cha(v_0), \ldots, Cha(v_{n-1})). \tag{12}$$

Similarly, $Mod_2(\cdot)$ can be applied component-wise in R_q.

Table 2. Symbols'/Notations' descriptions.

Symbols/Notations	Descriptions
$R = \frac{\mathbb{Z}}{<f(x)>}$	Finite Degree Ring
$R_q = \frac{\mathbb{Z}_q}{<f(x)>}$	Finite Ring
$B \in \mathbb{Z}^{M \times M}$	Square Matrix of Order M
$Cha(.)$	Characteristic Function
$Mod_2(.,.)$	Modular Function
$\|\|.\|\|_2$	Euclidean Norm
U_i	*ith* User
S_j	*jth* Server
q_{H_1}, q_{H_2}	Oracle Queries
$\mathcal{A}$	Adversary
ROM	Random Oracle Model
q	Prime Number
$\mathcal{D}_\beta^n$	n-Dimension Gaussian
sk	Session Key

6. Statement and Cryptanalysis of Moony et al.'s Protocol [13]

A forward secure mutual authentication protocol ensures that even if long-term private keys are compromised, previously established session keys or authentication transcripts

still remain secure. This is an important property in secure communications, especially in long-term deployments such as IoT, healthcare, or vehicular networks. In the reference of mutual authentication protocols, forward secrecy can be maintained by ensuring that session keys are not computed directly from long-term private keys of the server. We have tried to explain how the authentication protocol [13] is not maintained with forward secrecy. In the protocol in [13], the user sends PID_i, X_i, W_i, $\sum_i$ to the server over a public networking channel. Therefore, if we assume the server's master secret key is compromised, then the probabilistic polynomial times adversary easily computes any of the older session keys by executing these steps: (1) The adversary takes information PID_i, X_i, W_i, $\sum_i$ from the public channel and master secret s, then computes $t_i' = X_i.s$, $K_i' = \psi_2(t_i', W_i)$. Further, it computes real identity $iden_i' = PID_i \oplus H(K_i', X_i, W_i)$, $B_i' = H(iden_i', s)$, and finally computes session key $Sk_s = H(iden_i, PID_i, K_i', B_i')$.

6.1. Key Mismatch Attack on Moony et al.'s Scheme

To execute a key mismatch attack on the Moony et al. protocol [13], an adversary $\mathcal{A}$ initiates different sessions with the server to obtain information and recover the master secret s of the server. A brief overview of the protocol [13] is given in Table 3. To execute this attack, $\mathcal{A}$ sets public key x_i as below:

1. $\mathcal{A}$ computes key:
$$x_i = \alpha \cdot f_i + 2g_i,$$
where $f_i, g_i \leftarrow \mathcal{D}_\xi$ are random samples.
2. $\mathcal{A}$ fixes $f_i = \mathbf{0}$ and $g_i = \mathbf{0}$ for i, except
$$f_i[m-1-i] = 1, \quad g_i[m-1-j] = r,$$
where $r \in \mathbb{Z}_p$ is chosen by $\mathcal{A}$.
3. $\mathcal{A}$ computes the value j satisfying $s[j] = \pm 1$ (as described in Section 5.4 of [13]).
4. Then $\mathcal{A}$ computes:
$$k_i = f_i \cdot P = 0, \quad \text{since } f_i = \mathbf{0}$$
5. The characteristic function outputs:
$$\omega_i = \mathrm{Ch}(k_i) = \mathrm{Ch}(0) = 0$$
6. The modular reconciliation function outputs:
$$t_i = M_2(k_i, \omega_i) = M_2(0, 0) = 0$$
7. $\mathcal{A}$ mounts attack as below:
   (a) An honest $\mathcal{A}$ conducts the procedure, purposely switching $\omega_i[m-1]$ to 1 such that $t_i[m-1]$ can assist and infer $s[i]$ for some $i \in [0, m-1]$.
   (b) The server S receives (x_i, ω_i) from $\mathcal{A}$ along with the necessary parameters.
8. After receiving (x_i, ω_i), the server computes:
$$k_i' = x_i \cdot s = (\alpha f_i + 2g_i) \cdot s \tag{13}$$
$$\begin{aligned} k_i'[m-1] &= x_i \cdot s[m-1] = 2g_i \cdot s[m-1] \\ &= 2s[i] + 2r \end{aligned} \tag{14}$$

Since $\mathcal{A}$ changes $\omega_i[m-1]=1$, then the reconciliation function outputs:

$$\begin{aligned} t_i'[m-1] &= M_2(k_i', \omega_i)[m-1] \\ &= M_2(k_i'[m-1], \omega_i[m-1]) \\ &= \left(k_i'[m-1] + \frac{p-1}{2}\right) \bmod p \bmod 2 \\ &= \left(2s[i] + 2r + \frac{p-1}{2}\right) \bmod p \bmod 2 \end{aligned} \tag{15}$$

9. $\mathcal{A}$ constructs accessible $\mathcal{O}$ that observes the server S. The oracle $\mathcal{O}$ executes the following steps:
   - Gets $(\text{pid}_i, \omega_i, x_i, \beta_i)$
   - Computes $k_i' = x_i \cdot s$
   - Computes $t_i' = M_2(k_i', \omega_i)$
   - Computes masked value:

$$\text{id}_a' = \text{pid}_i \oplus H(t_i', x_i, \omega_i)$$

   - Computes:

$$V_i' = H(\text{id}_a', V_i, t_i'), \quad \beta_i' = H(\text{id}_a', V_i, t_i')$$

   - Returns:

$$\mathcal{O}(\text{pid}_i, \omega_i, x_i, \beta_i) = \begin{cases} 1, & \text{if } \beta_i = \beta_i' \\ 0, & \text{Else} \end{cases}$$

10. Since $r=0$ is set by $\mathcal{A}$, $x_i = 2g_i$. The result of querying x_i to the oracle $\mathcal{O}$ with $\mathcal{A}$ is:

$$k_i'[m-1] = 2s[i] + 2r$$

    from Equation. The sign of $k_i'[m-1]$ is the outcome that imitates $s[i]$'s sign. Therefore:
    - If $\mathcal{O}$ outputs 1, $\mathcal{A}$ concludes $\text{sign}(s[i]) < 0$.
    - If $\mathcal{O}$ outputs 0, $\mathcal{A}$ concludes $\text{sign}(s[i]) > 0$.

11. The output is 1 when $s[i]=0$; then $\mathcal{A}$ executes another query with $g_i[m-1-i] = -1$:

$$k_i'[m-1] = -2s[i]$$

    If $g_i[m-1-i] = 1$ and -1, then it returns 1, and $\mathcal{A}$ outputs $s[i]=0$.

12. To get the sign:
    - $\mathcal{A}$ takes the help of $g_i[m-1-j] = r$ to check negative $s[i]$.
    - $\mathcal{A}$ takes the help of $g_i[m-1-j] = -r$ to check positive $s[i]$.

    From the above computations for k, if $t_i'[m-1]=0$, then:

$$s[i] + k \equiv 0 \pmod{p} \quad \Rightarrow \quad s[i] \equiv -k \pmod{p}$$

    Thus, $\mathcal{A}$ guesses the correct $s[i]$, and recovers master secret s.

Table 3. Prototype [13] susceptible to key mismatch attack.

