Provably Secure Three-party Password Authenticated Key Exchange Protocol Based On Ring Learning With Error

Dongqing Xu[1], Debiao He[2,*], Kim-Kwang Raymond Choo[3], Jianhua Chen[1]

[1] *School of Mathematics and Statistics , Wuhan University, Wuhan, China*

[2] *State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China*

[3] *Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249, USA*

Abstract

Three-party Password Authenticated Key Exchange (3PAKE) protocol is an important cryptographic primitive, where clients can establish a session key using easy-to-remember passwords. A number of 3PAKE protocols based on traditional mathematical problems have been presented in the literature, but these protocols are not able to resist attacks using quantum computers. In this paper, we construct the first 3PAKE protocol from lattices. Lattice-based cryptography is a promising post-quantum cryptography approach. We then prove its security in the random oracle model, and implement the proposed protocol using LatticeCrypto. The implementation results shows our protocol is very efficient in practice.

Keywords: Key Exchange, Three-Party PAKE, Lattice, RLWE, Post-Quantum

1. Introduction

In a Password Authenticated Key Exchange (PAKE) protocol, participants use low-entropy, easy-to-remember password to establish a high-entropy session key. PAKE protocol has diverse applications, for example in mobile devices such as Android and iOS devices.

Bellovin and Merritt [5] designed the first 2PAKE protocol, in which two participants, say A and B, share a password to establish a session key via a public network. However, this scheme does not scale well due to the associated costs and complexity in password management for large number of users. Specifically, every pair of users have to share a password and each user has to remember $n-1$ password if there are n members in the network. To address this limitation, a trusted server S is introduced (i.e. a 3PAKE protocol). In a 3PAKE protocol, each client shares a password with the trusted server which is used to establish a session key with another client.

In the existing literature, 3PAKE protocols can be based on either public/private key cryptography (e.g. [21, 28, 14, 29]) or symmetric key cryptography (e.g. [20, 22]). In the former case, verifying the server's static public key results in additional computation for the clients. In the latter case, to prove the security of the protocols generally requires "the ideal cipher model" which is a much stronger assumption. A number of 3PAKE protocols are designed using neither static public key nor symmetric cryptosystems (e.g. [7, 30, 18, 23]), which generally do not have a formal security proof or shown to be insecure (e.g. [18, 23]).

One of the (relatively) recent trends in computing paradigm is quantum computing. In theory, a quantum computer can hold significantly more memory (than a conventional computer) and can be used to solve hard problems, such as large number factoring problem, and discrete logarithm problems in finite field or elliptic curve groups. These hard numbers form the basis of most of our existing cryptosystems; thus, it is important for the research community to design cryptographic algorithms and systems that are resilient to quantum computing attacks.

Existing 3PAKE protocols are largely based on traditional mathematical problems, which are insecure against attacks using quantum computers.

There are a number of hard problems that are known to be resilience against quantum computing attacks, and lattice based cryptography is one of the most promising post-quantum option.

Therefore, in this paper, we propose the first 3PAKE protocol from lattices, hereafter referred to as RLWE-3PAKE as it is based on the RLWE-PAK [13]. RLWE (ring learning with errors) is a computational problem, and is described in Section 3. Our protocol is somewhat parallel analogous to a 3PAKE protocol using elliptic curve cryptography (ECC).

We demonstrate the security of RLWE-3PAKE in the random oracle model [3, 4, 2] under the PWE assumption. We then implement RLWE-3PAKE using LatticeCrypto and evaluate its performance.

We will now describe the organization of this paper. In the next section, we briefly describe related literature. Background materials is presented in Section 3. We present RLWE-3PAKE, its security analysis, and performance evaluaton in Sections 4, 5 and 6, respectively. We conclude this paper in the last section.

2. Related literature

3PAKE. To establish a session key from shared password, a number of 3PAKE protocols based on discrete logarithms problem in finite field have been presented in the literature. Similar to the history in key establishment protocols [9, 10, 11], a number of published 3PAKE protocols were subsequently found to be insecure. For example, Ding et al. [14] and Sun et al. [29] revealed that the 3PAKE protocol in [28] is not able to resist online password guessing attacks. Similarly, Lin et al. [21] revealed the same protocol to be vulnerable to offline password guessing attacks. Both Sun et al. [29] and Lin et al. [21] presented an improved protocol designed to be secure against password guessing attacks. However, these improved 3PAKE protocols require the use of the server's static public key of the server, which does not scale well in practice. To enhance efficiency, Lin et al. [22] presented a 3PAKE protocol based on a symmetric cryptosystem where the server's public key is replaced by hash functions. However, such a protocol can only be proven secure in "the ideal cipher model", which is a much stronger assumption.

There have been other research directions in the design of 3PAKE protocols.

For example, Chang et al. [6] presented a round efficient 3PAKE protocol based on one way function with trapdoor. However, in this protocol, the server has to store the trapdoor in addition to the password table. Later, Chen et al. [8] pointed out the protocol of Chang et al. [6] is insecure against undetectable online password guessing attacks. The authors then presented a new 3PAKE protocol [8]. Lu et al. [23] presented a 3PAKE protocol based on the chosen basis CDH problem, which was subsequently revealed to be insecure in [12, 18, 25].

As previously discussed, there are 3PAKE protocols that do not require static public key or symmetric cryptosystem. For example, Chang et al. [7] constructed a 3PAKE protocol based on discrete logarithm problem in finite field by replacing the symmetric cryptosystem with a simple XOR operation. However, Wu et al. [30] pointed out that this protocol is vulnerable to partition attacks where an adversary can obtain the correct password offline. The authors then designed an ECC-based 3PAKE protocol.

AKE and PAKE from lattices. Zhang [31] designed an authenticated key exchange based on lattice similar to HMQV [19]. However, the validity of the security proof was questioned in [16].

There are only a small number of PAKE protocols based on lattice at the time of this research. One of these lattice-based PAKE protocols is that of Katz and Vaikuntanathan [17]. This protocol is proven secure in the standard model security, but it is not so efficient due to its common reference string (CRS)-based design. Compared to Ding et al.'s two-party protocol [13], Ding et al.'s two-party protocol is more efficient since it is proven secure in the random oracle model.

However, at the time of this research, there is no lattice-based 3PAKE protocol and this is the contribution we seek to fill in this paper.

3. Background

We will now introduce the notions used in this paper.

- k is the security parameter.

- For parameter N such that $f(k) < \frac{1}{k^c}$ for $k > N$, $c > 0$, function f is negligible about k.
- $\mathbb{Z}[x]$ (respectively, $\mathbb{Z}_q[x]$) is a polynomial ring in $\mathbb{Z}$ ($\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$).
- $n \in \mathbb{Z}$ is a power of 2 and $R = \mathbb{Z}[x]/(x^n + 1)$ is a ring.
- $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ for positive q.
- For an element y in R or R_q, we consider the coefficient vector of y is in $\mathbb{Z}^n$ or $\mathbb{Z}_q^n$, respectively. For a fixed $\beta > 0$, the discrete Gaussian distribution in $\mathbb{Z}^n$ can naturally be extended to that over R_q (also parametrized by β). In our protocol, we denote χ_β as this distribution over R_q.

