Provably Secure Two-Party Authenticated Key Agreement Protocol for Post-Quantum Environments

SK Hafizul Islam,*[1]

Abstract

A two-party authenticated key agreement (2PAKA) protocol is a cryptographic tool employed widely to allow two users to generate a shared and fresh session key between them in each session over an insecure network. The authenticated version of a two-party key agreement protocol is popular because it can easily withstand the impersonation of the user. In the literature, many 2PAKA protocols have been put forward with the intractability assumptions of the discrete logarithm (DLP) problem and integer factorization problem (IFP). Some recent studies showed that the 2PAKA protocols based on these assumptions are insecure in post-quantum environments. To resolve this issue, we have designed a lattice-based 2PAKA (LB-2PAKA) protocol with the intractability of the ring-learning-with-errors (RLWE) problem. The proposed LB-2PAKA protocol is also analyzed in the random oracle model to measure provable security and to estimate the breaching time. To evaluate the performance, we used the LatticeCrypto Library and estimated the running time of our LB-2PAKA protocol. Besides, we analyzed the communication cost requirement of our LB-2PAKA protocol.

Keywords: Authenticated Key Agreement, Lattice, Provable security, Random Oracle Model, Ring-Learning-with-Errors.

1. Introduction

A two-part key agreement (2PKA) protocol is a cryptographic primitive that allows two users to set up a common and fresh session key in each session among them over an insecure network. In this kind of protocol, two users are involved: (i) *initiator*, who will initiate the protocol, and (ii) *responder*, who will respond to the *initiator*. In the literature, an unauthenticated version of the 2PKA protocol was introduced by Diffie and Hellman [4]. However, this unauthenticated protocol was vulnerable to the man-in-the-middle attack. The main reason is that a sender sends a message without signing it to a receiver, and an adversary can replace the message by a forged message, which can not be detected by the receiver. Therefore, the authenticated key agreement (AKA) protocol is popular for practical applications. In an AKA protocol, each user can verify other users, so that an adversary cannot impersonate any user and hence, cannot perform the man-in-the-middle attack.

1.1. Motivations and Contributions

We have designed a two-party AKA (2PAKA) protocol in this paper. In the literature, many 2PAKA protocols [2, 8, 3, 10, 18] have been put forward based on the integer factorization problem (IFP), the RSA problem, the discrete logarithm problem (DLP), or the computational Diffie-Hellman (CDH) problem. However, some recent studies[15, 16] showed that these hard problems are vulnerable by a polynomial time-bounded algorithm in quantum computing environments. Recently, the NIST has inaugurated a post-quantum crypto project[14] to design and standardize robust cryptographic tools in the post-quantum environments. Recently, one of the mathematical tools, known as *ideal lattice* has attracted security scholars to design secure cryptographic mechanisms based on lattices due to the following reasons: (i) the operations performed in a lattice are much faster than other operations executed in a non-lattice-based protocol [7], (ii) lattice-based cryptographic tools are designed with the intractability assumption of the worst-case problems, while the traditional cryptographic protocols are devised using the average-case problems, and (iii) the

*Corresponding author. (SK Hafizul Islam)
Email address: hafi786@gmail.com (SK Hafizul Islam,*)
[1]Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, West Bengal 741235, India.

hardness assumptions on the ideal lattices can resist the quantum computing attacks. In contrast, non-lattice based computational problems are vulnerable in the same environments. Accordingly, we have motivated to design a lattice-based 2PAKA (LB-2PAKA) protocol suitable for post-quantum environments. Our LB-2PAKA protocol exhibit the following merits:

(1) We proposed an LB-2PAKA protocol on ideal lattices based on the ring-learning-with-errors (RLWE) problem. The RLWE problem is assumed as hard as some computational problems in the worst-case on ideal lattices.

(2) The proposed LB-2PAKA protocol is analyzed in the random oracle model (ROM), which exhibits that it is provably secure against the RLWE problem on ideal lattices.

(3) In our LB-2PAKA protocol, two users can establish a secure session key among them over an insecure network. To avoid the ephemeral secret leakage (ESL) attack, we included the long-term static private keys of the users and the ephemeral secrets selected by the users in the session key.

(4) The performance of our LB-2PAKA protocol is evaluated using the LatticeCrypto Library [9], which is used popularly in many lattice-based protocols [7, 5].

(5) The informal security analysis demonstrated that our LB-2PAKA protocol could provide resilience against the known active and passive attacks.

1.2. Reviews on Lattice-based Protocols

In the literature, many cryptographic scholars have initiated to proposed some lattice-based cryptographic protocols. In 2007, Ding et al. [6] firstly put forwarded an unauthenticated Diffie-Hellman key agreement protocol using a password. The protocol is analyzed to be provably secure in the ROM with the intractability assumption of the RLWE problem on ideal lattices. In the ROM, Ding et al. [5] designed two different client-server authentication (CSA) protocols with the intractability assumption of the RLWE problem. Later on, Feng et al. [7] put forwarded a provably secure and anonymous CSA protocol for mobile users with the intractability assumption of the RLWE problem on ideal lattices. Zhang et al. [19] firstly proposed a provably secure and two-pass lattice-based unauthenticated two-party key agreement protocol with the intractability assumption of the RLWE problem. In this protocol, the initiator U_i draws the random samples r_i, $f_i \xleftarrow{R} \chi_\beta$, computes $x_i = \boldsymbol{a} \cdot r_i + 2 \cdot \boldsymbol{f}_i$ and sends $\{x_i\}$ to the responder U_j. After receiving the message $\{x_i\}$, the responder U_j draws the random samples r_j, $f_j \xleftarrow{R} \chi_\beta$, computes the message $\{y_j, w_j\}$ and sends it to U_i. Note that no mechanism (i.e., signature [20], or message authentication code (MAC) [11]) has been adapted by U_i and U_j to verify the integrity of the messages $\{x_i\}$ and $\{y_j, w_j\}$ if changed during the communication. Accordingly, the protocol in [19] is vulnerable to the man-in-the-middle attack.

1.3. Organization

The remaining part of this paper is structured as follows. Section 2 discusses the basics of ideal lattices and related hardness problems on it. Section 3 proposes an adversarial model for an LB-2PAKA protocol. Section 4 explain our LB-2PAKA protocol. Section 5 justifies the correctness and security analysis of our LB-2PAKA protocol. Section 6 analyze the computation and communication costs of our LB-2PAKA protocol. Finally, Section 7 concludes the paper.

2. Preliminaries

Let $(x^n + 1) \in \mathbb{Z}[x]$ is an irreducible polynomial or prime polynomial over integers, where n signify the security parameter, $n = 2^t$, for some positive $t \in \mathbb{Z}$. Let $\mathbb{Z}[x]$ and $\mathbb{Z}_q[x]$ signify the rings of polynomials over $\mathbb{Z}$ and $\mathbb{Z}_q$, respectively, where $q \bmod 2n \equiv 1$ ($q = 2^{\omega(\log_2 n)} + 1$) be a public prime modulus. Let $\mathbb{R} = \mathbb{Z}[x]/(x^n + 1)$ be the ring of integer polynomials modulo $(x^n + 1)$ over $\mathbb{Z}$ and $\mathbb{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ is the rings of polynomials modulo $(x^n + 1)$ and each coefficient is reduced modulo q over $\mathbb{Z}_q$. The number of element of $\mathbb{R}_q$ is q^n and each element of $\mathbb{R}_q$ is a polynomial of degree less than n whose coefficients belongs to the set $\mathbb{Z}_q = \{0, 1, \cdots, q-1\}$ [12].

Lemma 1. Given any $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}$, we have $||\boldsymbol{a} \cdot \boldsymbol{b}||_2 \leq \sqrt{n}||\boldsymbol{a}||_2 \cdot ||\boldsymbol{b}||_2$ and $||\boldsymbol{a} \cdot \boldsymbol{b}||_\infty \leq n||\boldsymbol{a}||_\infty \cdot ||\boldsymbol{b}||_\infty$ [5], where $||\cdot||_2$ and $||\cdot||_\infty$ are the L_2 norm L_∞ norm [7].