User	Server
Public Key: $P_U = \alpha \cdot s_U + 2e_U$ Secret Key: $s_U \in \mathbb{R}_p$ where $s_U, e_U \leftarrow \chi_\xi$	Public Key: $P_S = \alpha \cdot s_S + 2e_S$ Secret Key: $s_S \in \mathbb{R}_p$ where $s_S, e_S \leftarrow \chi_\rho$
$\xrightarrow{P_U}$	
$k_S = P_U \cdot s_S$ $\omega_S = \mathrm{Ch}(k_S)$	
$\xleftarrow{P_S, \omega_S}$	
$k_U = P_S \cdot s_U$ $\mathrm{sk}_S = \sigma_S = M_2(k_S, \omega_S)$ $\mathrm{sk}_U = \sigma_U = M_2(k_U, \omega_S)$	

6.2. Unlinkability in Mishra et al.'s Scheme [29]

Mishra et al. [29] proposed a quantum enhanced authentication and key agreement protocol for autonomous vehicles, but this protocol does not support unlinkability. In this scheme, the vehicle user sends the message $< PID_i, x_i, \Sigma_i >$ which contains no repeating PID_i, and it helps to link older messages with the new message coming.

6.3. Unlinkability and Problem with Error Distribution in Pursharthi et al. [15]

Pursharthi et al. [15] proposed a quantum-safe authenticated key agreement scheme for mobile devices, but their protocol does not provide unlinkability. One can observe that user U_i sends information $\{PID_i,\ x_i,\ T_i,\ \sum_i\}$ to the server over a public channel. In the message $\{PID_i,\ x_i,\ T_i,\ \sum_i\}$, the value PID_i is chosen by the server, and it is a fixed value for each of the sessions; therefore, this protocol cannot provide unlinkability. Moreover, this protocol uses learning with error multiple times on the same component $k_j = (x_i.s_j + 2c)d + 2e'$ that will increase the error added from the Gaussian; therefore, this is mathematically incorrect. These findings underscore the urgent need for carefully designed lattice-based protocols that offer strong security properties such as unlinkability, key control, and resilience against side-channel and guessing attacks while maintaining correctness and efficiency under practical constraints.

7. Proposed Two-Party Key Establishment and Key Transport Protocols for Mobile Devices

7.1. Setup Phase

The Setup Phase for Registration establishes the foundation for secure communication between users, devices, or entities in a cryptographic system. This phase is typically executed once by a trusted authority (TA) to initialize system parameters and register legitimate users or devices.

1. The website server (S_j) selects three security parameters: $n = 2^i$ is an integer, a prime q of $2n + 1$ type, and n-dimensional discrete Gaussian $\mathcal{D}_\delta^n$ over finite ring R_q with standard deviation δ.
2. The server S_j chooses $a \leftarrow R_q$, and it samples $s,\ e \leftarrow \mathcal{D}_\delta^n$ randomly and computes public key $b = a.s + 2e$; here s is the master secret of the server.
3. The server S_j also chooses a secure collision resistant hashing function $\mathcal{H}$ that outputs 256 bits, publishes $n, q, \mathcal{D}_\beta^n, a, b, \mathcal{H}(.)$ in the public domain, and keeps "s" secret.

7.2. Registration Phase

The Registration Phase is responsible for securely enrolling a user or device into the system. This phase ensures that only legitimate and authorized participants can obtain valid credentials from the trusted authority (TA), enabling secure authentication, communication, and cryptographic operations.

1. The device user U_i chooses arbitrary strings as identity (id_i), password (pw_i), and random integer $\varrho_i \in Z_q$ and computes $\vartheta_i = \mathcal{H}(id_i||\mathcal{H}(pw_i||\varrho_i))$. Further, U_i sends $\vartheta_i,\ id_i$ to the server S_j.
2. The server receives $\{\vartheta_i,\ id_i\}$, computes masked identity $psid_i = \mathcal{H}(id_i||s)$, and $d_i = psid_i \oplus \vartheta_i$, and sends the message ϑ_i, d_i to the user (U_i).
3. The user U_i receives ϑ_i, d_i from the server S_j, computes $\vartheta_i = \mathcal{H}(id_i||\mathcal{H}(pw_i||\varrho_i))$, and recovers masked identity $d_i \oplus \vartheta_i = psid_i$, and a verification factor $v_i = \mathcal{H}(id_i||pw_i||psid_i||\varrho_i)$. Finally, the user ($U_i$) computes the verification value $\varrho_i^* = \varrho_i \oplus \mathcal{H}(\mathcal{H}(id_i||\mathcal{H}(pw_i))\ mod(p))$, where p is a suitable prime number, and stores $\{\varrho_i^*,\ v_i,\ \mathcal{H}(.)\}$ in the device.

7.3. Login and Authenticated Key Establishment Phase (Figure 1)

Authenticated Key Establishment (AKE) protocols enable two or more parties to mutually derive a shared secret key over an insecure channel, ensuring that the key is known only to the legitimate participants. Unlike unauthenticated key exchange mechanisms that are susceptible to man-in-the-middle (MitM) attacks, AKE protocols incorporate authentication mechanisms such as digital signatures, certificates, or message authentication codes (MACs) to confirm the identity of communicating parties. These protocols are widely used in internet security protocols such as TLS, IPsec, and SSH, and are especially important in scenarios requiring forward secrecy, where compromise of long-term keys does not affect past session keys. With the advent of quantum computing, classical AKE protocols based on discrete logarithms and elliptic curves are being reconsidered in favor of post-quantum AKE schemes, particularly those based on lattice problems, code-based, or hash-based cryptography.

1. The user U_i inputs id_i, pw_i, and computes $\varrho_i = \varrho_i^* \oplus \mathcal{H}(\mathcal{H}(id_i||\mathcal{H}(pw_i))\ mod(p))$, $\vartheta_i = \mathcal{H}(id_i||\mathcal{H}(pw_i||\varrho_i))$, and masked identity $psid_i = \mathcal{H}(id_i||s) = d_i \oplus \vartheta_i$, checks $v_i =?\mathcal{H}(id_i||pw_i||psid_i||\varrho_i)$, and gives permission to be logged into the device.
2. The U_i samples random $r_i,\ f_i \leftarrow \mathcal{D}_\beta^n$ from the n-dimensional Gaussian, and computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.b$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i,\ w_i)$, and a verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||psid_i)$. Finally, the user U_i sends $\{x_i,\ w_i,\ id_i,\ \alpha_i\}$ to the server over a public channel.
3. The server S_j receives the message $\{x_i,\ w_i,\ id_i,\ \alpha_i\}$, samples $r_s,\ f_s \leftarrow \mathcal{D}_\delta^n$, and computes $x_s = a.r_s + 2.f_s$, $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i',\ w_i)$, and the masked identity $psid_i' = \mathcal{H}(id_i'||s)$. Furthers, it verifies $\alpha_i = \mathcal{H}(x_i||\sigma_i'||psid_i')$, and computes $\kappa_s = x_i.r_s$, $w_s = Cha(\kappa_s)$, $\sigma_s = Mod_2(\kappa_s,\ w_s)$, and session key $sk = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s||id_i||id_s||psid_i)$. Finally, it computes a verification factor $\alpha_s = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s||psid_i||sk)$, and sends the message $\{x_s,\ w_s,\ \alpha_s\}$ to the corresponding user.
4. The U_i receives the message $\{x_s,\ w_s,\ \alpha_s\}$, and computes $\kappa_s' = r_i.x_s$, $\sigma_s' = Mod_2(\kappa_s,\ w_s)$, session key $sk' = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s'||id_i||id_s||psid_i)$. Finally, the user verifies the correctness of the session key by $\alpha_s = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s'||psid_i||sk')$.