Now, we will describe the computational hard problems underpinning the security of our proposed protocol.

LWE problem [26]. Let $n \geq 1$, $q \geq 2$ be integers and $\alpha \in (0, 1)$. For a vector $\mathbf{s} \in \mathbb{Z}_q^n$ (also known as the secret), $A_{s,\chi}$ is the LWE distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ by outputting $(\mathbf{a}, b =< \mathbf{s}, \mathbf{a} > +e \mod q)$, in which $\mathbf{a} \in \mathbb{Z}_q^n$ is chosen uniformly and randomly, e is chosen from χ .

There are two types of LWE problem, namely: search LWE problem and decision LWE problem. The former is to compute $\mathbf{s}$ for polynomial LWE samples and the latter (i.e. decision version of LWE assumption) is to distinguish between LWE samples and random samples.

RLWE Problem. For $s \in R_q$, we set A_{s,χ_β} as the distribution of $(a, as + 2x) \in R_q \times R_q$, in which $a \leftarrow R_q$ is chosen uniformly and randomly, x is chosen from χ_β independent from a. The RLWE problem assumes A_{s,χ_β} for s chosen from χ_β and the uniform distribution on R_q^2 are indistinguishable given polynomial many samples. Like the decision LWE problem, there is also decision ring learning with error (DRLWE) problem.

In this paper, the norm of a polynomial is defined as the norm of its coefficient vector. Thus, we know the following facts:

Lemma 1. R is a ring described above. If $s, t \in R$, then we have $||s \cdot t|| \leq \sqrt{n} \cdot ||s|| \cdot ||t||$ and $||s \cdot t||_\infty \leq n \cdot ||s||_\infty \cdot ||t||_\infty$

Lemma 2 [15, 24]. Given $\alpha = \omega(\sqrt{\log n})$, we have $Pr_{x \leftarrow \chi_\alpha}[||x|| > \alpha\sqrt{n}] \leq 2^{-n+1}$.

We will now review the **Cha** function and the $\mathbf{Mod_2}$ function introduced in [31]. We set $\mathbb{Z}_q = \{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$ and $E := \{-\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{q}{4} \rceil\}$. E is the "middle" of $\mathbb{Z}_q$, and **Cha** is a characteristic function with domain in $\mathbb{Z}_q$. The value of the **Cha** is 0 if input is in E; otherwise, the value is 1.

The function $\mathbf{Mod_2} : \mathbb{Z}_q \times \{0,1\} \rightarrow \{0,1\}$ is:

$\mathbf{Mod_2}(v, b) = (v + b \cdot \frac{q-1}{2}) \mod q \mod 2.$

The **Cha** function and $\mathbf{Mod_2}$ function have the following features:

Lemma 3 ([31]). $q = 2^{\omega(\log n)} + 1$ such that q is a prime and n is security parameter. v is sampled from $\mathbb{Z}_q$ uniformly and randomly. Given $b \in \{0,1\}$, $\mathbf{Cha}(v)$ and $v^{'} \in \mathbb{Z}_q$, the distribution of $\mathbf{Mod_2}(v + v^{'}, b)$ is indistinguishable from uniform on $\{0,1\}$.

Lemma 4 ([31]). Set q as a prime, and v, e are all chosen from $\mathbb{Z}_q$. We also set $|e| < \frac{q}{8}$. Given $\omega = v + 2e$, we can get $\mathbf{Mod_2}(v, \mathbf{Cha}(v)) = \mathbf{Mod_2}(\omega, \mathbf{Cha}(v))$.

This lemma also holds if it is in R_q using them coefficient-wise to the coefficients in $\mathbb{Z}_q$. That is to say, for ring element $v = (v_0, \cdots, v_{n-1}) \in R_q$ and binary-vector $\mathbf{b} = (b_0, \cdots, b_{n-1}) \in \{0,1\}^n$, we have $\mathbf{Cha}(v) = (Cha(v_0), \cdots, Cha(v_{n-1}))$ and $\mathbf{Mod_2}(v, \mathbf{b}) = (\mathbf{Mod_2}(v, b), \cdots, \mathbf{Mod_2}(v_{n-1}, b_{n-1}))$.

PWE assumption. We still use χ_β as the Gaussian distribution for a fixed $\beta \in \mathbb{R}_+^*$. For $(X, s) \in R_q^2$, we define $\tau(X, s) = \mathbf{Mod}_2(Xs, \mathbf{Cha}(Xs))$. Set $\mathcal{A}$ as probabilistic polynomial time (PPT) algorithm with input (a, X, Y, W), in which $(a, X, Y) \in R_q^3$, $W \in \{0,1\}^n$. Output will be $\{0,1\}^n$. The objective of $\mathcal{A}$ is to output the value $\tau(X, s)$, where s is chosen from R_q at random and W is the value of **Cha** function. Set

$Adv_{R_q}^{PWE}(\mathcal{A}) \overset{def}{=} Pr[a \leftarrow R_q; s \leftarrow \chi_\beta; X \leftarrow R_q; e \leftarrow \chi_\beta; Y \leftarrow as + 2e; W \leftarrow \mathbf{Cha}(Xs) : \tau(X, s) \in \mathcal{A}(a, X, Y, W)]$

According to the PWE problem, given time t and N, the value of $Adv_{R_q}^{PWE}(t, N)$ is negligible about k.

The PWE problem also has a decision version, namely: the decision pairing with errors problem (DPWE).

DPWE assumption. For $(a, X, Y, W, \sigma) \in R_q \times R_q \times R_q \times \{0,1\}^n \times \{0,1\}^n$ in which $\omega = \mathbf{Cha}K$ for $K \in R_q$ and $\sigma = \mathbf{Mod}_2(K, \omega)$. The DPWE problem is to distinguish between $(K = Xs + g, Y = as + 2e)$ for s, g and e sampled from χ_β and (K, Y) in which K, Y are all chosen from R_q randomly.

The DPWE assumption is computationally hard if the RLWE-DH problem holds, and the latter is computationally hard if the RLWE problem holds. We refer interested reader to [13] for more details.

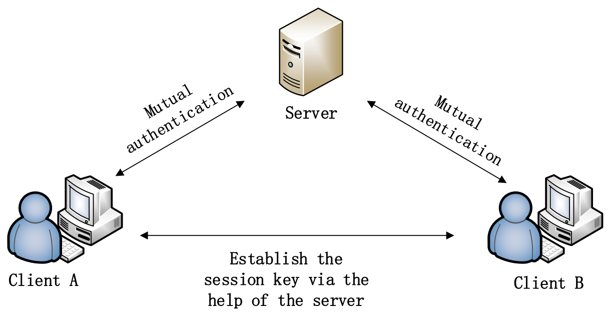

Figure 1: System model

3.1. System model

Similar to existing 3PAKE protocols, such as those in [7, 30], a typical system model has a server S and two clients, say A or B – see Figure 1.