Lemma 2. Given any fixed positive real number $\beta = \omega(\sqrt{\log_2 n})$, we have $\mathbf{Pr}_{\boldsymbol{a} \leftarrow \chi_\beta}[||\boldsymbol{a}||_2 > \beta \cdot \sqrt{n}] \leq 2^{-n+1}$ [13], where χ_β signify the *discrete Gaussian distribution* over $\mathbb{R}_q$. Let $E = \{-\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{q}{4} \rfloor\}$ be the middle set of $\mathbb{Z}_q = \{-\frac{q}{2}, \cdots, \frac{q}{2}\}$. For any $x \in \mathbb{Z}_q$, the characteristic function **Cha** is described as follows:

$$\mathrm{Cha}(x) = \begin{cases} 0 & x \in E \\ 1 & x \notin E \end{cases}$$

The auxiliary modular function $\mathbf{Mod}_2 : \mathbb{Z}_q \times \{0,1\} \rightarrow \{0,1\}$ is described as $\mathbf{Mod}_2(\boldsymbol{v}, \boldsymbol{b}) = (\boldsymbol{v} + \boldsymbol{b} \cdot \frac{q-1}{2}) \bmod q \bmod 2$, where $\boldsymbol{v} \in \mathbb{Z}_q$ and $\boldsymbol{b} = \mathbf{Cha}(\boldsymbol{v})$ [19].

Lemma 3. Let $q \geq 3$ be a odd prime number and $n = 2^t$ is the security parameter, for some positive $t \in \mathbb{Z}$ and $q = 2^{\omega(\log_2 n)} + 1$. Let $\boldsymbol{v} \in \mathbb{Z}_q$ be selected uniformly at random. For any $b \in \{0,1\}$ and any $\boldsymbol{w} \in \mathbb{Z}_q$, the output distribution of $\mathbf{Mod}_2(\boldsymbol{v} + \boldsymbol{w}, b)$ given $\mathbf{Cha}(\boldsymbol{v}))$ is statistically close to uniform on $\{0,1\}$ [19].

Lemma 4. Given a prime number $q \geq 3$ and $\boldsymbol{a}, \boldsymbol{e} \in \mathbb{R}_q$ such that $|e| < \frac{q}{8}$, we have $\mathbf{Mod}_2(\boldsymbol{a}, \mathbf{Cha}(\boldsymbol{a})) = \mathbf{Mod}_2(\boldsymbol{w}, \mathbf{Cha}(\boldsymbol{a}))$, where $\boldsymbol{w} = \boldsymbol{a} + 2 \cdot \boldsymbol{e}$ [19].

The functions **Cha** and $\mathbf{Mod}_2$ can be extended on $\mathbb{R}_q$ as follows: Given $\boldsymbol{a} = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} \in \mathbb{R}$, we considered it as a vector $\boldsymbol{a} = (a_0, a_1, \cdots, a_{n-1})$. For $\boldsymbol{v} = (v_0, v_1, \cdots, v_{n-1}) \in \{0,1\}^n$, we have $\mathbf{Cha}(\boldsymbol{a}) = (\mathbf{Cha}(a_0), \mathbf{Cha}(a_1), \cdots, \mathbf{Cha}(a_{n-1}))$ and $\mathbf{Mod}_2(\boldsymbol{a}, \boldsymbol{v}) = (\mathbf{Mod}_2(a_0, v_0), \mathbf{Mod}_2(a_1, v_1), \cdots, \mathbf{Mod}_2(a_{n-1}, v_{n-1}))$ [19].

Definition 1 (Ring Learning with Errors (RLWE))**.** Let $\boldsymbol{s} \in \mathbb{R}_q$ and $\boldsymbol{A}_{\boldsymbol{s},\chi_\beta}$ be the distribution over $(\boldsymbol{a}, \boldsymbol{a} \cdot \boldsymbol{s} + 2 \cdot \boldsymbol{e}) \in \mathbb{R}_q \times \mathbb{R}_q$, where $\boldsymbol{a}$ is randomly selected from $\mathbb{R}_q$ and $\boldsymbol{e} \xleftarrow{R} \chi_\beta$. It is computationally infeasible for a probabilistic polynomial time-bounded (PPT) algorithm $\mathscr{B}$ to distinguish the distribution $\boldsymbol{A}_{\boldsymbol{s},\chi_\beta}$ from the uniform distribution on $\mathbb{R}_q \times \mathbb{R}_q$, for given a fixed $\boldsymbol{a}$ sampled from χ_β and polynomially many samples [12].

Definition 2 (Pairing with errors (PWE))**.** Let $\boldsymbol{x}, \boldsymbol{s} \in \mathbb{R}_q$, we describe $\psi(\boldsymbol{x}, \boldsymbol{s}) = \mathbf{Mod}_2(\boldsymbol{x} \cdot \boldsymbol{s}, \mathbf{Cha}(\boldsymbol{x} \cdot \boldsymbol{s}))$. For given $\boldsymbol{a}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}_q$, it is computationally infeasible for any $\mathscr{B}$ to compute $\psi(\boldsymbol{x}, \boldsymbol{s})$, where $\boldsymbol{y} = \boldsymbol{a} \cdot \boldsymbol{s} + 2 \cdot \boldsymbol{e}$ for the unknowns $\boldsymbol{s}, \boldsymbol{e} \xleftarrow{R} \chi_\beta$.

Definition 3 (Decision pairing with errors (DPWE))**.** Given $\boldsymbol{a}, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}_q$ and $\omega, \sigma \in \{0,1\}^*$, where $\omega = \mathbf{Cha}(\boldsymbol{k})$ for $\boldsymbol{x} \in \mathbb{R}_q$ and $\sigma = \mathbf{Mod}_2(\boldsymbol{k}, \omega)$. It is computationally infeasible for any $\mathscr{B}$ to determine whether $\boldsymbol{k} = \boldsymbol{x} \cdot \boldsymbol{s} + 2 \cdot \boldsymbol{g}$ and $\boldsymbol{y} = \boldsymbol{a} \cdot \boldsymbol{s} + 2 \cdot \boldsymbol{e}$ for the unknowns $\boldsymbol{s}, \boldsymbol{g}, \boldsymbol{e} \in \chi_\beta$ or $(\boldsymbol{k}, \boldsymbol{y})$ is uniformly random in $\mathbb{R}_q \times \mathbb{R}_q$.

3. Adversary Model

Here, we explain the formal adversarial model of an LB-2PAKA protocol based on the computational model called the random oracle model (ROM), which was introduced in [1]. This model tests the semantic security of the session key and the mutual authentication between two users. In our adversarial model, we assume that $\mathbb{U} = \{U_1, U_2, \cdots, U_n\}$ be a set of n users and any two distinct users from $\mathbb{U}$ can cooperatively execute the proposed LB-2PAKA protocol. The set $\mathbb{U}$ includes both honest users and dishonest users. The dishonest user may be a malicious insider user or an outsider user. We assume that each U_i ($1 \leq i \leq n$) must have their long-term static private and public keys $\langle (s_i, e_i), p_i \rangle$ and issued the public key certificate for the public key p_i from a trusted certificate authority (CA). The proposed LB- 2PAKA protocol can be viewed as a collection of n programs, which are cooperatively executed by different n pairs of users. An instance of the protocol among two different users is considered as a session, and each user may execute multiple sessions simultaneously with different users. The communication in every session is assumed to be fully monitored by an adversary $\mathscr{A}$, who can modify, insert, or delete the messages exchanges between two users.