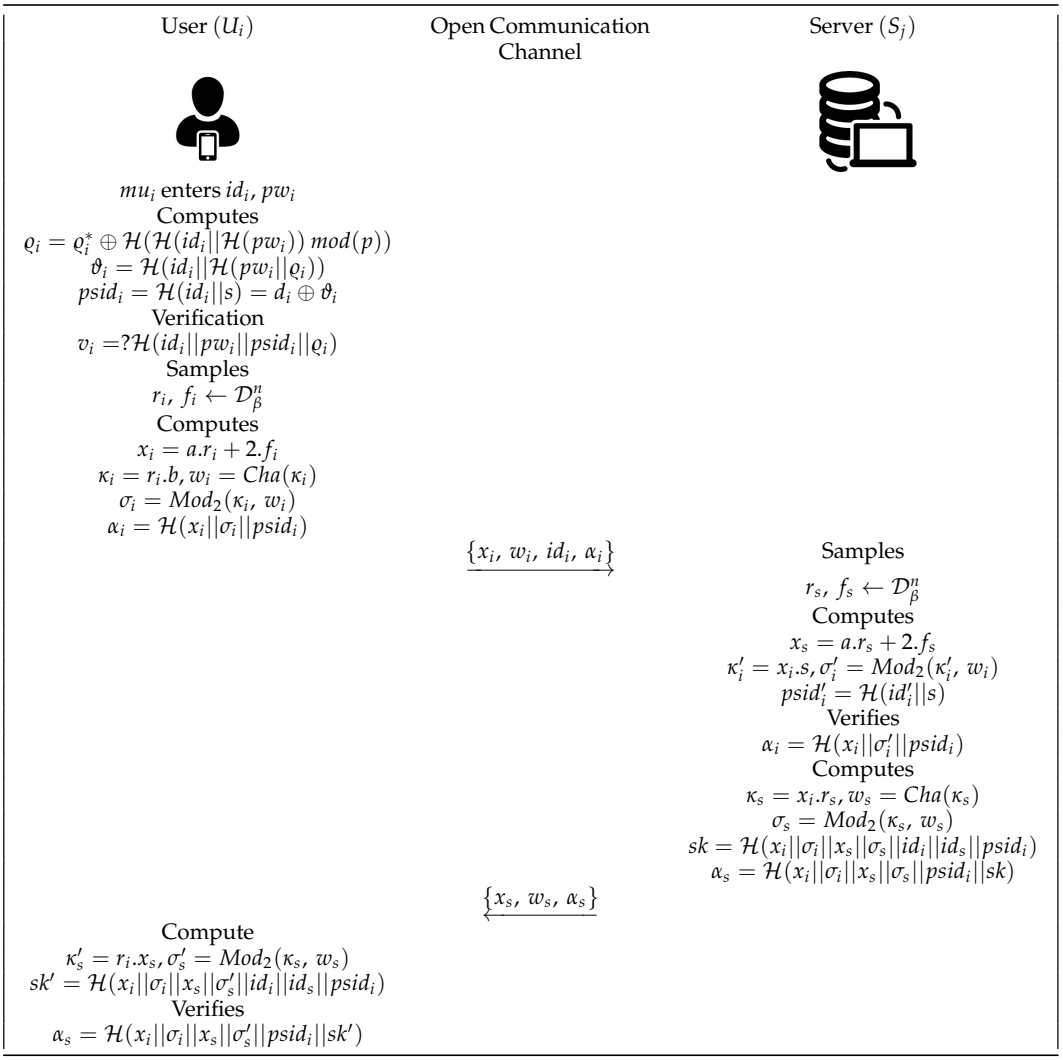

Figure 1. Mutual authentication and key establishment protocol overview.

7.4. Login and Authenticated Key Transport Phase (Figure 2)

1. The user U_i inputs id_i, pw_i, and computes $\varrho_i = \varrho_i^* \oplus \mathcal{H}(\mathcal{H}(id_i||\mathcal{H}(pw_i))\ mod(p))$, a masked value $\vartheta_i = \mathcal{H}(id_i||\mathcal{H}(pw_i||\varrho_i))$, masked identity $psid_i = \mathcal{H}(id_i||s) = d_i \oplus \vartheta_i$, and it checks $v_i = ?\mathcal{H}(id_i||pw_i||psid_i||\varrho_i)$, and gives permission to be logged into the device (see Figure 2).
2. The user chooses random sample $r_i,\ f_i \leftarrow \mathcal{D}_\delta^n$, and computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.b$, $w_i = Cha(\kappa_i)$, and $\sigma_i = Mod_2(\kappa_i,\ w_i)$. Now, the user computes masked dynamic identity $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$, $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i,\ w_i,\ aid_i,\ \alpha_i\}$ to the server.
3. The server receives the information $\{x_i,\ w_i,\ aid_i,\ \alpha_i\}$ from U_i, and computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i',\ w_i)$, $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i')$, and $psid_i' = \mathcal{H}(id_i'||s)$. Now, the server verifies whether $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$ holds or not. If the information received is correct, then the server computes and chooses the random session key $sk \in \{0,\ 1\}^{256}$, encrypts it to obtain $sk^* = e_\varrho(sk)$, and computes $\alpha_s = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s||psid_i||sk)$, then sends $\{sk^*,\ \alpha_s\}$ to the user.
4. The U_i receives the information $\{sk^*,\ \alpha_s\}$ from the server, decrypts $d_\varrho(sk^*) = sk'$, and obtains the session key. Finally, the user verifies the session key with $\alpha_s = \mathcal{H}(x_i||\sigma_i||x_s||\sigma_s'||psid_i||sk')$, and stores the session key for current communication.

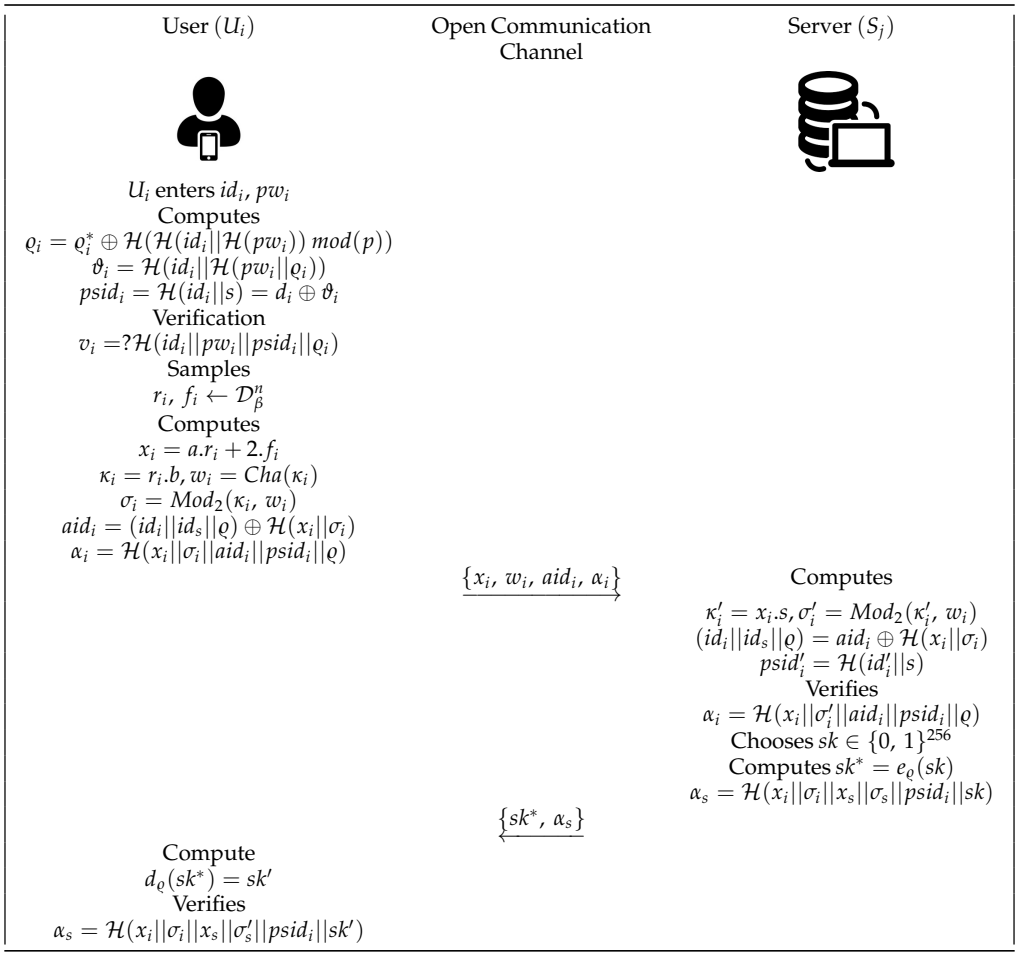

Figure 2. Key transport protocol overview.