• Server S is trusted, whose task is to authenticate A and B and relay messages.

• Clients A and B seek to establish a shared session key with the help of the server. Both A and B will also be authenticated by the server using the shared password.

3.2. Security requirements

A 3PAKE protocol should satisfy the following basic requirements:

Mutual authentication: The protocol should provide mutual authentication between two partnering clients and the server.

Session key security: Two partnering clients in the proposed protocol can establish a common session key.

Known key security: Even after an adversary $\mathcal{A}$ has acquired one particular session key, other session keys are still secure.

Resistance to three classes of password guessing attacks: To ensure the security of the password, the protocol should resist undetectable online password guessing attacks, detectable online password guessing attacks, and offline password guessing attacks.

Forward secrecy: Even if a client's password is leaked to the adversary, the adversary is not able to acquire past session keys.

Resistance to other various attacks: The protocol should withstand attacks such as user impersonation, modification and man-in-the-middle.

4. The proposed protocol

In our proposed protocol (see Figure 2), we let $q = 2^{\omega(\log n)} + 1$ and q is a prime, $H_1 : \{0,1\}^* \to R_q$ be a hash function, and $H_2 : \{0,1\}^* \to \{0,1\}^k$ be another hash function for verification of communications and key derivation, in which k is the bit-length of the session key. a is a public element chosen from R_q uniformly and randomly. a is given to all clients by the server.

In the protocol,

1. Client A sends $\{ID_A, ID_B\}$ to the server S to initiate the protocol.

2. Upon receiving $A's$ request message, S randomly samples s_1, s_2, e_1, $e_2 \leftarrow \chi_\beta$, and computes $R_{S1} = Ms_1 + 2e_1$, $R_{S2} = Ms_2 + 2e_2$, $Y_A = R_{S1} + H_1(ID_A||S||PW_A)$, $Y_B = R_{S2} + H_1(ID_B||S||PW_B)$, where $H_1(ID_A||S||PW_A)$ and $H_1(ID_A||S||PW_A)$ are shared values with A and B, respectively. Then S sends $\{Y_A, Y_B\}$ to A.

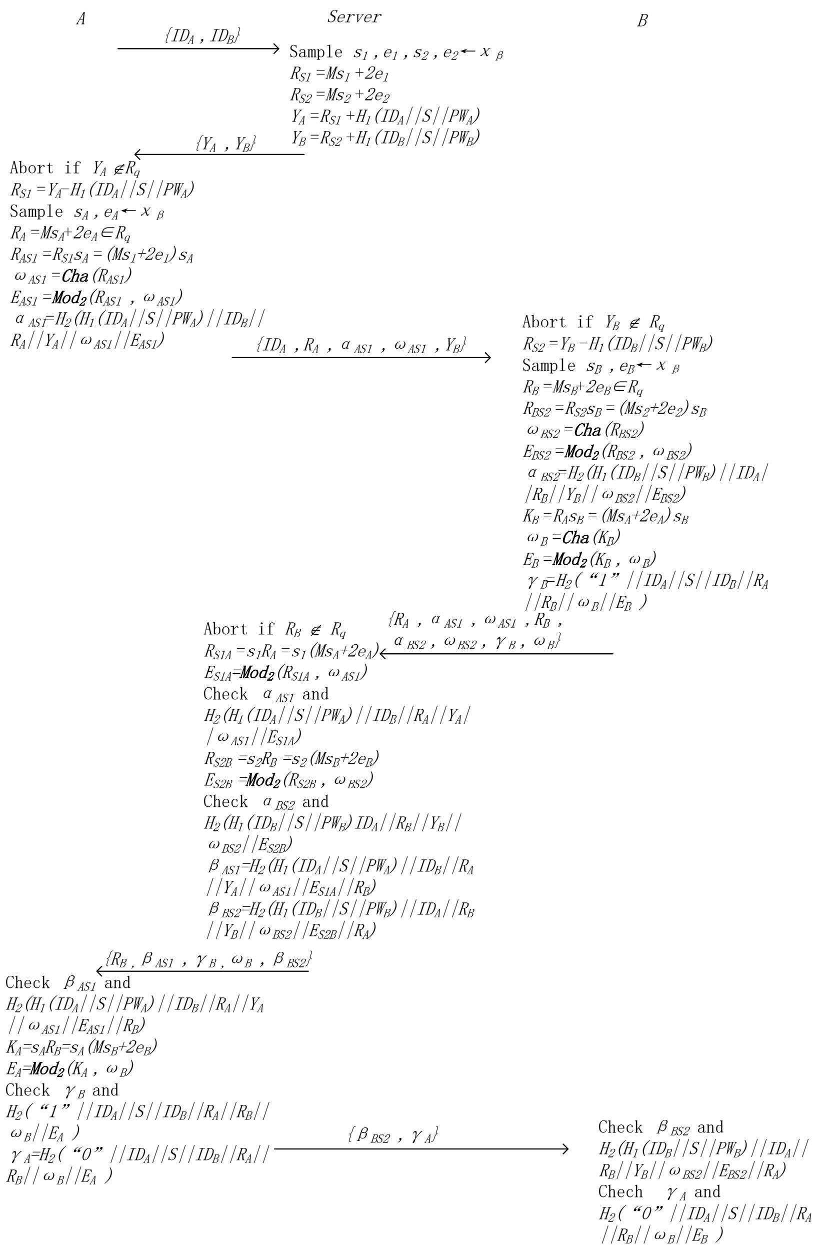
A
$Server$
B
$\{ID_A , ID_B\}$
Sample $s_1 , e_1 , s_2 , e_2 \leftarrow \chi_{\beta}$
$R_{S1} = Ms_1 + 2e_1$
$R_{S2} = Ms_2 + 2e_2$
$Y_A = R_{S1} + H_1(ID_A//S//PW_A)$
$Y_B = R_{S2} + H_1(ID_B//S//PW_B)$
$\{Y_A , Y_B\}$
Abort if $Y_A \notin R_q$
$R_{S1} = Y_A - H_1(ID_A//S//PW_A)$
Sample $s_A , e_A \leftarrow \chi_{\beta}$
$R_A = Ms_A + 2e_A \in R_q$
$R_{AS1} = R_{S1}s_A = (Ms_1+2e_1)s_A$
$\omega_{AS1} = \boldsymbol{Cha}(R_{AS1})$
$E_{AS1} = \boldsymbol{Mod_2}(R_{AS1} , \omega_{AS1})$
$\alpha_{AS1} = H_2(H_1(ID_A//S//PW_A)//ID_B//$
$R_A//Y_A//\omega_{AS1}//E_{AS1})$
$\{ID_A , R_A , \alpha_{AS1} , \omega_{AS1} , Y_B\}$
Abort if $Y_B \notin R_q$
$R_{S2} = Y_B - H_1(ID_B//S//PW_B)$
Sample $s_B , e_B \leftarrow \chi_{\beta}$
$R_B = Ms_B + 2e_B \in R_q$
$R_{BS2} = R_{S2}s_B = (Ms_2+2e_2)s_B$
$\omega_{BS2} = \boldsymbol{Cha}(R_{BS2})$
$E_{BS2} = \boldsymbol{Mod_2}(R_{BS2} , \omega_{BS2})$
$\alpha_{BS2} = H_2(H_1(ID_B//S//PW_B)//ID_A/$
$/R_B//Y_B//\omega_{BS2}//E_{BS2})$
$K_B = R_As_B = (Ms_A+2e_A)s_B$
$\omega_B = \boldsymbol{Cha}(K_B)$
$E_B = \boldsymbol{Mod_2}(K_B , \omega_B)$
$\gamma_B = H_2(\text{“1”}//ID_A//S//ID_B//R_A$
$//R_B//\omega_B//E_B)$
$\{R_A , \alpha_{AS1} , \omega_{AS1} , R_B ,$
$\alpha_{BS2} , \omega_{BS2} , \gamma_B , \omega_B\}$
Abort if $R_B \notin R_q$
$R_{S1A} = s_1R_A = s_1(Ms_A+2e_A)$
$E_{S1A} = \boldsymbol{Mod_2}(R_{S1A} , \omega_{AS1})$
Check α_{AS1} and
$H_2(H_1(ID_A//S//PW_A)//ID_B//R_A//Y_A/$
$/\omega_{AS1}//E_{S1A})$
$R_{S2B} = s_2R_B = s_2(Ms_B+2e_B)$
$E_{S2B} = \boldsymbol{Mod_2}(R_{S2B} , \omega_{BS2})$
Check α_{BS2} and
$H_2(H_1(ID_B//S//PW_B)ID_A//R_B//Y_B//$
$\omega_{BS2}//E_{S2B})$
$\beta_{AS1} = H_2(H_1(ID_A//S//PW_A)//ID_B//R_A$
$//Y_A//\omega_{AS1}//E_{S1A}//R_B)$
$\beta_{BS2} = H_2(H_1(ID_B//S//PW_B)//ID_A//R_B$
$//Y_B//\omega_{BS2}//E_{S2B}//R_A)$
$\{R_B , \beta_{AS1} , \gamma_B , \omega_B , \beta_{BS2}\}$
Check β_{AS1} and
$H_2(H_1(ID_A//S//PW_A)//ID_B//R_A//Y_A$
$//\omega_{AS1}//E_{AS1}//R_B)$
$K_A = s_AR_B = s_A(Ms_B+2e_B)$
$E_A = \boldsymbol{Mod_2}(K_A , \omega_B)$
Check γ_B and
$H_2(\text{“1”}//ID_A//S//ID_B//R_A//R_B//$
$\omega_B//E_A)$
$\gamma_A = H_2(\text{“0”}//ID_A//S//ID_B//R_A//$
$R_B//\omega_B//E_A)$
$\{\beta_{BS2} , \gamma_A\}$
Check β_{BS2} and
$H_2(H_1(ID_B//S//PW_B)//ID_A//$
$R_B//Y_B//\omega_{BS2}//E_{BS2}//R_A)$
Chech γ_A and
$H_2(\text{“0”}//ID_A//S//ID_B//R_A$
$//R_B//\omega_B//E_B)$