We denote the oracle $\Pi^k_{U_i,U_j}$ is the kth instance of the user U_i involved with its partner U_j in a session k. The session identifier $sid^k_{U_i,U_j}$ is considered as the concatenation of the messages exchanged among U_i with identity id_i and U_j with identity id_j. The partner identifier $pid^k_{U_i,U_j}$ of $\Pi^k_{U_i,U_j}$ is a set containing the identity id_i of U_i and the identity id_j of the party U_j with whom U_i wants to establish a session. The semantic security of the session key and mutual authentication between the involved users in a session of LB-2PAKA protocol is defined by a series of challenge-response games, which were executed by a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$. In each game, $\mathscr{A}$ must solve a challenge on a ***Test*** session to violate the semantic security and mutual authentication of LB-2PAKA protocol. In each game, $\mathscr{A}$ has given the capability to execute the following queries to $\mathscr{C}$, which in turn, returns the answer of the corresponding queries to $\mathscr{A}$:

- **Execute($\Pi^k_{U_i,U_j}$):** $\mathscr{A}$ can ask an **Execute**($\Pi^k_{U_i,U_j}$) query to $\mathscr{C}$ to get the messages computed as per the description of the protocol among the instances $\Pi^k_{U_i}$ and $\Pi^k_{U_j}$.
- **Send(Π^k_P, m):** $\mathscr{A}$ can ask a **Send**(Π^k_P, m) query to acquire the messages spawned by the instance Π^k_P. Note that $\mathscr{A}$ can also commence the protocol by executing a **Send**(Π^k_P, ***start***) query to $\mathscr{C}$.
- **Corrupt(P):** This query models an insider attack and an unknown key share attack. To compromise an entity $P \in \mathbb{U}$ and to acquire the long-term private key of P, $\mathscr{A}$ can ask a **Corrupt**(P) query to $\mathscr{C}$ and then $\mathscr{C}$ will return the long-term private key of P to $\mathscr{A}$.
- **Reveal($\Pi^k_{U_i,U_j}$):** The execution of this query models an active attack, called known-key attack. With a **Reveal**($\Pi^k_{U_i,U_j}$), $\mathscr{A}$ obtains the session key SK for misusing it. Assume that SK is freshly computed in the kth session among the instances $\Pi^k_{U_i}$ and $\Pi^k_{U_j}$, and $\Pi^k_{U_i}$ and it's partner $\Pi^k_{U_j}$ enter in to the *accepted* state in the session k. Suppose $\mathscr{A}$ sends a **Reveal**($\Pi^k_{U_i,U_j}$) query to $\mathscr{C}$, then $\mathscr{C}$ returns the answers as follows:
  - $\mathscr{C}$ returns the session key SK to $\mathscr{A}$, provided $\Pi^k_{U_i}$ and it's partner $\Pi^k_{U_j}$ are in *accepted* state and they freshly compute a correct session key SK.
  - Otherwise, $\mathscr{C}$ returns *null* value to $\mathscr{A}$ as output.
- **RevealEph($\Pi^k_{U_i,U_j}$):** To obtain the ephemeral secrets used by the participants U_i and U_j in a session k, $\mathscr{A}$ can ask a **RevealEph**($\Pi^k_{U_i,U_j}$) query to $\mathscr{C}$, then $\mathscr{C}$ returns the ephemeral secrets used in the session k. This query models the goal of the ephemeral secret leakage attack.
- **Test($\Pi^k_{U_i,U_j}$):** This query evaluates the semantic security of SK. $\mathscr{A}$ is allowed to send only a single **Test**($\Pi^k_{U_i,U_j}$) query to $\mathscr{C}$ for each session k. In response to a **Test**($\Pi^k_{U_i,U_j}$) query, $\mathscr{C}$ returns SK if it is computed between $\Pi^k_{U_i}$ and its partner $\Pi^k_{U_j}$, else, returns a *null value*. When $\mathscr{A}$ simulates a **Test**($\Pi^k_{U_i,U_j}$) query, $\mathscr{C}$ flips an unbiased coin b and outputs SK as answer if $b = 1$ holds, else, chooses a *random value* from $\{0,1\}^l$ as answer for the **Test**($\Pi^k_{U_i,U_j}$) query asked by $\mathscr{A}$.

Definition 4 (Accepted state)**.** An oracle $\Pi^k_{U_i,U_j}$ is called *accepted* if (i) the instances $\Pi^k_{U_i}$ and $\Pi^k_{U_j}$ are mutually authenticate each other in a fresh session k and holds the same session key SK, (ii) $\Pi^k_{U_i}$ and $\Pi^k_{U_j}$ hold a session identifier $sid^k_{U_i,U_j}$, and (3) $\Pi^k_{U_i}$ and $\Pi^k_{U_j}$ hold a partner identifier $pid^k_{U_i,U_j}$.

Definition 5 (Partner)**.** Two oracles $\Pi^k_{U_i,U_j}$ and $\Pi^l_{U_j,U_i}$ are said to be *partnered iff* the following holds: (1) $\Pi^k_{U_i,U_j}$ and $\Pi^l_{U_j,U_i}$ entered into *accepted* state, (ii) $sid^k_{U_i,U_j} = sid^l_{U_j,U_i}$, and (ii) $pid^k_{U_i,U_j} = pid^l_{U_j,U_i}$.

Definition 6 (Freshness of a session)**.** An instance $\Pi^k_{U_i,U_j}$ is *fresh* if the following holds: (i) $\Pi^k_{U_i,U_j}$ is in the *accepted* state, (ii) No **Reveal**($\Pi^k_{U_i,U_j}$) query should be executed, (iii) No **Send**($\Pi^k_{U_i}$, m) or **Send**($\Pi^k_{U_j}$, m) query should be executed, (iv) Only **Corrupt**(U_i) or **Corrupt**(U_j) can be executed.

Definition 7 (Session key secrecy)**.** An LB-2AKA protocol satisfy the session key secrecy, if no polynomial time bounded adversary $\mathscr{A}$ has a non-negligible advantage in the following game played between $\mathscr{A}$ and infinite set of oracles $\Pi^k_{U_i,U_j}$: (i) A long-term private key is assigned to each user $U_i \in \mathbb{U}$, (ii) $\mathscr{A}$ may ask several queries and get back the response from the corresponding oracles, (iii) There is no **Reveal**($\Pi^k_{U_i,U_j}$) query or **Corrupt**(U_i)/**Corrupt**(U_j) query being asked before the **Test**($\Pi^k_{U_i,U_j}$) query has been asked, and (iv) $\mathscr{A}$ may ask other queries during asking the **Test**($\Pi^k_{U_i,U_j}$) query where $\Pi^k_{U_i,U_j}$ is fresh. $\mathscr{A}$ outputs its guess bit b' for the bit b, which is chosen in the **Test**($\Pi^k_{U_i,U_j}$) query eventually and the game is successfully terminated.

Definition 8 (Probability of success)**.** Let $Succ(\mathscr{A})$ be the event that $\mathscr{A}$ makes a single **Test**($\Pi^k_{U_i,U_j}$) query to some fresh session k to the instance $\Pi^k_{U_i,U_j}$ that has end successfully, and eventually returns a guess bit b', where $b' = b$, the bit b was selected in the ***Test*** query. The advantage, i.e., probability of success of $\mathscr{A}$ with in the polynomial time bound t in violating the semantic security of the session key of our LB-2PAKA protocol is defined as $Adv^{LB-2PAKA}_{\mathscr{A}}(t) = |Pr[Succ(\mathscr{A})] - \frac{1}{2}|$.

Definition 9 (Semantic security)**.** The proposed LB-2PAKA protocol is semantically secure if (i) in the presence of $\mathscr{A}$, $\Pi_{U_i}^k$ and its partner $\Pi_{U_j}^k$ are in *accepted* state, and (ii) $Adv_{\mathscr{A}}^{LB-2PAKA}(t)$ is negligible.

Table 1: Notations

Notation	Meaning
q	Odd prime number
χ_β	Discrete Gaussian distribution for $\beta > 0$
$\boldsymbol{x} \xleftarrow{R} \chi_\beta$	$\boldsymbol{x}$ is sampled from $\mathbb{R}_q$ uniformly at random according to χ_β
$\mathbb{R}$, $\mathbb{R}_q$	Ring, where $\mathbb{R} = \mathbb{Z}[x]/(x^n+1)$ and $\mathbb{R}_q = \mathbb{Z}_q[x]/(x^n+1)$
SA	System administrator
CA	Certificate authority
U_i/U_j	Initiator (User)/Responder (User)
id_i	Identity of U_i
(s_i, e_i)	long-term static private key of U_i
p_i	Long-term static public key of U_i, $p_i = \boldsymbol{a} \cdot s_i + 2 \cdot \boldsymbol{e}_i$
$H_j(\cdot)$	Cryptographic hash functions, $H_j : \{0,1\}^* \rightarrow \{0,1\}^l$, $j = 1,2,3$

4. Proposed LB-2PAKA Protocol

Here, we propose a new LB-2PAKA protocol in which three entities are involved: i) the initiator U_i, ii) the responder U_j, and iii) the system administrator (SA). The role of SA is to generate all the system parameters. The systems parameters are essential to perform the execution of†the proposed protocol. Table 1 lists the different notations and their meanings. In our LB-2PAKA protocol, both the users U_i and U_j generate their pair of keys: a static long-term private key and a corresponding static long-term public key. The static long-term public keys of U_i and U_j must be authenticated by a certificate authority (CA) before using them. We describe our LB-2PAKA protocol with the following phases.