7.5. Proof of Correctness

To provide a mathematical proof of correctness for an authentication protocol, we aim to formally show that if all parties follow the protocol honestly, and no adversary interferes, then the authentication between entities succeeds correctly, i.e., the intended parties are authenticated and session keys match.

Theorem 1. *If both user (U_i) and server (S_j) shared a common session key sk, then we need to show that for user (U_i) and server (S_j), if $|k_i - k_i'| < \frac{q}{8}$, then mathematical correctness holds.*

Proof. If we can show that $|k_i - k_i'| < \frac{q}{8}$, then mathematical correctness will hold obviously. For this reason, a researcher computes the error difference

$$\begin{aligned} |k_i - k_i'| &= |r_i.b - x_i.s| \\ &= |r_i(as + 2e) - (ar_i + 2f_i)s| \\ &= |2e \cdot r_i - 2s \cdot f_i| \end{aligned}$$

and

$$\begin{aligned}\|2e \cdot r_i - 2s \cdot f_i\|_2 &\leq \|e \cdot r_i\|_2 + \|s \cdot f_i\|_2 \\ &\leq \sqrt{m}\|r_i\|_2\|e\|_2 + \sqrt{m}\|f_i\|_2\|s\|_2 \\ &< 2\sqrt{m}\gamma \cdot \sqrt{m}\gamma \cdot \sqrt{m} \\ &= 2m^{3/2}\gamma^2 < \frac{q}{8}\end{aligned}$$

Hence, the result follows Definition 5.

□

8. Informal Security Analysis

An informal security analysis helps identify potential vulnerabilities in a two-party key establishment protocol without going into deep formal proofs. Below is a sample informal security analysis of such a protocol, typically involving parties A and B, aiming to establish a shared session key sK over an insecure channel.

1. **Anonymity**: An anonymous protocol is a type of communication protocol designed to protect the identity of the participants involved. These protocols are widely used in privacy-preserving applications such as secure messaging, anonymous voting, electronic cash systems, and privacy-preserving authentication in networking systems. In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks the real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. Since the real identity is masked during communication over a public channel, it is hard for the adversary ($\mathcal{A}$) to guess the real identity of the user.
2. **Session key freshness**: Session key freshness refers to the guarantee that a newly generated session key is unique and has not been used in any previous communication session. It ensures that each session between parties uses a fresh (i.e., previously unused and unpredictable) key, which is crucial for maintaining confidentiality, forward secrecy, and resistance to replay attacks. In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks the real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. The server receives the message $\{x_i, w_i, aid_i, \alpha_i\}$, computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i', w_i)$, and recovers real identities $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i)$. Further, it computes $psid_i' = \mathcal{H}(id_i'||s)$, and verifies the information received from the public channel by $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$. Finally, the server chooses a random session key $sk \in \{0, 1\}^{256}$, and transports it to the corresponding user. Since the session key is chosen randomly, and is new for each session, it concludes the freshness.
3. **Forward Secure**: A forward secure protocol (also called forward secrecy or perfect forward secrecy) is a cryptographic protocol designed to ensure that compromise of long-term secret keys does not compromise past session keys. In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. The server receives the message $\{x_i, w_i, aid_i, \alpha_i\}$, computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i', w_i)$, and recovers real identities $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i)$. Further, it computes $psid_i' = \mathcal{H}(id_i'||s)$, and verifies the information received from the public channel by $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$. Finally, the server chooses a random

session key $sk \in \{0, 1\}^{256}$, and transports it to the corresponding user. The session key is dependent on the particular session and is different for other sessions. Therefore, it is not possible to recover previous session keys even if the current session key is compromised.

4. **Replay Attack**: A replay attack occurs when an adversary intercepts and retransmits a previously sent message to trick the receiver into performing an action again or unauthorizedly, without knowing the actual message content or context. An adversary records a valid message exchange between two parties and replays the same message later to impersonate a legitimate user or repeat an operation. In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks the real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$ along with a random number. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. The server receives the message $\{x_i, w_i, aid_i, \alpha_i\}$, computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i', w_i)$, and recovers real identities $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i)$. Further, it computes $psid_i' = \mathcal{H}(id_i'||s)$, and verifies the information received from the public channel by $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$. Finally, the server chooses a random session key $sk \in \{0, 1\}^{256}$, and encrypts it with a random number sent by the user. Therefore, it is not possible to replay older messages.
5. **Impersonation Attack**: An impersonation attack occurs when an adversary pretends to be a legitimate user or entity in a communication protocol to deceive another party, without actually possessing the user's credentials or secrets. In an impersonation attack, the attacker mimics the identity of a legitimate party (e.g., Alice) and communicates with another party (e.g., Bob) to gain unauthorized access, establish a session, or steal sensitive data. In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks the real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. The server receives the message $\{x_i, w_i, aid_i, \alpha_i\}$, computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i', w_i)$, and recovers real identities $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i)$. Further, it computes $psid_i' = \mathcal{H}(id_i'||s)$, and inserts it in the verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$. Thus, it is not possible to impersonate the user or server.
6. **Man-in-the-Middle (MITM) attack**: A protocol is said to be resistant to a Man-in-the-Middle (MITM) attack if it can detect and prevent an adversary from secretly intercepting and altering the communication between two parties without being detected. In an MITM attack, an adversary (Eve) places herself between two parties (Alice and Bob) and modifies, drops, or replaces their messages, often establishing two separate connections:
   In this protocol, the user U_i samples random r_i, $f_i \leftarrow \mathcal{D}_\delta^n$, computes $x_i = a.r_i + 2.f_i$, $\kappa_i = r_i.\xi$, $w_i = Cha(\kappa_i)$, $\sigma_i = Mod_2(\kappa_i, w_i)$, and masks the real identity as $aid_i = (id_i||id_s||\varrho) \oplus \mathcal{H}(x_i||\sigma_i)$. Finally, U_i computes verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i||aid_i||psid_i||\varrho)$, and sends $\{x_i, w_i, aid_i, \alpha_i\}$ to the S_j. The server receives the message $\{x_i, w_i, aid_i, \alpha_i\}$, computes $\kappa_i' = x_i.s$, $\sigma_i' = Mod_2(\kappa_i', w_i)$, and recovers real identities $(id_i||id_s||\varrho) = aid_i \oplus \mathcal{H}(x_i||\sigma_i)$. Further, it computes $psid_i' = \mathcal{H}(id_i'||s)$, and inserts it in the verification factor $\alpha_i = \mathcal{H}(x_i||\sigma_i'||aid_i||psid_i||\varrho)$. Finally, the server chooses a random session key $sk \in \{0, 1\}^{256}$, and encrypts it with a random number sent by the user. Therefore, a man-in-the-middle attack is not possible.