Figure 2: Proposed protocol: RLWE-3PAKE

3. Upon receiving the message from S, A aborts if $Y_A \notin R_q$. Otherwise, A obtains R_{S1} by computing $R_{S1} = Y_A - H_1(ID_A||S||PW_A)$. A then randomly samples $s_A, e_A \leftarrow \chi_\beta$ and calculates $R_A = Ms_A + 2e_A \in R_q$, $R_{AS1} = R_{S1}s_A = (Ms_1 + 2e_1)s_A$, $\omega_{AS1} = \mathbf{Cha}(R_{AS1})$, $E_{AS1} = \mathbf{Mod_2}(R_{AS1}, \omega_{AS1})$ and $\alpha_{AS1} = H_2(H_1(ID_A||S||PW_A)||ID_B||R_A||Y_A|| \omega_{AS1}||E_{AS1})$. Finally, it sends $\{ID_A, R_A, \alpha_{AS1}, \omega_{AS1}, Y_B\}$ to B.

4. Upon receiving $A's$ message, B aborts if $Y_B \notin R_q$ or $R_A \notin R_q$. Otherwise, B obtains R_{S2} by computing $R_{S2} = Y_B - H_1(ID_B||S||PW_B)$. Then, B randomly samples $s_B, e_B \leftarrow \chi_\beta$ and computes $R_B = Ms_B + 2e_B \in R_q$, $R_{BS2} = R_{S2}s_B = (Ms_2 + 2e_2)s_B$, $\omega_{BS2} = \mathbf{Cha}(R_{BS2})$, $E_{BS2} = \mathbf{Mod_2}(R_{BS2}, \omega_{BS2})$, $\alpha_{BS2} = H_2(H_1(ID_B||S||PW_B)||ID_A||R_B||Y_B||\omega_{BS2} ||E_{BS2})$, $K_B = R_As_B = (Ms_A+2e_A)s_B$, $\omega_B = \mathbf{Cha}(K_B)$, $E_B = \mathbf{Mod_2}(K_B, \omega_B)$, $\gamma_B = H_2(\text{“1”}||ID_A||S||ID_B||R_A||R_B||\omega_B||E_B)$. B sends $\{R_A, \alpha_{AS1}, \omega_{AS1}, R_B, \alpha_{BS2}, \omega_{BS2}, \gamma_B, \omega_B\}$ to S.

5. Upon receiving $B's$ message, S aborts if $R_B \notin R_q$. First, S computes $R_{S1A} = s_1R_A = s_1(Ms_A + 2e_A)$, $E_{S1A} = \mathbf{Mod_2}(R_{S1A}, \omega_{AS1})$ and checks whether α_{AS1} is equal to $H_2(H_1(ID_A||S||PW_A)||ID_B||R_A||Y_A||\omega_{AS1}||E_{S1A})$ to decide if authenticating A. Then, S computes $R_{S2B} = s_2R_B = s_2(Ms_B+ 2e_B)$, $E_{S2B} = \mathbf{Mod_2}(R_{S2B}, \omega_{BS2})$ and checks whether α_{BS2} is equal to $H_2(H_1(ID_B||S||PW_B)||ID_A||R_B||Y_B||\omega_{BS2}||E_{S2B})$ to decide whether to authenticate B. If both equation hold, S authenticates A, B and computes $\beta_{AS1} = H_2(H_1(ID_A||S||PW_A)||ID_B||R_A||Y_A||\omega_{AS1}||E_{S1A}||R_B)$, $\beta_{BS2} = H_2(H_1(ID_B||S||PW_B)||ID_A||R_B||Y_B||\omega_{BS2}||E_{S2B}||R_A)$. Then, S sends $\{R_B, \beta_{AS1}, \gamma_B, \omega_B, \beta_{BS2}\}$ to A.