4.1. Initialization Phase

This phase is executed by the SA to generate the system parameters. SA selects a sufficiently large odd prime $q > 16\beta^2 n^{\frac{3}{2}}$ such that $q \bmod 2n \equiv 1$, where $n \in \mathbb{Z}_q$, $n = 2^t$ for some positive $t \in \mathbb{Z}$. SA further selects $\mathbb{R} = \mathbb{Z}[x]/(x^n+1)$, $\mathbb{R}_q = \mathbb{Z}_q[x]/(x^n+1)$, a discrete Gaussian distribution χ_β, a fixed element $\boldsymbol{a}$ uniformly at random from $\mathbb{R}_q$ and three one-way secure hash functions $H_1, H_2, H_3 : \{0,1\}^* \rightarrow \{0,1\}^l$. In our protocol, we choose the SHA-512 hash function for hashing. SA publishes $\langle n, q, \chi_\beta, H_1(\cdot), H_2(\cdot), H_3(\cdot)\rangle$ as system parameters.

4.2. Key Generation Phase

User U_i with an identity id_i generates his/her long-term static secret key as $(s_i, e_i) \xleftarrow{R} \chi_\beta$ and computes his/her static long-term public key as $p_i = \boldsymbol{a} \cdot s_i + 2 \cdot \boldsymbol{e}_i \in \mathbb{R}_q$.

4.3. Authenticated Key Agreement Phase

(a) The initiator U_i draws two samples $r_i, f_i \xleftarrow{R} \chi_\beta$ and computes the followings:

$$x_i = \boldsymbol{a} \cdot r_i + 2 \cdot \boldsymbol{f}_i$$

$$t_{i_1} = s_i \cdot p_j$$

$$w_{i_1} = \mathbf{Cha}(t_{i_1})$$

$$\sigma_{i_1} = \mathbf{Mod}_2(t_{i_1}, w_{i_1})$$

$$\alpha_{i_1} = H_1(id_i, x_i, \sigma_{i_1})$$

U_i sends the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1} \rangle$ to the responder U_j.

(b) After receiving the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1} \rangle$ from U_i, the responder U_j draws two samples r_j, $f_j \xleftarrow{R} \chi_\beta$ and computes the followings:

$$x_j = \boldsymbol{a} \cdot r_j + 2 \cdot \boldsymbol{f}_j$$

$$t_{j_1} = s_j \cdot p_i$$

$$w_{j_1} = \mathbf{Cha}(t_{j_1})$$

$$\sigma_{j_1} = \mathbf{Mod}_2(t_{j_1}, w_{j_1})$$

$$\alpha_{j_1} = H_1(id_j, x_j, \sigma_{j_1})$$

U_j sends the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1} \rangle$ to U_i.

(c) After receiving the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1} \rangle$ from U_j, U_i further computes the followings:

$$\sigma'_{j_1} = \mathbf{Mod}_2(t_{i_1}, w_{j_1})$$

$$\alpha'_{j_1} = H_1(id_j, x_j, \sigma'_{j_1})$$

If $\alpha'_{j_1} = \alpha_{j_1}$, U_i confirms that the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1} \rangle$ is correctly received from U_j. User U_i further computes the followings:

$$t_{i_2} = r_i \cdot x_j$$

$$w_{i_2} = \mathbf{Cha}(t_{i_2})$$

$$\sigma_{i_2} = \mathbf{Mod}_2(t_{i_2}, w_{i_2})$$

$$\alpha_{i_2} = H_2(id_i, x_i, x_j, \sigma_{i_2})$$

U_i sends the message $\langle id_i, w_{i_2}, \alpha_{i_2} \rangle$ to U_j.

(d) After receiving the message $\langle id_i, w_{i_2}, \alpha_{i_2} \rangle$ from U_i, U_j computes

$$\sigma'_{i_1} = \mathbf{Mod}_2(t_{j_1}, w_{i_1})$$

$$\alpha'_{i_1} = H_2(id_i, x_i, \sigma'_{i_1})$$

If $\alpha'_{i_1} = \alpha_{i_1}$, U_j confirms that the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1} \rangle$ is correctly received from U_i. U_j further computes the followings:

$$t_{j_2} = r_j \cdot x_i$$

$$w_{j_2} = \mathbf{Cha}(t_{j_2})$$

$$\sigma_{j_2} = \mathbf{Mod}_2(t_{j_2}, w_{j_2})$$

$$\alpha_{j_2} = H_2(id_j, x_j, x_i, \sigma_{j_2})$$

U_j sends the message $\langle id_j, w_{j_2}, \alpha_{j_2} \rangle$ to U_i.

After receiving the message $\langle id_j, w_{j_2}, \alpha_{j_2} \rangle$ from U_j, U_i computes

$$\sigma'_{j_2} = \mathbf{Mod}_2(t_{i_2}, w_{j_2})$$

$$\alpha'_{j_2} = H_2(id_j, x_j, x_i, \sigma'_{j_2})$$

If $\alpha'_{j_2} = \alpha_{j_2}$, then U_i computes the session identifier as $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$ and the session key as $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$.

After receiving the message $\langle id_i, w_{i_2}, \alpha_{i_2} \rangle$ from U_i, U_j computes

$$\sigma'_{i_2} = \mathbf{Mod}_2(t_{j_2}, w_{i_2})$$

$$\alpha'_{i_2} = H_2(id_i, x_i, x_j, \sigma'_{i_2})$$

If $\alpha'_{i_2} = \alpha_{i_2}$, then U_j computes the session identifier as $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$ and the session key as $SK = H_3(sid, \sigma'_{i_1}, \sigma'_{i_2}, \sigma_{j_1}, \sigma_{j_2})$.

The key agreement phase of our LB-2PAKA protocol is illustrated in the Figure 1.