7. **Authenticated key establishment and key transport schemes resist key mismatch attacks**: The U_i generates

$$x_i = \alpha f_i + 2g_i \quad \text{with} \quad f_i = 0 \in R_q$$

and transmits to the server S_j. Now, if an adversary $\mathcal{A}_i$ has information about the protocol and the ability to submit polynomial times queries to $\mathcal{A}_i$, then it is easy to execute **key mismatch attacks** to guess the secret key s [13].
To get jth coefficient $s[j]$, the adversary $\mathcal{A}_i$ chooses g_i satisfying:

$$g_i[j] = 0 \quad \forall j = 0 \text{ to } n-1 \quad \text{except} \quad j = n-1-j,\ n-1-\eta$$

$$g_i[n-1-j] = 1, \quad g_i[n-1-\eta] = k$$

Then, $\mathcal{A}_i$ runs this protocol and tries to flip the bit $(n-1)$ of ω_i and send it to the server. It is essential to choose η satisfying $s[\eta] = \pm 1$. Further, $\mathcal{A}_i$ guesses the sign of $s[j]$, and submits queries. If the sign received is positive, then

$$k_i[n-1] = 2s[j] - 2k$$

It is interesting to see that this value flips from positive to negative, increasing k whenever $k > s[j]$, and $Sk_s[n-1] = 1$ whenever $k_i[n-1] > 0$.
In this protocol, the multiplication of $s[j]$ is performed with random samples f_i and g_i to get $k_i[j-1]$, where $\mathcal{A}_i$ controls f_i only, and server S_j controls g_i. Thus, $\mathcal{A}_i$ has no control over changing g_i. Therefore,

$$k_i[n-1] = 2s[j]g + 2k_s[\eta]g_i + 2g_if_i + 2g_i = 2s[j]g_i + 2k_ig_i + 2f_ig_i + 2g_i$$

If $\mathcal{A}_i$ guesses the sign of $s[j]$ is positive, it increases $k > 0$ and observes changes from 1 to 0; this flip is not there if:

$$|2s[j]g_i + 2f_ig_i + 2g_i| \leq 2g_i$$

Due to this condition, the proposed protocol resists key mismatch attacks.

9. Formal Security Proof

This section illustrates a structured and academically appropriate Formal Security Proof outline for a lattice-based key agreement protocol based on the Ring Learning With Errors (RLWE) problem [2,30]. The proof follows the Bellare–Rogaway model under the random oracle model (ROM) and assumes the hardness of RLWE. We have provided a detailed formal security proof for a lattice-based key agreement protocol based on the Ring Learning With Errors (RLWE) assumption. The proof is constructed within the Bellare–Rogaway (BR) model and under the random oracle model (ROM) [31].

9.1. Security Model

The security of the proposed protocol is analyzed under the BR model, where multiple users may participate in many concurrent sessions. The adversary $\mathcal{A}$ is modeled as a probabilistic polynomial time (PPT) algorithm capable of issuing the following queries [31]:

- **Send**(U_i, m) — Sends a message m to user U_i and receives the response.
- **Reveal**(s) — Reveals the session key of completed session s.
- **Corrupt**(U_i) — Reveals the long-term secret of user U_i.
- **Test**(s^*) — The challenge query. A random bit b is chosen. If $b = 0$, the real session key is returned; if $b = 1$, a random string is returned. $\mathcal{A}$ outputs a guess b'.

9.2. Assumptions

- **Hardness of RLWE:** Let $R = \mathbb{Z}_q[x]/(f(x))$ be a cyclotomic ring. Given samples $(a, a \cdot s + e)$ for uniformly random $a \in R_q$, secret $s \in R_q$, and error e sampled from a discrete Gaussian, no PPT adversary can distinguish these from uniform with non-negligible advantage.
- **Hash functions** used in the protocol are modeled as random oracles.

9.3. Game-Based Proof

We define a sequence of games, G_0, G_1, G_2, G_3, where G_0 represents the real attack scenario and G_3 is the ideal game with zero advantage.

9.3.1. Game G_0

This is the real execution of the protocol. The adversary interacts with the honest parties and eventually issues a Test query.

$$\Pr[\mathcal{A} \text{ wins in } G_0] = \text{Adv}_{\mathcal{A}}^{\text{AKE}}(\lambda)$$

9.3.2. Game G_1

In this game, the session key is no longer derived from the actual shared secret but from a randomly chosen value. Since the key derivation uses a random oracle, the output is indistinguishable.

$$|\Pr[G_0] - \Pr[G_1]| \leq \text{negl}(\lambda)$$

9.3.3. Game G_2

Here, the RLWE shared secret is replaced with a uniform random value in R_q. By the hardness of RLWE, the adversary cannot distinguish this change:

$$|\Pr[G_1] - \Pr[G_2]| \leq \epsilon_{\text{RLWE}}(\lambda)$$

9.3.4. Game G_3

In this ideal game, the session key is uniformly random and independent of the adversary's view. Therefore, the adversary's success probability is exactly $1/2$.

$$\Pr[G_3] = \frac{1}{2}$$

9.4. Advantage Bound

By the hybrid argument, we have:

$$\text{Adv}_{\mathcal{A}}^{\text{AKE}}(\lambda) = |\Pr[G_0] - \Pr[G_3]| \leq \epsilon_{\text{RLWE}}(\lambda) + \text{negl}(\lambda)$$

Thus, the adversary's advantage is negligible, proving the protocol is secure under the RLWE assumption.

9.4.1. Game 0: Real Protocol Execution

This game represents the real protocol. The adversary $\mathcal{E}$ interacts with the protocol participants and receives the real shared secret k derived using the parties' private and public keys. The adversary's advantage in distinguishing k from a random string is denoted as:

$$\text{Adv}_{\mathcal{E}}^{\text{Game 0}} = \Pr[\mathcal{E} \text{ distinguishes } k \text{ from random}] - \frac{1}{2}.$$

9.4.2. Game 1: Replace Secret with a Random Value

In this game, the shared secret k is replaced with a random string r of the same length. The adversary interacts with the protocol as before. The change from Game 0 to Game 1 is indistinguishable if the shared secret k is computationally secure.

Reduction: Assume there exists a PPT adversary $\mathcal{E}$ that distinguishes Game 0 from Game 1 with non-negligible advantage ϵ. We construct a reduction $\mathcal{R}$ that uses $\mathcal{E}$ to solve the underlying post-quantum problem:

1. $\mathcal{R}$ is given a challenge instance of the hard problem.
2. $\mathcal{R}$ embeds the challenge into the public key pk_A or pk_B.
3. $\mathcal{R}$ simulates the protocol for $\mathcal{E}$ and uses $\mathcal{E}$'s output to solve the hard problem.

If $\mathcal{E}$ has non-negligible advantage ϵ, then $\mathcal{R}$ solves the hard problem with non-negligible probability, contradicting the hardness assumption.

Thus:

$$\mathrm{Adv}_{\mathcal{E}}^{\text{Game 1}} = \mathrm{Adv}_{\mathcal{E}}^{\text{Game 0}} - \epsilon.$$

9.4.3. Game 2: Random Oracle Simulation

In this game, the hash function is replaced with a random oracle. The adversary cannot distinguish this change due to the random oracle model assumption. Thus:

$$\mathrm{Adv}_{\mathcal{E}}^{\text{Game 2}} = \mathrm{Adv}_{\mathcal{E}}^{\text{Game 1}}.$$

9.5. Conclusions

In Game 2, the shared secret is computationally indistinguishable from a random value. Therefore:

$$\mathrm{Adv}_{\mathcal{E}}^{\text{Game 2}} = 0.$$

Combining the results of all games:

$$\mathrm{Adv}_{\mathcal{E}}^{\text{Game 0}} \leq \epsilon.$$

Since ϵ is negligible, the protocol is secure in the ROR model.