6. Upon receiving the message from S, A checks whether β_{AS1} is equal to $H_2(H_1(ID_A||S||PW_A)||ID_B||R_A||Y_A||\omega_{AS1}||E_{AS1}||R_B)$ to decide whether to authenticate S. If the result is positive, then A authenticates S and computes $K_A = s_A \cdot R_B = s_B \cdot (Ms_B+2e_B)$, $E_A = \mathbf{Mod_2}(K_A, \omega_B)$. Then, A checks whether γ_B is equal to $H_2(\text{“1”}||ID_A||S||ID_B||R_A||R_B||\omega_B||E_B)$.

If the equation holds, then A believes B can calculate the session key. Finally, A sends $\{\beta_{BS2}, \gamma_A\}$ to B. On $B's$ side, after receiving $A's$ message, B checks whether β_{BS2} is equal to $H_2(H_1(ID_B||S||PW_B)||ID_A||R_B||Y_B||\omega_{BS2}$ $||E_{BS2}||R_A)$ to decide whether to authenticate S. If the equation holds, then B authenticates S and then checks whether γ_A is equal to $H_2($"0"$||ID_A$ $||S||ID_B||R_A||R_B||\omega_B||E_B)$. If equal, B is assured that A can compute the session key.

Then A and B can calculate the same session key $SK = H_2(2||ID_A||S||ID_B$ $||R_A||R_B||E_A)$ (or $SK = H_2(2||ID_A||S||ID_B||R_A||R_B||E_B)$).

Correctness. Set q as a prime such that $q > 16\beta^2 n^{3/2}$. The three participants A, B and S honestly follow the proposed protocol to establish a session key. Then, A and B will acquire the same session key with overwhelming probability.

Proof. To demonstrate the correctness of the proposed 3PAKE protocol, it is sufficient for us to show $\mathbf{Mod_2}(K_A, \omega_B) = \mathbf{Mod_2}(K_B, \omega_B)$. By Lemma 4, if $|K_A - K_B| < q/4$, then A and B arrive at the same value. Then, the correctness holds. Comparing K_A and K_B, we know that $K_A - K_B = 2(e_B s_A - e_A s_B)$. By Lemma 2, the norm of each individual e_B, s_A, e_A, s_B term is less than $\beta\sqrt{n}$ with overwhelming probability. Then, we obtain $K_A - K_B \leq 4\beta^2 n^{3/2} < q/4$ with overwhelming probability by Lemma 1 and the triangle inequality. Thus, $\mathbf{Mod_2}(K_A, \omega_B) = \mathbf{Mod_2}(K_B, \omega_B)$.

5. Security analysis

In this section, we prove the proposed 3PAKE protocol secure, namely: an adversary $\mathcal{A}$ is not able to distinguish between a session key SK and a same length random bit string with an advantage greater than that of an online dictionary attack.

We will first describe the security model used in the the proof.

5.1. Security model

The security model is that of [3, 4, 2], which consists of the following constructs:

- Protocol Participants: Participants of the protocol include clients and a trusted server. Any participant has one or more instances, as any participant can be involved in several concurrent executions of the protocol. Instance i of a participant U is denoted as $\prod_U^i$.
- Long lived keys: Every client has a password pw_C which is defined as a long-lived key of C and chosen randomly from dictionary D of size $|D|$. For the server, pw_S is a hash value of the client's password with some other information, such as $H_1(A||S||PW_A)$ and $H_1(B||S||PW_B)$ in this paper.
- Oracle queries: The adversary $\mathcal{A}$ can ask as many oracle queries as he/she wishes, and the description of the oracles queries is as follows:
  - **Send**$(\prod_U^i, m)$: This models the capability of $\mathcal{A}$ to send a message m to instance $\prod_U^i$. The instance $\prod_U^i$ behaves according to the protocol specification process. Send$(\prod_U^i, start)$ denotes $\mathcal{A}$ initiates an execution of the protocol.
  - **Execute**$(\prod_A^i, \prod_B^j, \prod_S^k)$: This query allows $\mathcal{A}$ to execute a run of the protocol honestly / passively, and obtains the relevant outputs.
  - **Reveal**$(\prod_U^i)$: If $\prod_U^i$ has an accepted SK, then $\mathcal{A}$ obtains a session key SK returned by $\prod_U^i$ after sending this query.
  - **corrupt**(U): $\mathcal{A}$ obtains the password pw_C of U or pw_S of S.
  - **Hash**(m): $\mathcal{A}$ queries a random oracle to obtain the hash results. The random oracle returns the results in the list if it exists, otherwise returns a random number r to $\mathcal{A}$, and stores (m, r) in the H-table. The latter is a record list for recording the Hash queries.

– **Test**($\prod_U^i$): This query relates to the semantic security of the session key SK. During the execution of the protocol, after $\mathcal{A}$ has made a number of the other oracle queries, this query is asked (and only once in the game). Once this query is asked, $\prod_U^i$ will flip a coin and based on the result of the coin flip, say b, it returns either SK (if $b = 1$) or a random string with length $|SK|$ (if $b = 0$). The query Test($\prod_U^i$) is only available to $\mathcal{A}$ if $\prod_U^i$ is fresh.

In the protocol, a session key SK is fresh if all three requirements below are satisfied.

1. $\prod_U^i$ has accepted,

2. $\prod_U^i$ is not corrupted, and both $\prod_U^i$ and its partner have not been asked a Reveal query.

In the protocol, $\prod_A^i$ and $\prod_B^j$ are considered partners if they satisfy all the following requirements:

1. $\prod_A^i$ and $\prod_B^j$ have exchanged the required number of messages,

2. $\prod_A^i$ and $\prod_B^j$ have established the same session key SK, and

3. other than $\prod_A^i$ and $\prod_B^j$, no other oracle holds the established session key.

After obtaining an outcome from asking the Test query (as described above), $\mathcal{A}$ outputs a single bit $b^{'}$. The AKE advantage of $\mathcal{A}$ in attacking the 3PAKE protocol is:

$Adv_P^{ake}(A) = 2Pr[Succ_P^{ake}(\mathcal{A})] - 1$

Definition 1 (AKE-secure): The 3PAKE protocol is AKE-secure if $Adv_P^{ake}(A)$ is negligible for all PPT adversary $\mathcal{A}$.