User U_i (Initiator)	Open network	**User U_j (Responder)**
Sample r_i, $f_i \xleftarrow{R} \chi_\beta$		Sample r_j, $f_j \xleftarrow{R} \chi_\beta$
Compute $x_i = \boldsymbol{a} \cdot r_i + 2 \cdot \boldsymbol{f}_i$, $t_{i_1} = s_i \cdot p_j$		Compute $x_j = \boldsymbol{a} \cdot r_j + 2 \cdot \boldsymbol{f}_j$, $t_{j_1} = s_j \cdot p_i$
Compute $w_{i_1} = \mathbf{Cha}(t_{i_1})$, $\sigma_{i_1} = \mathbf{Mod}_2(t_{i_1}, w_{i_1})$		Compute $w_{j_1} = \mathbf{Cha}(t_{j_1})$, $\sigma_{j_1} = \mathbf{Mod}_2(t_{j_1}, w_{j_1})$
Compute $\alpha_{i_1} = H_1(id_i, x_i, \sigma_{i_1})$		Compute $\alpha_{j_1} = H_1(id_j, x_j, \sigma_{j_1})$
	$\xrightarrow{\langle id_i, x_i, w_{i_1}, \alpha_{i_1} \rangle}$	
	$\xleftarrow{\langle id_j, x_j, w_{j_1}, \alpha_{j_1} \rangle}$	
Compute $\sigma'_{j_1} = \mathbf{Mod}_2(t_{i_1}, w_{j_1})$, $\alpha'_{j_1} = H_1(id_j, x_j, \sigma'_{j_1})$		Compute $\sigma'_{i_1} = \mathbf{Mod}_2(t_{j_1}, w_{i_1})$, $\alpha'_{i_1} = H_1(id_i, x_i, \sigma'_{i_1})$
If $\alpha'_{j_1} \neq \alpha_{j_1}$, abort		If $\alpha'_{i_1} \neq \alpha_{i_1}$, abort
Else, compute $t_{i_2} = r_i \cdot x_j$, $w_{i_2} = \mathbf{Cha}(t_{i_2})$		Else, compute $t_{j_2} = r_j \cdot x_i$, $w_{j_2} = \mathbf{Cha}(t_{j_2})$
Compute $\sigma_{i_2} = \mathbf{Mod}_2(t_{i_2}, w_{i_2})$, $\alpha_{i_2} = H_2(id_i, x_i, x_j, \sigma_{i_2})$		Compute $\sigma_{j_2} = \mathbf{Mod}_2(t_{j_2}, w_{j_2})$, $\alpha_{j_2} = H_2(id_j, x_j, x_i, \sigma_{j_2})$
	$\xrightarrow{\langle id_i, w_{i_2}, \alpha_{i_2} \rangle}$	
	$\xleftarrow{\langle id_j, w_{j_2}, \alpha_{j_2} \rangle}$	
Compute $\sigma'_{j_2} = \mathbf{Mod}_2(t_{i_2}, w_{j_2})$, $\alpha'_{j_2} = H_2(id_j, x_j, x_i, \sigma'_{j_2})$		Compute $\sigma'_{i_2} = \mathbf{Mod}_2(t_{j_2}, w_{i_2})$, $\alpha'_{i_2} = H_2(id_i, x_i, x_j, \sigma'_{i_2})$
If $\alpha'_{j_2} \neq \alpha_{j_2}$, abort		If $\alpha'_{i_2} \neq \alpha_{i_2}$, abort
Else, compute $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$		Else, compute $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$
Compute session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$		Compute session key $SK = H_3(sid, \sigma'_{i_1}, \sigma'_{i_2}, \sigma_{j_1}, \sigma_{j_2})$

Figure 1: Authenticated key agreement phase

5. Security Analysis

5.1. Correctness

Theorem 1. Let $q > 16\beta^2 n^{\frac{3}{2}}$ be an odd prime number. Let U_i and U_j follow the proposed LB-2PAKA protocol, then U_i and U_j will successfully end with a common session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$.

Proof. In the authenticated key agreement phase of our LB-2PAKA protocol, user U_i justifies the legitimacy of U_j by verifying whether $\alpha'_{j_1} = \alpha_{j_1}$ and $\alpha'_{j_2} = \alpha_{j_2}$. Since:

$$t_{i_1} = s_i \cdot p_j = a \cdot s_i \cdot s_j + 2 \cdot s_i \cdot e_j \tag{1}$$

$$t_{j_1} = s_j \cdot p_i = a \cdot s_i \cdot s_j + 2 \cdot s_j \cdot e_i \tag{2}$$

Subtracting the Eq. (2) from the Eq. (1), we get:

$$t_{i_1} - t_{j_1} = 2 \cdot s_i \cdot e_j - 2 \cdot s_j \cdot e_i \tag{3}$$

Based on the Lemma 1 and Lemma 2, we have:

$$\begin{aligned} ||s_i \cdot e_j - s_j \cdot e_i||_2 &\leq ||s_i \cdot e_j||_2 + ||s_j \cdot e_i||_2 \\ &< \sqrt{n} \cdot ||s_i||_2 \cdot ||e_j||_2 + \sqrt{n} \cdot ||s_j||_2 \cdot ||e_i||_2 \\ &< \sqrt{n} \cdot \beta \cdot \sqrt{n} \cdot \beta \cdot \sqrt{n} + \sqrt{n} \cdot \beta \cdot \sqrt{n} \cdot \beta \cdot \sqrt{n} \\ &= 2 \cdot \beta^2 \cdot n^{\frac{3}{2}} \\ &< \frac{q}{8} \ [\text{Since } q > 16\beta^2 n^{\frac{3}{2}}] \end{aligned} \tag{4}$$

Hence, from the Lemma 3 and Eq. (4), we get:

$$\begin{aligned} \sigma'_{j_1} &= \mathbf{Mod}_2(t_{i_1}, w_{j_1}) \\ &= \mathbf{Mod}_2(t_{i_1}, \mathbf{Cha}(t_{j_1})) \\ &= \mathbf{Mod}_2(t_{j_1}, \mathbf{Cha}(t_{j_1})) \\ &= \sigma_{j_1} \end{aligned} \tag{5}$$

Thus, according to the Eq. (5), we also get:

$$\begin{aligned} \alpha'_{j_1} &= H_1(id_j, x_j, \sigma'_{j_1}) \\ &= H_1(id_j, x_j, \sigma_{j_1}) \\ &= \alpha_{j_1} \end{aligned} \tag{6}$$

Again, we have:

$$t_{i_2} = x_i \cdot x_j = a \cdot r_i \cdot r_j + 2 \cdot r_i \cdot f_j \tag{7}$$

$$t_{j_2} = x_j \cdot x_i = a \cdot r_i \cdot r_j + 2 \cdot x_j \cdot f_i \tag{8}$$

Subtracting the Eq. (8) from the Eq. (7), we get:

$$t_{i_2} - t_{j_2} = 2 \cdot r_i \cdot f_j - 2 \cdot r_j \cdot f_i \tag{9}$$

From the Lemma 1 and Lemma 2, we get:

$$||r_i \cdot f_j - r_j \cdot f_i||_2 < \frac{q}{8} \tag{10}$$

Hence, from the Lemma 3 and Eq. (10), we have:

$$\begin{aligned} \sigma'_{j_2} &= \mathbf{Mod}_2(t_{i_2}, w_{j_2}) \\ &= \mathbf{Mod}_2(t_{i_2}, \mathbf{Cha}(t_{j_2})) \\ &= \mathbf{Mod}_2(t_{j_2}, \mathbf{Cha}(t_{j_2})) \\ &= \sigma_{j_2} \end{aligned} \tag{11}$$

Thus, according to the Eq. (11), we also get:

$$\begin{aligned} \alpha'_{j_2} &= H_2(id_j, x_j, x_i, \sigma'_{j_2}) \\ &= H_2(id_j, x_j, x_i, \sigma_{j_2}) \\ &= \alpha_{j_2} \end{aligned} \tag{12}$$

According to the Eq. (5), Eq. (12) and $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$, user U_i computes the session key as

$$SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2}) = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2}) \tag{13}$$

User U_j justifies the legitimacy of U_i by verifying whether $\alpha'_{i_1} = \alpha_{i_1}$ and $\alpha'_{i_2} = \alpha_{i_2}$. Based on the Lemma 3 and Eq. (6), we get:

$$\begin{aligned} \sigma'_{i_1} &= \mathbf{Mod}_2(t_{j_1}, w_{i_1}) \\ &= \mathbf{Mod}_2(t_{j_1}, \mathbf{Cha}(t_{i_1})) \\ &= \mathbf{Mod}_2(t_{i_1}, \mathbf{Cha}(t_{i_1})) \\ &= \sigma_{i_1} \end{aligned} \tag{14}$$

Thus, according to the Eq. (14), we also get:

$$\begin{aligned} \alpha'_{i_1} &= H_1(id_i, x_i, \sigma'_{i_1}) \\ &= H_1(id_i, x_i, \sigma_{i_1}) \\ &= \alpha_{i_1} \end{aligned} \tag{15}$$

Hence, from the Lemma 3 and Eq. (10), we get:

$$\begin{aligned} \sigma'_{i_2} &= \mathbf{Mod}_2(t_{j_2}, w_{i_2}) \\ &= \mathbf{Mod}_2(t_{j_2}, \mathbf{Cha}(t_{i_2})) \\ &= \mathbf{Mod}_2(t_{i_2}, \mathbf{Cha}(t_{i_2})) \\ &= \sigma_{i_2} \end{aligned} \tag{16}$$

Thus, according to the Eq. (16), we also get:

$$\begin{aligned} \alpha'_{i_2} &= H_2(id_i, x_i, x_j, \sigma'_{i_2}) \\ &= H_2(id_i, x_i, x_j, \sigma_{i_2}) \\ &= \alpha_{i_2} \end{aligned} \tag{17}$$