Theorem 2. *Let $Adv^{A}_{Proposed}(t)$ represent the probability that A successfully compromises the semantic security of the session key and achieves mutual authentication between U_i and U_j within a polynomial time bound t in the proposed protocol. Suppose that $Adv^{A}_{RLWE}(t')$ denotes the advantage that A has in solving an RLWE problem instance within a polynomial time limit t'. The adversary A can make at most q_{H_i} (for $i = 1, 2, 3$), q_{se}, and q_{ex} queries to the hash function H_i, Send oracle, and Execute oracle, respectively. Let T_{smul} be the time needed for one component-wise multiplication with a scalar in $\mathbb{R}_q$. Then, the following inequality holds:*

$$Adv^{A}_{Proposed}(t) \leq \frac{q_{H_1}^2}{2l} + \frac{q_{H_2}^2}{2l} + \frac{(q_{ex} + q_{se})^2}{2^l} + \frac{q_{H_3}^2}{2^l} + (q_{ex} + q_{se}) \cdot Adv^{A}_{RLWE}(t'), \quad (16)$$

where

$$t' \leq t + (2q_{ex} + 4q_{sc} + 2q_{H_1} + 2q_{H_2} + 1) \cdot T_{smul}. \quad (17)$$

Proof. Suppose that an adversary A can compromise the semantic security of the session key in the proposed protocol. In such a scenario, a challenger C, executing a PPT algorithm, would be capable of solving an instance of the RLWE problem in $R_q \times R_q$. To demonstrate this, we consider a series of games, Game_i ($0 \leq i \leq 4$), where Game_0 represents the real attack [10]. Each Game_i is an interactive simulation between A and C, where C uses a PPT algorithm to correctly respond to oracle queries.

For each game, Game_i, an event X_i is defined such that X_i corresponds to A successfully breaching the semantic security of the session key in Game_i. Let an event E, independent of X_i, occur during the protocol simulation and be detectable by C. The games Game_i and Game_{i+1} are indistinguishable unless event E occurs. Therefore:

$$|\Pr[X_{i+1}] - \Pr[X_i]| \leq \Pr[E]. \tag{19}$$

Game 0: This game represents the real attack on the LB-2PAKA protocol in the random oracle model (ROM). In this simulation, A interacts with the protocol oracles as in an actual execution, and the users $P \in U$ execute the protocol as the real execution of the LB-2PAKA protocol in the ROM. Thus:

$$\text{Adv}^A_{\text{Proposed}}(t) = \left|\Pr[X_0] - \tfrac{1}{2}\right|. \tag{20}$$

Game 1: This game differs from Game_0 only in how hash queries are handled. C maintains initially empty hash lists H^j_{list} $(j = 1, 2, 3)$ for tuples (x, y), where x is the hash input, and y is the output. If A queries a hash with input x, C checks H^j_{list}. If x exists, C returns the corresponding y; otherwise, C selects $y \in \mathbb{Z}_q$ randomly, updates H^j_{list}, and returns y. The simulations of other oracles remain identical to those in Game_0. Hence:

$$\Pr[X_1] = \Pr[X_0]. \tag{21}$$

Game 2: This game differs from Game_1 by aborting if collisions are detected in the simulation of $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$, $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$, and $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$. By the birthday paradox, the probabilities of collisions in the outputs of oracles H_1, H_2, and H_3 are at most $\frac{q^2_{H_1}}{2l}$, $\frac{q^2_{H_2}}{2l}$, and $\frac{(q_{\text{sc}}+q_{\text{ex}})^2}{2^l}$, respectively. Thus:

$$|\Pr[X_2] - \Pr[X_1]| \leq \frac{q^2_{H_1}}{2l} + \frac{q^2_{H_2}}{2l} + \frac{(q_{\text{se}} + q_{\text{ex}})^2}{2^l}. \tag{22}$$

Game 3: In this game, A queries H_3 with inputs of the form $(\text{sid}, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$. The probability of correctly guessing the bit b chosen in the Test query of oracle H_3 is at most $\frac{q^2_{H_3}}{2^l}$. Thus:

$$|\Pr[X_3] - \Pr[X_2]| \leq \frac{q^2_{H_3}}{2^l}. \tag{23}$$

Game 4: This game considers offline guessing attacks, where A attempts to compute the session key SK without interacting with H_3. According to the protocol,

$$SK = H_3(\text{sid}, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2}),$$

where

$$\begin{aligned} \text{sid} &= (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1}), \\ \sigma_{i_1} &= \text{Mod}_2(t_{i_1}, w_{i_1}), \quad \sigma'_{j_1} = \text{Mod}_2(t_{j_1}, w_{j_1}), \\ \sigma_{i_2} &= \text{Mod}_2(t_{i_2}, w_{i_2}), \quad \sigma'_{j_2} = \text{Mod}_2(t_{j_2}, w_{j_2}). \end{aligned}$$

If A can successfully predict SK, they solve an RLWE instance in time t'. Therefore:

$$|\Pr[X_4] - \Pr[X_3]| \leq (q_{\text{ex}} + q_{\text{se}}) \cdot \text{Adv}^A_{\text{RLWE}}(t'). \tag{24}$$

If A fails to simulate H_3 correctly, they gain no advantage in distinguishing the real and random session keys:

$$\Pr[X_4] = \tfrac{1}{2}. \tag{25}$$

From the above equations, we have:

$$\begin{aligned} \mathrm{Adv}^{A}_{\mathrm{Proposed}}(t) &= \left|\Pr[X_0] - \tfrac{1}{2}\right| = |\Pr[X_0] - \Pr[X_4]| \\ &\leq \frac{q^2_{H_1}}{2l} + \frac{q^2_{H_2}}{2l} + \frac{(q_{\mathrm{ex}} + q_{\mathrm{se}})^2}{2^l} + \frac{q^2_{H_3}}{2^l} + (q_{\mathrm{ex}} + q_{\mathrm{se}}) \cdot \mathrm{Adv}^{A}_{\mathrm{RLWE}}(t'). \end{aligned} \tag{26}$$

□

10. Performance Analysis

The efficiency and practicality of the proposed key agreement protocol are crucial for its adoption in real-world post-quantum secure applications. Performance is analyzed in terms of computational cost, communication overhead, and implementation feasibility, particularly on resource-constrained devices such as mobile platforms. This section contains details of various components such as the processor, system type, operating system, and RAM used to analyze the performance of the proposed and relevant protocols. These attributes are very important to derive the computation cost of existing and proposed protocols. A comprehensive performance analysis of the proposed scheme with other relevant schemes is given in Table 4. We have considered an ideal-lattice, $n = 1024$ bits, Gaussian $log\ \delta = 17.01$, and a large random odd prime $q > 2$. This scheme was executed using C/C++ with multi-threading or parallel processing. This analysis has also taken help from some libraries, such as lattice Crypto and MIRACLE, where the server-side laptop's components are an Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz, system x64-based pc, Graphics, NVIDIA GeForce RTX 2060, RAM 8GB, Clock speed 1.00 GHz, Crypto Libraries Miracle, Charm-crypto, NumPy, hash lib, environment Python 3.9.0, Cores 8, and Operating System Linux (ubuntu) (Table 3). The user-side mobile phone's components are Android 13, MIUI 14, Android Version 13 TKQ1.221114.001, CPU Snapdragon 680, Octa-core Max 2.40GHz, Model 2201117TI, RAM 6.0+2.0 GB, Kernel Version, 4.19.157-perf-gcb1ffc010755, and Cores (8) (Table 5).

In the context of a cryptographic protocol, the notation of the execution cost typically refers to the computational resources required to execute the protocol, such as time, memory, and communication overhead. Here are some common notations used to denote the execution cost on the server side: $\tau_h \approx 13.4577$ ns for hashing, $\tau_{ecca} \approx 10$ ns for the cost of ECC point addition, $\tau_{ecpm} \approx 405$ ns for the cost of ECC point multiplication, $\tau_{Ge} \approx 72.50ns$, $\tau_{smul} \approx 0.2597$ ns, $\tau_{pmul} \approx 0.2908$ ns, $\tau_{pma} \approx 2.3455$ ns, $\tau_{cha} \approx 0.6455$ ns, and $\tau_{fe} \approx 405$ ns for Gaussian sampling, scalar multiplication under Q_q, component-wise multiplications under Q_q, component-wise multiplications and additions under Q_q, and computing the hashing.