5.2. Provable Security

Theorem 1. We have an adversary $\mathcal{A}$ that in time t respectively asks q_s, q_e, q_{re}, q_{co} Send, Execute, Reveal, Corrupt queries. $\mathcal{A}$ also queries H_1 oracle,

H_2 oracle for q_{H_1}, q_{H_2} times, respectively. For $t^{'} = O(t + (q_{re} + q_s + q_e)t_{exp})$, we have

$Adv_{P}^{ake}(\mathcal{A}) \leq \frac{q_s}{|D|} + \frac{q_{H_1}^2 + q_{H_2}^2 + (q_e^2 + q_s^2)}{q^n} + 3Adv_{R_q}^{DRLWE}(t^{'}, q_{ro}) + 2Adv_{R_q}^{PWE}(t^{'}, q_{ro})$

Proof. In our security proof, we have the following series of protocols $P_0, P_1, \cdots, P_5$. $P_0 = P$ and in P_5, it is only possible for online guessing attack by the adversary $\mathcal{A}$. We also have

$Adv_{P_0}^{ake}(\mathcal{A}) \leq Adv_{P_1}^{ake}(\mathcal{A}) + \epsilon_1 \leq Adv_{P_2}^{ake}(\mathcal{A}) + \epsilon_2 \leq \cdots \leq Adv_{P_5}^{ake}(\mathcal{A}) + \epsilon_5$

in which $\epsilon_1, \epsilon_2, \cdots, \epsilon_5$ are negligible values about k. We can acquire the advantage of adversary $\mathcal{A}$ by adding these negligible values with the success probability of online password guessing attack.

We assume q_{H_1}, q_{H_2} and $q_s + q_e$ are all ≥ 1. And we also assume when $H_1(\cdot)$ is made, the response is $as_h + 2e_h \in R_q$ by the simulator, in which s_h, e_h are all sampled uniformly at random from R_q.

Now, we describe the sequence of these protocols. At the end, we compute the advantage of $\mathcal{A}$.

Protocol P_0: This protocol corresponds to the real execution of our proposed 3PAKE protocol in the random oracle model.

Protocol P_1: In this protocol, we simulate the random oracles H_1, H_2 by maintaining hash lists $\wedge_{H_1}$ and $\wedge_{H_2}$.

On hash query $H_1(m)$, if there is a record (m, r) in the list $\wedge_{H_1}$, return r. Otherwise, choose an element $r \in R_q$, add (m, r) to the list $\wedge_{H_1}$, and return r.

On hash query $H_2(m)$, if there is a record (m, r) in the list $\wedge_{H_2}$, return r. Otherwise, select a string $r \in \{0, 1\}^k$, add (m, r) to the list $\wedge_{H_2}$, and return r.

Other oracles are simulated as in the real attack.

Claim 1. For adversary $\mathcal{A}$,

$$Adv_{P_0}^{ake}(\mathcal{A}) = Adv_{P_1}^{ake}(\mathcal{A})$$

Proof. In the proposed protocol, $H_1()$, $H_2()$ act as random oracles and the adversary cannot distinguish output of hash functions from random string.

Protocol P_2: P_2 is identical to P_1 except the collision event happens on the partial transcripts (R_A, Y_A) or (R_B, Y_B) and also on hash values. As we assume there is at least one honest party in protocol and thus one of R_A or Y_A (resp. R_B or Y_B) is uniformly distributed. According to the the birthday paradox, we can get:

Claim 2. For adversary $\mathcal{A}$,

$$Adv_{P_1}^{ake}(\mathcal{A}) \leq Adv_{P_2}^{ake}(\mathcal{A}) + \frac{q_{H_1}^2 + q_{H_2}^2 + (q_e^2 + q_s^2)}{q^n}$$

Proof. The cardinal of ring element R_q is q^n. According to the birthday paradox, we know the probability of the transcripts above happens before is $\frac{q_{H_1}^2+q_{H_2}^2+(q_e^2+q_s^2)}{q^n}$.

Protocol P_3: P_3 is identical to P_2 except the adversary gets the session key before a corrupt instance occurs.

Claim 3. For adversary $\mathcal{A}$,

$$Adv_{P_2}^{ake}(\mathcal{A}) \leq Adv_{P_3}^{ake}(\mathcal{A}) + 2Adv_{R_q}^{DRLWE}(t^{'}, q_{ro}) + 2Adv_{R_q}^{PWE}(t^{'}, q_{ro})$$

Proof. If the adversary gets the session key before a corrupt instance occurs, we can construct an algorithm D that solves PWE via running $\mathcal{A}$ on a simulation of the protocol. For (a, X, Y, W), D simulates P_2 with the following changes for $\mathcal{A}$:

1. In an Execute($\prod_A^i, \prod_B^j, \prod_S^k$) query, D sets $R_A = X + (as_f + 2e_f)$ in which $s_f, e_f \leftarrow R_q$, $R_B = Y + (as_{ff} + 2e_{ff})$ in which $s_{ff}, e_{ff} \leftarrow R_q$, and selects $\omega_B = \{0,1\}^n$.

2. When $\mathcal{A}$ finishes and correctly guesses the bit b after the Test($\prod_U^i$) query. Then the simulator D can compute,

$$\begin{aligned}
K_B &= R_A(s_y + s_{ff}) \\
&= (X + as_f + 2e_f)(s_y + s_{ff}) \\
&= Xs_y + (as_f + 2e_f)s_y + (X + as_f + 2e_f)s_{ff} \\
&\approx Xs_y + Ys_f + (X + as_f + 2e_f)s_{ff}
\end{aligned}$$

So,

$Xs_y = K_B - Ys_f - (X + as_f + 2e_f)s_{ff}.$

And,

$\sigma = \mathbf{Mod_2}(K_B - Ys_f - (X + as_f + 2e_f)s_{ff}, \omega)$

Finally, σ is added to the list of possible values for $\tau(X, s)$.

The simulation sets $R_A = X + (as_f + 2e_f)$ instead of the actual $R_A = (as_f+2e_f)$. These two are distinguishable except someone can solve the decision ring learning with error (DRLWE) problem. Thus P_3 is indistinguishable from P_2 until $\mathcal{A}$ can distinguish the session key from random string or DRLWE can be solved with non negligible advantage. If these happen, P_3 will be distinguishable from P_2. However, we still assumes $\mathcal{A}$ runs the appropriate time and query bounds.

Protocol P_4: P_4 is identical to P_3 except the adversary guesses the password offline.

Claim 4. For adversary $\mathcal{A}$,

$$Adv_{P_3}^{ake}(\mathcal{A}) \leq Adv_{P_4}^{ake}(\mathcal{A}) + Adv_{R_q}^{DRLWE}(t^{'}, q_{ro}) + 2Adv_{R_q}^{PWE}(t^{'}, q_{ro})$$

Proof. If this event happens, we can construct an algorithm D solving P-WE via running $\mathcal{A}$ on a simulation of the protocol. For (a, X, Y, W), D simulates P_3 with the following changes for $\mathcal{A}$:

1. In an Execute($\prod_A^i, \prod_B^j, \prod_S^k$) query, D sets $Y_A = X + (as_f + 2e_f)$ in which $s_f, e_f \leftarrow R_q$, $R_A = Y + (as_{ff} + 2e_{ff})$ in which $s_{ff}, e_{ff} \leftarrow R_q$,and selects $\omega_B = \{0, 1\}^n$. In an H_1 query returned $-\gamma^{'} = as_h + 2e_h \in R_q$.