According to the Eq. (14), Eq. (16) and $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$, U_j calculates the session key as

$$SK = H_3(sid, \sigma'_{i_1}, \sigma'_{i_2}, \sigma_{j_1}, \sigma_{j_2}) H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2}) \tag{18}$$

According to the Eq. (13) and Eq. (18), U_i and U_j successfully established a common secret session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ between them. □

5.2. Provable Security

Theorem 2. Let $Adv_{\mathscr{A}}^{LB-2PAKA}(t)$ be the probability of success of $\mathscr{A}$ to infringement the semantic security of the session key and mutual authentication between U_i and U_j in our LB-2PAKA protocol with in a polynomial time-bound t. Assume that $Adv_{\mathscr{A}}^{RLWE}(t')$ is the advantage of $\mathscr{A}$ to solve an instance of the RLWE problem within a polynomial time-bound t'. $\mathscr{A}$ is allowed to ask maximum q_{H_i} $(i = 1,2,3)$, q_{se}, and q_{ex} number of queries to H_i oracle, *Send* oracle, and *Execute* oracle, respectively and T_{smul} denotes the running time required to perform one componentwise multiplication with scalar operation in $\mathbb{R}_q$. Therefore we have:

$$Adv_{\mathscr{A}}^{LB-2PAKA}(t) \leq \frac{q_{H_1}^2}{2l} + \frac{q_{H_2}^2}{2l} + \frac{(q_{ex}+q_{se})^2}{2^l} + \frac{q_{H_3}^2}{2^l} + (q_{ex}+q_{se}) Adv_{\mathscr{A}}^{RLWE}(t')$$

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

Proof. Assume that the semantic security of the session key of our LB-2PAKA protocol can be violated by $\mathscr{A}$, then a challenger $\mathscr{C}$, which runs a PPT algorithm, can solve an instance of the RLWE problem in $\mathbb{R}_q \times \mathbb{R}_q$. We consider the games $Game_i$ $(0 \leq i \leq 4)$, where $Game_0$ defines the real attack on our LB-2PAKA protocol [11, 17]. In each game $Game_i$ is interactively played between $\mathscr{A}$ and $\mathscr{C}$. In each game, to get a response with the correct output for an oracle query, $\mathscr{C}$ runs a PPT algorithm. It can be noted that, for every $Game_i$, an event X_i $(0 \leq i \leq 4)$ is defined based on which $\mathscr{A}$ wins in breaching the semantic security a session key of our LB-2PAKA protocol in $Game_i$. Suppose that the event E that is independent of X_i, may occur during the simulation of the LB-2PAKA protocol by $\mathscr{A}$ such that E is detectable by $\mathscr{C}$. Noted that $Game_i$ and $Game_{i+1}$ are indistinguishable unless E occurs. Therefore, we have:

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

$Game_0$: The simulation of this game is indistinguishable to the real attack on our LB-2PAKA protocol in ROM. In this game, $\mathscr{A}$ does not alter the simulation strategies of the different oracles, i.e., all the instances of the users $P \in \mathbb{U}$ is modeled as the real execution of the LB-2PAKA protocol in ROM. Therefore, we have:

$$Adv_{\mathscr{A}}^{LB-2PAKA}(t) = |Pr[X_0] - \frac{1}{2}| \tag{20}$$

$Game_1$: The simulation of this game is indistinguishable with the game $Game_0$ other than $\mathscr{A}$ executes different **Hash queries** other than the ways executed in the game $Game_0$. In this game, $\mathscr{C}$ maintains three initially-empty lists H_j^{list}, $j = 1,2,3$ with the tuples of the form (x, y), where x is the input to a **Hash** oracle and y is the corresponding output. If $\mathscr{A}$ asks a **Hash** query for the input x, $\mathscr{C}$ explores H_j^{list}, $j = 1,2,3$ and returns y as answer to $\mathscr{A}$ if a record (x,y) is found in H_j^{list}, $j = 1,2,3$. Otherwise, $\mathscr{C}$ randomly chooses a value $y \in \mathbb{Z}_q$, returns it to $\mathscr{A}$ and incorporates the new tuple (x, y) to H_j^{list}, $j = 1,2,3$. The real attack in ROM and the simulation of this game are absolutely indistinguishable since the other oracles **Execute**, **Send**, **Reveal**, **RevealEph**, **Corrupt**, and **Test** are also simulated similar way as executed in the real attack. Thus, we have:

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

Game$_2$: The simulation of this game is indistinguishable with the game *Game*$_1$ other than it will be aborted if a *collision* is found in the simulation of $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$, $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$, $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$, and $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$. Based on the birthday paradox, the probability of collisions in output of the oracles H_1 and H_2 are at most $\frac{q_{H_1}^2}{2l}$ and $\frac{q_{H_2}^2}{2l}$, respectively. Likewise, the probability of collision in output of the oracle H_3 is at most $\frac{(q_{sc}+q_{ex})^2}{2^l}$. Thus, we have:

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

Game$_3$: The simulation of this game is indistinguishable with the game *Game*$_2$ other than $\mathscr{A}$ queries oracle H_3 with the input of $(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$. Therefore, the probability of guessing the bit b, which was selected in the ***Test*** query with the oracle H_3 is at most $\frac{q_{H_3}^2}{2^l}$. Thus, we have:

$$|Pr[X_3] - Pr[X_2]| \quad \leq \quad \frac{q_{H_3}^2}{2^l} \tag{23}$$

Game$_4$: In this game, different types of guessing attacks through offline modes are considered, i.e., he session key SK may be computed without simulating H_3 oracle.

- Here, SK is predicted without simulating the oracle H_3, i.e., SK is absolutely independent from H_3 and $(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$. According to our LB-2PAKA protocol, both the usresU_i and U_j compute the session key as $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma'_{j_1}, \sigma'_{j_2})$, where $sid = (id_i, id_j, x_i, x_j, w_{i_1}, w_{j_1}, \alpha_{i_1}, \alpha_{j_1})$, $\sigma_{i_1} = \mathbf{Mod}_2(t_{i_1}, w_{i_1})$, $\sigma_{j_1} = \mathbf{Mod}_2(t_{j_1}, w_{j_1})$, $\sigma_{i_2} = \mathbf{Mod}_2(t_{i_2}, w_{i_2})$ and $\sigma_{j_2} = \mathbf{Mod}_2(t_{j_2}, w_{j_2})$. Accordingly, we say that if $\mathscr{A}$ can successfully guess SK, then the adversary $\mathscr{A}$ can solve an instance of RLWE problem with in polynomial time bound t' by interacting with the challenger $\mathscr{C}$. Therefore, we have:

$$|Pr[X_4] - Pr[X_3]| \quad \leq \quad (q_{ex}+q_{se}) Adv_{\mathscr{A}}^{RLWE}(t') \tag{24}$$

- If $\mathscr{A}$ doesn't simulate H_3 with the correct input $(sid, \sigma_A, \sigma_B)$, then $\mathscr{A}$ has no advantage in distinguishing the real session key and a random session key

Therefore, we have:

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

From the Eq. (20), we have:

$$\begin{aligned} Adv_{\mathscr{A}}^{LB-2PAKA}(t) &= |Pr[X_0] - \frac{1}{2}| \\ &= |Pr[X_0] - Pr[X_4] + Pr[X_4] - \frac{1}{2}| \\ &= |Pr[X_0] - Pr[X_4]| \end{aligned} \tag{26}$$

From the Eq. (26) and using the Eqs. (20) - (25), we have:

$$\begin{aligned} Adv_{\mathscr{A}}^{LB-2PAKA}(t) &= |Pr[X_0] - Pr[X_1] + Pr[X_1] - Pr[X_2] + Pr[X_2] - Pr[X_3] + Pr[X_3] - Pr[X_4]| \\ &\leq \frac{q_{H_1}^2}{2l} + \frac{q_{H_2}^2}{2l} + \frac{(q_{ex}+q_{se})^2}{2^l} + \frac{q_{H_3}^2}{2^l} + (q_{ex}+q_{se}) Adv_{\mathscr{A}}^{RLWE}(t') \end{aligned}$$

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

□

5.3. Further Security Discussion

Ths subsection is proposed to justify the security attributes [10] provided by our LB-2PAKA protocol.