Table 4. Device configuration.

Components	Details
Operating System	Android 13, MIUI 14
Android Version	13 TKQ1.221114.001
CPU	Snapdragon 680, Octa-core Max 2.40GHz
Model	2201117TI
RAM	6.0+2.0 GB
Kernel Version	4.19.157-perf-gcb1ffc010755
Cores	8

Table 5. Server configuration.

Components	Details
Processor	Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz
System Type	x64-based pc
Graphics	NVIDIA GeForce RTX 2060
RAM	8gb
Clock Speed	1.00 GHz
Crypto Libraries	Miracle, Charm-crypto, NumPy, hash lib
Environment	Python 3.9.0
Cores	8
Operating System	Linux(ubuntu)

Here are some common notations used to denote the execution costs on the user side: $\tau_h \approx 178.37$ ns for hashing, $\tau_{ecca} \approx 72$ ns for the cost of ECC point addition, $\tau_{ecpm} \approx 3363$ ns for the cost of ECC point multiplication, $\tau_{Ge} \approx 540.86$ ns, $\tau_{smul} \approx 6.3434$ ns, $\tau_{pmul} \approx 12.087ns$, $\tau_{pma} \approx 29.7645$ ns, $\tau_{cha} \approx 33.3456$ ns, and $\tau_{fe} \approx 3363$ ns for Gaussian sampling, scalar multiplication under Q_q, component-wise multiplications under Q_q, component-wise multiplications and additions under Q_q, and computing the hashing.

As shown in Table 6, we have analyzed that the scheme denoted $[A]$ [10] by Islam et al. takes an execution cost of $8\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 2824.0939$ on the user side, and $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme denoted by $[B]$ Rana et al. [24] takes an execution cost of $7\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 2822.0938$ on the user side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 219.0072$ on the server side. The scheme $[C]$ [32] by Dharminder et al. takes an execution cost of $8\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 2824.0939$ on the user side, and $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme denoted by $[D]$ Feng et al. [7] takes an execution cost of $6\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 2645.7243$ on the user side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme denoted by $[E]$ Dabra et al. [9] takes an execution cost of $8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 3597.1134$ on the user side, and $8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 337.546$ on the server side. The scheme denoted by $[F]$ Wang et al. [26] takes an execution cost of $8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 3597.1138$ on the user side, and $8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 337.546$ on the server side. Finally, the scheme denoted by $[G]$ [13] takes the execution cost of $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}2288.9851$ on the user side and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme denoted by $[H]$ [14] Seyhan et al. takes an execution cost of $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 2624.0939$ on the user-side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme $[I]$ [15] takes an execution cost of $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}2288.9851$ on the user side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The scheme $[J]$ [18] takes an execution cost of $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + 2\tau_{pma} + \tau_{smul} + 2\tau_{cha} + 2\tau_{mod}2288.9851$ on the user side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side. The proposed scheme denoted by $[K]$ (proposed scheme) takes an execution cost of $4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}2288.9851$ on the user side, and $5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod} \approx 220.0092$ on the server side (see Figure 3). To compute communication costs (see Figure 4), we have considered the password, and identified each of 128 bits, SHA2-256 bits, and 256 bits for reconciliation. Therefore, we have the communication costs for the proposed as $|x_i| + |w_i| + |aid_i| + |\alpha_i| + |x_s| + |w_s| + |\alpha_s| =$

1024 + 256 + 256 + 256 + 1024 + 256 + 256 = 3328 bits, and those of [*A*] Islam et al. [10], [*B*] Rana et al. [24] [*C*] Dharminder et al. [32], [*D*] Feng et al. [7] [*E*] Dabra et al. [9] [*F*] Wang et al. [26] [*G*] [13], [*H*] Seyhan et al. [14], [*I*] [15], [J] [18], and [*K*] (proposed scheme) are 3840, 3840, 3840, 3584, 3584, 3840, 3584, 3328, 3428, and 3328 bits, respectively.

Table 6. Proposed and relevant scheme's comparison for computation costs.

Schemes	User Side Computation Cost	Server Side Computation Cost
[*A*] [10] Islam et al.	$8\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*B*] [24] Rana et al.	$7\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*C*] [32] Dharminder et al.	$8\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*D*] [7] Feng et al.	$6\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*E*] [9] Dabra et al.	$8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*F*] [26] Wang et al.	$8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$8\tau_h + 3\tau_{Ge} + \tau_{pmul} + 3\tau_{pma} + 2\tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*G*] [13] Moony et al.	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*H*] [14] Seyhan et al.	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*I*] [15] Pursharthi et al.	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*J*] [18] Pursharthi et al.	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$
[*K*] Proposed scheme	$4\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$	$5\tau_h + 2\tau_{Ge} + 2\tau_{pmul} + \tau_{pma} + \tau_{smul} + \tau_{cha} + 2\tau_{mod}$

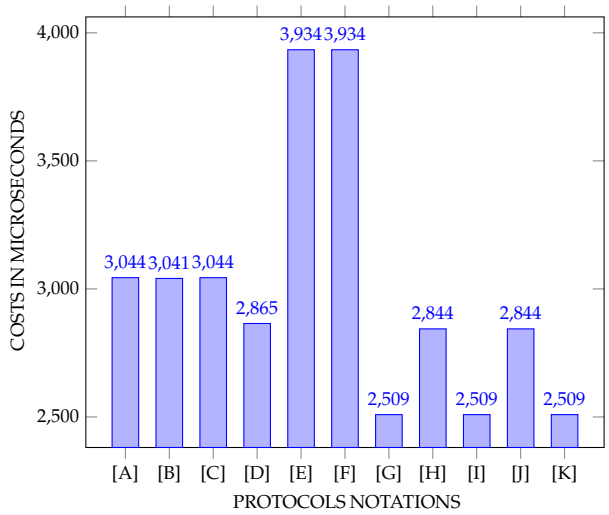

Figure 3. Illustration of computation costs of protocols.

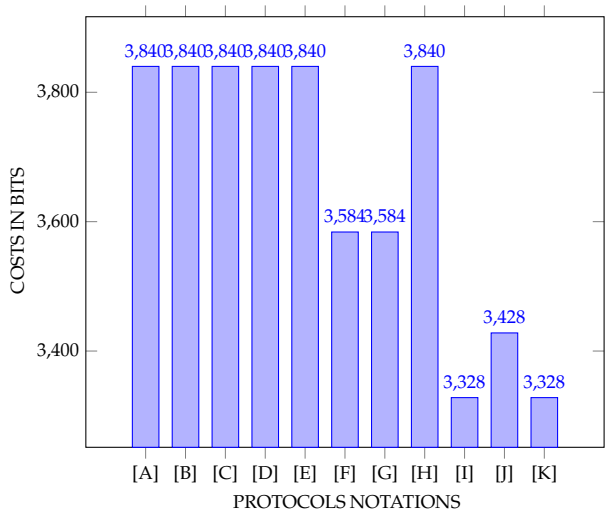

Figure 4. Illustration of communication costs of protocols.

11. Conclusions/Future Directions

In the modern landscape of cryptographic security, traditional protocols are increasingly inadequate in the face of emerging quantum threats. The advanced technology forces researchers to shift toward lattice-based cryptographic primitives, which offer strong post-quantum security based on hard mathematical assumptions such as Learning With Errors (LWE) and Short Integer Solution (SIS). This work contains a deep analysis and construction of two-party authenticated key establishment (AKE) and authenticated key transport protocols under lattice-based paradigm. These protocols enable secure and authenticated session key generation between two parties over a public network, while ensuring essential properties such as mutual authentication, forward secrecy, and resistance to man-in-the-middle and key compromise impersonation attacks. Additionally, this technique uses an error reconciliation mechanism and facilitates error-tolerant key agreement and secure key transport. In conclusion, lattice-based authenticated key agreement and transport serve a robust foundation for quantum-secure communication, and their efficiency makes them highly suitable for deployment to secure mobile communications, IoT networks, and mission-critical cyber–physical systems. Despite their robustness, lattice-based protocols still face challenges related to key size, computational overhead, and latency. Future work should target the design of optimized primitives and reconciliation techniques to support real-time communication in bandwidth-limited or low-power environments.