2. When $\mathcal{A}$ finishes and correctly guesses the password offline. Then the simulator D can compute,

$$\begin{aligned}
R_{AS1} &= R_{S1}(s_y + s_{ff}) \\
&= [X + a(s_f - s_h) + 2(e_f - e_h)](s_y + s_{ff}) \\
&= Xs_y + [a(s_f - s_h) + 2(e_f - e_h)]s_y + [X + a(s_f - s_h) + 2(e_f - e_h)]s_{ff} \\
&\approx Xs_y + Y(s_f - s_h) + [X + a(s_f - s_h) + 2(e_f - e_h)]s_{ff} \\
&= Xs_y + Y(s_f - s_h) + [X + \gamma^{'} + a(s_f + 2e_f)]s_{ff}
\end{aligned}$$

So,

$$Xs_y = R_{AS1} - Y(s_f - s_h) - [X + \gamma^{'} + a(s_f + 2e_f)]s_{ff}.$$

And,

$\sigma = \mathbf{Mod_2}(R_{AS1} - Y(s_f - s_h) - [X + \gamma^{'} + a(s_f + 2e_f)]s_{ff}), \omega)$

Finally, σ is added to the list of possible values for $\tau(X, s)$.

The simulation sets $R_A = X + (as_f + 2e_f)$ instead of it being $R_A = (as_f + 2e_f)$ which is distinguishable if someone can solve the DRLWE problem. As a result P_4 is indistinguishable from P_3 until the adversary guesses the password offline or DRLWE is solved with non negligible advantage. In former case, D adds the correct $\tau(X, s)$ to the list. And if it is the latter case, the simulation will be distinguishable from P_3. However, we suppose that $\mathcal{A}$ still follows the appropriate time and query bounds even if $\mathcal{A}$ distinguishes the simulation from P_3.

Protocol P_5: P_5 is identical to P_4 except the adversary guesses the password online.

Claim 5. For adversary $\mathcal{A}$,

$$Adv_{P_5}^{ake}(\mathcal{A}) \leq \frac{q_s}{|D|}$$

Proof. As the adversary can only guess the password online for q_s times in this protocol. Then the probability for the adversary to get the password is $\frac{q_s}{|D|}$.

From Claims 1-5, Theorem 1 is proven.

5.3. Security requirement analysis

We will now explain how the proposed 3PAKE protocol satisfies the requirements presented in Section 3.2.

Mutual authentication: In our proposed protocol, the server and the client authenticate each other via the shared password. Anyone who does not have the password PW_A cannot compute the right Y_A and α_{AS1}. Hence, he cannot be authenticated by the other client. Therefore, the proposed 3PAKE protocol provides mutual authentication between the server and the client.

Session key security: Since the protocol provides mutual authentication, any pairs of legitimate clients will establish the same session key with the involvement of the server. Unauthorized client will not be authenticated by the server and, therefore, unable to interact with other legitimate client. Thus, the proposed 3PAKE protocol provides session key security.

Know key security: The clients use an ephemeral key to establish the session key. Thus, a session key has no relation to other session key(s). If $\mathcal{A}$ acquires one session key, other session key(s) will still be secure. Therefore, the proposed 3PAKE protocol provides known key security.

Resilience to three classes of password guessing attacks: Since the proposed protocol provides mutual authentication, it is also secure against undetectable online password guessing attack and online guessing attack (e.g. generally in most implementations, a client account will be suspended after a number of failed attempts). For an adversary to guess the password offline, the adversary needs to know R_{S1} or R_{S1}, E_{AS1} or E_{BS2} to verify if the guess is right. Thus, such an offline guessing attack will not work in this protocol.

Forward secrecy: Since the protocol provides known key security and the session key is independent of the password, leakage of the password will not allow the adversary to acquire previously established session keys. However, the adversary can select the ephemeral key to establish the session key with the password. Therefore, the proposed 3PAKE protocol provides weak perfect forward secrecy.

Resilience to other various attacks: The proposed protocol can withstand user impersonation attack, modification attack and man-in-the-middle attack as follows:

• user impersonation attack: In the proposed protocol,without the shared password the adversary cannot solve Y_A and compute the right α_{AS1}. Then, the adversary will not be authenticated by the server and cannot launch user impersonation attack successfully. Therefore, the proposed 3PAKE protocol can withstand user impersonation attack.

• modification attack: In our proposed protocol, password is the only authentication factor. All participants with the shared password can respond accordingly by computing the right value. Even by modifying the message, the adversary will not be authenticated successfully in the next step by the client or the server. Therefore, this protocol can withstand modification attack.

• man-in-the-middle attack: From the above description, we know the proposed 3PAKE protocol provides mutual authentication and the adversary cannot launch the modification attack successfully. Therefore, this protocol is able to withstand man-in-the-middle attack.

6. Performance analysis

Table 1: Runtime of related operations

Operations	Time(in ms)
T_{smul}	0.002658
T_{pmul}	0.008738
$T_{pmuladd}$	0.009562
$T_{geterror}$	0.000010
T_{mpre}	0.000017
T_{Cha}	0.004603
T_{Mod_2}	0.004179
T_h	0.006
T_{sm}	3.797
T_{pa}	3.815

We implement the proposed protocol, and similar to the implementation in [1] we let $n = 1024$, $q = 12289$. We use LatticeCrypto with C for the implementation of the proposed protocol. To have a better understanding of the computation cost of the proposed protocol, we compare it with the bench protocol in [30] based on ECC. We choose $y^2 = x^3 + ax + b$ as the elliptic curve in [30]. These two protocols get the same security level of 3072-bits RSA algorithms. We use the MIRACL library [27] for implementation of [30]. The notations used in the performance evaluation is as follows:

• T_{smul}: runtime of a scalar multiplication with a ring element.

- T_{pmul}: runtime of the multiplication between two ring elements.
- $T_{pmuladd}$: runtime of multiplication between two ring elements and then adding an other ring element.
- $T_{geterror}$: runtime of sampling an error from a Gaussian distribution.
- T_{mpre}: runtime of map-to-ring-element hash function.
- T_{Cha}: runtime of the **Cha** function in the proposed protocol.
- T_{Mod_2}: runtime of the $\mathbf{Mod_2}$ function in the proposed protocol.
- T_h: runtime of a general hash function.
- T_{sm}: runtime of a scalar multiplication in the group of bench protocol.
- T_{pa}: runtime of a point addtion in the group of bench protocol.