5.3.1. ***Man-in-the-Middle (MITM) Attack***

In our LB-2PAKA protocol, user U_i sends the messages $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$ and $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$ to the user U_j, where $\alpha_{i_1} = H_1(id_i, x_i, \sigma_{i_1})$ and $\alpha_{i_2} = H_2(id_i, x_i, x_j, \sigma_{i_2})$. Assume that an adversary $\mathscr{A}$ captures these two messages and then tried to fabricated them. However, the fabrication of the messages $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$ and $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$ is not possible. Because $\mathscr{A}$ has to compute two forged authenticators in place of α_{i_1} and α_{i_2}. For this purpose, $\mathscr{A}$ has to compute $w_{i_1} = \mathbf{Cha}(t_{i_1})$ and $\sigma_{i_1} = \mathbf{Mod}_2(t_{i_1}, w_{i_1})$, which is equivalent solve the RLWR problem for the instance $(\boldsymbol{a}, p_i) = (\boldsymbol{a}, \boldsymbol{a}\cdot s_i + 2\cdot \boldsymbol{e}_i)$ or $(\boldsymbol{a}, p_j) = (\boldsymbol{a}, \boldsymbol{a}\cdot s_j + 2\cdot \boldsymbol{e}_j)$. Similarly, $\mathscr{A}$ has to solve the RLWE problem for the instance $(\boldsymbol{a}, x_i) = (\boldsymbol{a}, \boldsymbol{a}\cdot r_i + 2\cdot \boldsymbol{f}_i)$ or $(\boldsymbol{a}, x_j) = (\boldsymbol{a}, \boldsymbol{a}\cdot r_j + 2\cdot \boldsymbol{f}_j)$ to compute the forged authenticator in place of α_{i_2}. Therefore, $\mathscr{A}$ cannot impersonate the initiator U_i. In the same way, user U_j sends the messages $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$ and $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$ to the user U_i, where $\alpha_{j_1} = H_1(id_j, x_j, \sigma_{j_1})$ and $\alpha_{j_2} = H_2(id_j, x_j, x_i, \sigma_{j_2})$. By the similar reason, $\mathscr{A}$ cannot impersonate the responder U_j. Accordingly, our LB-2PAKA protocol withstands the Man-in-the-Middle (MITM) attack.

5.3.2. ***Known-Key (KK) Attack***

The known-key attack says that an adversary $\mathscr{A}$ is unable to generate the current session key even if the exchanged messages and the session keys of some earlier sessions are known to $\mathscr{A}$. In our LB-2PAKA protocol, U_i and U_j calculates the session $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ in a session. Assume that $\mathscr{A}$ captures all the messages $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$, $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$, $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$ and $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$ for a session and the session key SK of that session is disclosed. Despite the disclosure of SK, $\mathscr{A}$ is unable compute the session keys of other sessions due to the onewayness of the hash function $H_3(\cdot)$ and the freshness of the samples r_i, r_j, f_i and f_j drawn at random from χ_β.

5.3.3. ***Key-Compromise Impersonation (KCI) Attack***

Assume that an adversary $\mathscr{A}$ knows the private key (s_i, e_i) of the initiator U_i. However, $\mathscr{A}$ may not be able to impersonate the responder U_j to U_i. In a session, U_i draws two random samples $r_i, f_i \xleftarrow{R} \chi_\beta$, then computes the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$ and sends it to U_j ($\mathscr{A}$). Assume that $\mathscr{A}$ captures the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$ and tried to impersonate U_j to U_i as follows. To impersonate U_j and to know the session key SK of that session, $\mathscr{A}$ draws two random samples $r_j, f_j \xleftarrow{R} \chi_\beta$, computes the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$ and sends it to U_i. As a response, U_i computes $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$ and sends it to $\mathscr{A}$. In response, $\mathscr{A}$ computes $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$ and sends it to U_i. However, $\mathscr{A}$ cannot compute the session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ because σ_{i_1}, σ_{i_2}, σ_{j_1}, and σ_{j_2} are unknown to him/her due to private key (s_j, e_j) and secret random samples (r_i, f_i). Accordingly, our LB-2PAKA protocol withstands the KCI attack.

5.3.4. ***Unknown Key-Share (UKS) Attack***

Assume that U_i and U_j generate a fresh and common session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ among them in a session. In our LB-2PKA protocol, neither entity believe that SK is shared with an adversary $\mathscr{A}$. Because, both of U_i and U_j compute the session key SK after mutually authenticating each other by verifying whether $\alpha'_{j_1} \stackrel{?}{=} \alpha_{j_1}$, $\alpha'_{i_1} \stackrel{?}{=} \alpha_{i_1}$, $\alpha'_{j_2} \stackrel{?}{=} \alpha_{j_2}$ and $\alpha'_{i_2} \stackrel{?}{=} \alpha_{i_2}$ are correct. Hence, our LB-2PKA protocol is free from the UKS attack.

5.3.5. ***Ephemeral Secret Leakage (ESL) Attack***

In our LB-2PKA protocol, U_i and U_j generate a fresh and common session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ between then in each session, where the computation of σ_{i_1} and σ_{j_1} depends on the private keys (s_i, e_i) of U_i and (s_j, e_j) of U_j. Besides, computation of σ_{i_1} and σ_{j_1} depends on the random samples (r_i, f_i) selected by U_i and (r_j, f_j) selected by U_j. Therefore, an adversary $\mathscr{A}$ is unable to generate the session key SK even if (r_i, f_i) and (r_j, f_j) are known to him/her. Thus, our LB-2PAKA protocol withstands the ESL attack.

5.3.6. ***Perfect Forward Secrecy (PFS)***

Assume that the private keys (s_i, e_i) of U_i and (s_j, e_j) of U_j are somehow disclosed to an adversary $\mathscr{A}$. Despite having the private keys of U_i and U_j, $\mathscr{A}$ cannot compute the $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$, which is computed between U_i and U_j in a session. Because the random samples (r_i, f_i) selected by U_i and (r_j, f_j) selected by U_j are still unknown to $\mathscr{A}$. Thus, our LB-2PKA protocol provides the PFS of the session key.

*5.3.7. **No Key Control (NKC)***

In our LB-2PKA protocol, U_i and U_j establish a common session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ by incorporating the the random samples (r_i, f_i) selected by U_i and (r_j, f_j) selected by U_j. Therefore, none of U_i and U_j alone can determine the session key. Hence, we can conclude that our LB-2PKA protocol provides NKC property of the session key.

6. Performance Analysis

This section is devoted to present the performance analysis of our LB-2PAKA protocol against the efficiency measurement attributes (i) execution cost, and (ii) communication cost. According to [5, 7], we choose $\log_2\beta = 17.1$ for the Gaussian sampling distribution. We use the SHA-512 hash function for hashing operation. The length of the digest of the SHA-512 function is 512 bits. We choose the length of n, q, and an element in $\mathbb{R}_q$ are 1024 bits, 12289 bits, 4096 bits, respectively. As discussed in section 1.2 that Zhang et al.'s protocol [19] is an unauthenticated two-party key agreement protocol; therefore, we are not comparing it with our LB-2PAKA protocol. Accordingly, we only estimate the execution cost and communication cost of our LB-2PAKA protocol in this section.

6.1. Execution Cost

Here, we estimated the execution cost of our LB-2PAKA protocol. To calculate the running time of our LB-2PAKA protocol, we consider different cryptographic operations as follows: T_{Ge}, T_{smul}, T_{pmul}, T_{pma}, T_{cha} and T_H denote the running time time to draw a sample value from the χ_β distribution, execute a component-wise multiplication with scalar operation in $\mathbb{R}_q$, to execute a component-wise multiplication operation in $\mathbb{R}_q$, to execute a component-wise multiplication and addition operation in $\mathbb{R}_q$, to execute a **Cha** operation in $\mathbb{R}_q$ and to calculate a digest value for the SHA-512 hash function H, respectively. In comparison, we ignored the $\mathbf{Mod}_2$ operation in $\mathbb{R}_q$ since it is equivalent AND operation and thus its running time is ignorable.