Author Contributions: M.R.: conception and design of study, acquisition of data, analysis and/or interpretation of data, writing—review and editing. D.C.: conception and design of study, writing—original draft. S.A.L. and C.-C.L.: conception and design of study, writing—original draft. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No data was used for the research described in the article. All authors approved the version of the manuscript to be published.

Acknowledgments: During the preparation of this work, the authors used Chat GPT in order to improve the use of English. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 20–22 November 1994, Santa Fe, NM, USA; IEEE: Piscataway, NJ, USA, 1994; pp. 124–134.
2. Regev, O. Lattice-based cryptography. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 131–141.
3. Lyubashevsky, V.; Peikert, C.; Regev, O. On ideal lattices and learning with errors over rings. In *Advances in Cryptology–EUROCRYPT 2010, Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, France, 30 May–3 June 2010*; Proceedings 29; Springer: Berlin/Heidelberg, Germany, 2010, pp. 1–23.
4. Mao, W.; Boyd, C. Towards formal analysis of security protocols. In Proceedings of the [1993] Proceedings Computer Security Foundations Workshop VI, Franconia, NH, USA, 15–17 June 1993; IEEE: Piscataway, NJ, USA, 1993; pp. 147–158.
5. Ding, J.; Xie, X.; Lin, X. A simple provably secure key exchange scheme based on the learning with errors problem. *Cryptol. Eprint Arch.* **2012**. Available online: https://eprint.iacr.org/2012/688.pdf (accessed on 23 September 2025).
6. Ding, J.; Alsayigh, S.; Saraswathy, R.; Fluhrer, S.; Lin, X. Leakage of signal function with reused keys in RLWE key exchange. In Proceedings of the 2017 IEEE international conference on communications (ICC), Paris, France, 21–25 May 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
7. Feng, Q.; He, D.; Zeadally, S.; Kumar, N.; Liang, K. Ideal lattice-based anonymous authentication protocol for mobile devices. *IEEE Syst. J.* **2018**, *13*, 2775–2785.

8. Dharminder, D.; Chandran, K.P. LWESM: Learning with error based secure communication in mobile devices using fuzzy extractor. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 4089–4100.
9. Dabra, V.; Bala, A.; Kumari, S. LBA-PAKE: Lattice-based anonymous password authenticated key exchange for mobile devices. *IEEE Syst. J.* **2020**, *15*, 5067–5077.
10. Islam, S.H. Provably secure two-party authenticated key agreement protocol for post-quantum environments. *J. Inf. Secur. Appl.* **2020**, *52*, 102468.
11. Islam, S.H.; Basu, S. PB-3PAKA: Password-based three-party authenticated key agreement protocol for mobile devices in post-quantum environments. *J. Inf. Secur. Appl.* **2021**, *63*, 103026.
12. Kumar, U.; Garg, M.; Kumari, S.; Dharminder, D. A construction of post quantum secure and signal leakage resistant authenticated key agreement protocol for mobile communication. *Trans. Emerg. Telecommun. Technol.* **2023**, *34*, e4660.
13. Moony, B.; Barnwal, A.K.; Singh, M.; Mishra, D. Quantum secure two party authentication protocol for mobile devices. *Peer-Netw. Appl.* **2023**, *16*, 2548–2559.
14. Seyhan, K.; Akleylek, S. A new lattice-based password authenticated key exchange scheme with anonymity and reusable key. *PeerJ Comput. Sci.* **2024**, *10*, e1791.
15. Pursharthi, K.; Mishra, D. Towards post-quantum authenticated key agreement scheme for mobile devices. *J. Inf. Secur. Appl.* **2024**, *82*, 103754.
16. Sarkar, P.; Nag, A. Lattice-based device-to-device authentication and key exchange protocol for IoT system. *Int. J. Inf. Technol.* **2024**, *16*, 4167–4179.
17. Jiang, C.; Xu, C.; Han, Y.; Zhang, Z.; Chen, K. Two-factor authenticated key exchange from biometrics with low entropy rates. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 3844–3856.
18. Pursharthi, K.; Mishra, D. A computationally efficient and randomized RLWE-based key exchange scheme. *Clust. Comput.* **2024**, *27*, 1599–1610.
19. Pursharthi, K.; Mishra, D. Cryptanalysis and amendment of authenticated key exchange protocol for mobile devices. *Peer-Netw. Appl.* **2025**, *18*, 108.
20. Zhang, J.; Zhang, Z.; Ding, J.; Snook, M.; Dagdelen, Ö. Authenticated key exchange from ideal lattices. In Proceedings of the Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015; Proceedings, Part II 34; Springer: Berlin/Heidelberg, Germany, 2015; pp. 719–751.
21. Fluhrer, S. Cryptanalysis of ring-LWE based key exchange with key share reuse. *Cryptol. Eprint Arch.* **2016**. Available online: https://eprint.iacr.org/2016/085 (accessed on 23 September 2025).
22. Chaudhary, D.; Dadsena, P.K.; Pal, Y.; Yadav, D.; Jain, J.; Kumar, M.R.; Preetham, L.M. Security Issues and Solutions in Post Quantum Authenticated Key Exchange for Mobile Devices. In Proceedings of the International Conference on Data Science and Applications, Bandung, Indonesia, 9–10 August 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 125–134.
23. Islam, S.H.; Zeadally, S. Provably secure identity-based two-party authenticated key agreement protocol based on CBi-ISIS and Bi-ISIS problems on lattices. *J. Inf. Secur. Appl.* **2020**, *54*, 102540.
24. Rana, S.; Mishra, D. Lattice-based key agreement protocol under ring-LWE problem for IoT-enabled smart devices. *Sādhanā* **2021**, *46*, 84.
25. Akleylek, S.; Soysaldı, M. A new lattice-based authentication scheme for IoT. *J. Inf. Secur. Appl.* **2022**, *64*, 103053.
26. Wang, Q.; Wang, D.; Cheng, C.; He, D. Quantum2fa: Efficient quantum-resistant two-factor authentication scheme for mobile devices. *IEEE Trans. Dependable Secur. Comput.* **2021**, *20*, 193–208.
27. Alkim, E.; Ducas, L.; Pöppelmann, T.; Schwabe, P. Post-quantum key {Exchange—A} new hope. In Proceedings of the 25th USENIX security symposium (USENIX Security 16), Austin, TX, USA, 10–12 August 2016; pp. 327–343.
28. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans. Dependable Secur. Comput.* **2016**, *15*, 708–722.
29. Mishra, D.; Pursharthi, K.; Rewal, P. Development of quantum-enhanced authenticated key agreement protocol for autonomous vehicles. *Veh. Commun.* **2023**, *44*, 100688.
30. Regev, O. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM (JACM)* **2009**, *56*, 34.
31. Bellare, M.; Pointcheval, D.; Rogaway, P. Authenticated key exchange secure against dictionary attacks. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Bruges, Belgium, 14–18 May 2000; Springer: Berlin/Heidelberg, Germany, 2000; pp. 139–155.
32. Dharminder, D.; Reddy, C.B.; Das, A.K.; Park, Y.; Jamal, S.S. Post-Quantum Lattice-Based Secure Reconciliation Enabled Key Agreement Protocol for IoT. *IEEE Internet Things J.* **2022**, *10*, 2680–2692.