Table 2: Runtime comparisons of the two protocols

Participant	bench protocol	ours
A	15.23	0.066694
B	15.23	0.071297
S	22.842	0.121611
total	53.302	0.259602

The program is executed on a 2.7GHz Intel(R) Core(TM) i5-3340M CPU and 8GB RAM computer with 64bit system. The actual runtime for these operations based on the average of 1000000 executions is presented in Table 1. In the proposed protocol, we do not include the computation of the server in computing $H_1(ID_A||S||PW_A)$ or $H_1(ID_B||S||PW_B)$ since these values are stored in the server in advance. A client (A or B) only needs to compute $H_1(ID_A||S||PW_A)$ or $H_1(ID_B||S||PW_B)$ once. Then, the runtime of client A is $T_{smul}+2\times T_{pmul}+T_{pmuladd}+2\times T_{geterror}+T_{mpre}+T_{Cha}+2\times T_{Mod_2}+4\times T_h =$ 0.066694 milliseconds, client B is $T_{smul}+2\times T_{pmul}+T_{pmuladd}+2\times T_{geterror}+T_{mpre}+2\times T_{Cha}+2\times T_{Mod_2}+4\times T_h = 0.071297$ milliseconds, server S is $2\times T_{smul}+2\times T_{pmul}+2\times T_{pmuladd}+4\times T_{geterror}+2\times T_{Mod_2}+4\times T_h = 0.121611$ milliseconds and the total time is 0.259602 milliseconds.

In the bench protocol in [30], the runtime of client A is $3 \times T_{sm} + T_{pa} + 4 \times T_h = 15.23$ milliseconds, client B is $3 \times T_{sm} + T_{pa} + 4 \times T_h = 15.23$ milliseconds, server S is $4 \times T_{sm} + 2 \times T_{pa} + 4 \times T_h = 22.842$ milliseconds and the total time is 53.302 milliseconds– see Table 2.

From Table 2 we can see, the runtime of the proposed protocol is much less than the bench protocol which shows the proposed protocol is computation-efficient.

7. Conclusions

Quantum computers are likely to be commercially available in the very near future, and this necessitates the design of post-quantum cryptographic algorithms such as 3PAKE protocols which have widespread application in our consumer technologies.

In this paper, we proposed a 3PAKE protocol based on the RLWE-PAK [13]. We demonstrated the security of the proposed protocol secure in the random oracle model. We also implemented and evaluated the proposed protocol to demonstrate its efficiency and utility in real-world deployment. This is the first lattice-based 3PAKE protocol to be presented in the literature, at the time of this research.

Future research includes extending the proposed protocol to a group-based setting (i.e. lattice-based group PAKE protocol). In addition, the security model we used in this paper is not designed for a quantum adversary. Thus, there is a need for research on Quantum Random Oracle Model (QROM), which allows the modeling of queries in a quantum computing environment.

References

[1] Alkim E, Ducas L, Pöppelmann T, Schwabe P. Post-quantum key exchange-a new hope. IACR Cryptology ePrint Archive 2015;2015:1092.

[2] Bellare M, Pointcheval D, Rogaway P. Authenticated key exchange secure against dictionary attacks. In: International Conference on the Theory and Applications of Cryptographic Techniques. Springer; 2000. p. 139–55.

[3] Bellare M, Rogaway P. Entity authentication and key distribution. In: Annual International Cryptology Conference. Springer; 1993. p. 232–49.

[4] Bellare M, Rogaway P. Provably secure session key distribution: the three party case. In: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing. ACM; 1995. p. 57–66.

[5] Bellovin SM, Merritt M. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In: Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium on. IEEE; 1992. p. 72–84.

[6] Chang CC, Chang YF. A novel three-party encrypted key exchange protocol. Computer Standards & Interfaces 2004;26(5):471–6.

[7] Chang TY, Hwang MS, Yang WP. A communication-efficient three-party password authenticated key exchange protocol. Information Sciences 2011;181(1):217–26.

[8] Chen HB, Chen TH, Lee WB, Chang CC. Security enhancement for a three-party encrypted key exchange protocol against undetectable on-line password guessing attacks. Computer Standards & Interfaces 2008;30(1):95–9.

[9] Choo KKR. Secure Key Establishment. volume 4 of *10*. Advances in Information Security, Springer, 2009.

[10] Choo KKR, Boyd C, Hitchcock Y. Errors in computational complexity proofs for protocols. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2005. Springer; 2005. p. 624–43.

[11] Choo KKR, Boyd C, Hitchcock Y. Examining indistinguishability-based proof models for key establishment protocols. In: International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2005. Springer; 2005. p. 585–604.

[12] Chung HR, Ku WC. Three weaknesses in a simple three-party key exchange protocol. Information Sciences 2008;178(1):220–9.

[13] Ding J, Alsayigh S, Lancrenon J, Saraswathy R, Snook M. Provably secure password authenticated key exchange based on rlwe for the post-quantum world. In: Cryptographers Track at the RSA Conference. Springer; 2017. p. 183–204.

[14] Ding Y, Horster P. Undetectable on-line password guessing attacks. ACM SIGOPS Operating Systems Review 1995;29(4):77–86.

[15] Gentry C, Peikert C, Vaikuntanathan V. Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. ACM; 2008. p. 197–206.

[16] Gong B, Zhao Y. Small Field Attack, and Revisiting RLWE-Based Authenticated Key Exchange from Eurocrypt'15. Technical Report; Cryptology ePrint Archive, Report 2016/913, 2016. http://eprint. iacr. org/2016/913; 2016.

[17] Katz J, Vaikuntanathan V. Smooth projective hashing and password-based authenticated key exchange from lattices. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer; 2009. p. 636–52.

[18] Kim HS, Choi JY. Enhanced password-based simple three-party key exchange protocol. Computers & Electrical Engineering 2009;35(1):107–14.

[19] Krawczyk H. Hmqv: A high-performance secure diffie-hellman protocol. In: Annual International Cryptology Conference. Springer; 2005. p. 546–66.

[20] Lee TF, Hwang T, Lin CL. Enhanced three-party encrypted key exchange without server public keys. Computers & Security 2004;23(7):571–7.

[21] Lin CL, Sun HM, Hwang T. Three-party encrypted key exchange: attacks and a solution. ACM SIGOPS Operating Systems Review 2000;34(4):12–20.

[22] Lin CL, Sun HM, Steiner M, Hwang T. Three-party encrypted key exchange without server public-keys. IEEE Communications letters 2001;5(12):497–9.

[23] Lu R, Cao Z. Simple three-party key exchange protocol. Computers & Security 2007;26(1):94–7.

[24] Micciancio D, Regev O. Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing 2007;37(1):267–302.

[25] Nam J, Paik J, Kang HK, Kim UM. An off-line dictionary attack on a simple three-party key exchange protocol. IEEE Communications Letters 2009;13(3):205–7.

[26] Regev O. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) 2009;56(6):34.

[27] Scott M. Miracl-a multiprecision integer and rational arithmetic c/c++ library. shamus software ltd. 2013.

[28] Steiner M, Tsudik G, Waidner M. Refinement and extension of encrypted key exchange. ACM SIGOPS Operating Systems Review 1995;29(3):22–30.

[29] Sun HM, Chen BC, Hwang T. Secure key agreement protocols for three-party against guessing attacks. Journal of Systems and Software 2005;75(1):63–8.

[30] Wu S, Pu Q, Wang S, He D. Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol. Information Sciences 2012;215:83–96.

[31] Zhang J, Zhang Z, Ding J, Snook M, Dagdelen Ö. Authenticated key exchange from ideal lattices. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer; 2015. p. 719–51.