The implementation of our LB-2PAKA protocol simulated using LatticeCrypto Library [9]. Based on the experimental setup given in [7], it is assumed that U_i and U_j executes the proposed LB-2PAKA protocol using a Dell desktop PC with the following configurations (i) Operating system: Windows 10 Professional, (ii) CPU: 3.4-GHz Intel(R) Core(TM) i7-6700 processor, and (iii) RAM: 8GB. The mode of message exchange between U_i and U_j is Bluetooth communication. In this environment, the average running time of different ideal lattice-based operations (in nanoseconds), which have been estimated over 100000 executions, are given in Table 2.

Table 2: Execution cost

Notation	T_{Ge}	T_{smul}	T_{pmul}	T_{pma}	T_{cha}	T_H
Time	73.503 *ns*	00.298 *ns*	00.307 *ns*	02.549 *ns*	00.698 *ns*	14.09 *ns*

In key agreement phase, the initiator U_i draws two random samples r_i, $f_i \xleftarrow{R} \chi_\beta$ and generates the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$. The execution cost for the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$ is $2T_{Ge} + T_{smul} + T_{pma} + T_{pmul} + T_{cha} + T_H \approx 164.948$ *ns*. After receiving the message $\langle id_i, x_i, w_{i_1}, \alpha_{i_1}\rangle$, the responder U_j verifies it by checking whether $\alpha'_{i_1} \stackrel{?}{=} \alpha_{i_1}$ is correct and the associated execution cost is $T_H \approx 14.09$ *ns*. In addition, the responder U_j performed the similar computations to generated the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$ and hence the execution cost is 164.948 *ns*. After receiving the message $\langle id_j, x_j, w_{j_1}, \alpha_{j_1}\rangle$, U_i verifies it by checking whether $\alpha'_{j_1} \stackrel{?}{=} \alpha_{j_1}$ is correct and the associated execution cost is $T_H \approx 14.09$ *ns*. In addition, U_i computes the messages $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$ and its execution cost is $T_{pma} + T_{cha} + T_H \approx 17.337$ *ns*. After receiving the message $\langle id_i, w_{i_2}, \alpha_{i_2}\rangle$, U_j verifies it by checking whether $\alpha'_{i_2} \stackrel{?}{=} \alpha_{2_1}$ is correct and the associated execution cost is $T_H \approx 14.09$ *ns*. Next, U_j computes the message $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$ and the corresponding execution cost for this message is $T_{pma} + T_{cha} + T_H \approx 17.337$ *ns*. In addition, U_j computes a session key $SK = H_3(sid, \sigma_{i_1}, \sigma_{i_2}, \sigma_{j_1}, \sigma_{j_2})$ and the execution cost for it is $T_H \approx 14.09$ *ns*. The execution cost for U_i to verify the message $\langle id_j, w_{j_2}, \alpha_{j_2}\rangle$ by executing whether $\alpha'_{j_2} \stackrel{?}{=} \alpha_{j_2}$ is $T_H \approx 14.09$ *ns*. The execution cost for U_i to compute the session key SK is $T_H \approx 14.09$ *ns*. Therefore, the overall execution cost of our LB-2PAKA protocol is $2 \times (164.948 + 3 \times 14.09 + 17.337) \approx 224.555$ *ns*.

6.2. Communication Cost

To compute a session key, the initiator U_i sends the messages $\langle id_i, x_i, w_{i_1}, \alpha_{i_1} \rangle$ and $\langle id_i, w_{i_2}, \alpha_{i_2} \rangle$ to the responder U_j. Then the responder U_j, sends the messages $\langle id_j, x_j, w_{j_1}, \alpha_{j_1} \rangle$ and $\langle id_j, w_{j_2}, \alpha_{j_2} \rangle$ to the initiator U_i. Here $\{x_i, x_j\} \in \{0,1\}^{4096}$, $\{w_{i_1}, w_{i_2}, w_{j_1}, w_{j_2}\} \in \{0,1\}$, $\{\alpha_{i_1}, \alpha_{i_2}, \alpha_{j_1}, \alpha_{j_2}\} \in \{0,1\}^{512}$. Therefore, the communication overhead of our LB-2PAKA protocol is estimated as $4096 \times 2 + 1 \times 4 + 512 \times 6 = 10244$ bits per session.

7. Conclusion

We design an LB-2PAKA protocol with the hardness assumption of the RLWE problem. In our protocol, two users can set up a shared session key for each session among them over an insecure network, which is monitored by an adversary. Our LB-2PAKA protocol offers provable security in the random oracle model. Our provable security analysis showed that the probability of an adversary to infringement the semantic security of the session key is negligible, and the time need is more than the time required to break the RLWE problem. We also examine the execution time of our LB-2PAKA protocol using the LatticeCrypto Library. Besides, we analyze the communication overhead of our LB-2PAKA protocol. We believe that our LB-2PAKA protocol will be more suitable than the traditional 2PAKA protocols for many Internet-based applications in post-quantum environments.

References

[1] Mihir Bellare and Phillip Rogaway Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73, Fairfax, Virginia, USA, 1993.

[2] Xuefei Cao, Weidong Kou, and Xiaoni Du. A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges. *Information Sciences*, 180:2895–2903, 2010.

[3] Lanjun Dang, Jie Xu, Xuefei Cao, Hui Li, Jie Chen, Yueyu Zhang, and Xiaotong Fu. Efficient identity-based authenticated key agreement protocol with provable security for vehicular ad hoc networks. *International Journal of Distributed Sensor Networks*, 14(4), 2018.

[4] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transaction on Information Theory*, 22(6):644–654, 1976.

[5] Jintai Ding, Saed Alsayigh, Jean Lancrenon, R V Saraswathy, and Michael Snook. Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In *Proceedings of the Cryptographers Track RSA Conference*, pages 183–204, Cham, 2017. Springer International Publishing.

[6] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. *IACR Cryptology ePrint Archieve*, Report No.: 2012/688, 2012.

[7] Qi Feng, Debiao He, Sherali Zeadaly, Neeraj Kumar, and Kaitai Liang. Ideal lattice-based anonymous authentication protocol for mobile devices. *IEEE Systems Journal*, pages 1–11, 2018.

[8] Marko Hölbl, Tatjana Welzer, and Boštjan Brumen. An improved two-party identity-based authenticated key agreement protocol using pairings. *Journal of Computer and System Sciences*, 78(1):142–150, 2012.

[9] IBM. LatticeCrypto Library. [2016. Available: https://www.microsoft.com/en-us/download/confirmation.aspx?id=52371].

[10] SK Hafizul Islam and Gosta Pada Biswas. Design of two-party authenticated key agreement protocol based on ECC and self-certified public keys. *Wireless Personal Communications*, 82(6):2727–2750, 2015.

[11] SK Hafizul Islam, Mohammad S. Obaidat, and Ruhul Amin. An anonymous and provably secure authentication scheme for mobile user. *International Journal Communication Systems*, 29:1529–1544, 2016.

[12] Vadim Lyubashevskyy, Chris Peikertz, and Oded Regev. On ideal lattices and learning with errors over rings. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT 2010)*, pages 1–23, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[13] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal of Computing*, 37(1):267–302, 2007.

[14] NIST. Post-quantum crypto project. [2016; Available: http://csrc.nist.gov/groups/ST/post-quantum-crypto/].

[15] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE, November 1994.

[16] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):302–332, 1999.

[17] Victor Shoup. Sequences of games: A tool for taming complexity in security proofs. *Cryptology ePrint Archieve, Report 2004/332*, 2004. [Available: http://eprint.iacr.org/2004/332].

[18] Yuh-Min Tseng. An efficient two-party identity-based key exchange protocol. *Informatica*, 18(1):125–136, 2007.

[19] Jiang Zhang, Zhenfeng Zhang, Jintai Ding, Michael Snook, and *Özgü*r Dagdelen. Authenticated key exchange from ideal lattices. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT 2015)*, pages 719–751, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[20] Yudi Zhang, Debiao He, Xinyi Huang, Ding Wang, Kim-Kwang Raymond Choo, and Jiang Wangrumen. White-box implementation of the identity-based signature scheme in the IEEE P1363 standard for public key cryptography. *IEICE Transactions on Information and Systems*, 2019.