*Journal of Computer Security and Data Forensics* ©2021 ISSN xxxx-xxxx
Volume **1**, Number **1**, March **2021**

# Password-based authenticated key exchange from lattices for client server model

Jen-Chien Hsu[1], Yi-Siou Jheng[1], Sk Md Mizanur Rahman[2], Raylin Tso[1,*]

[1]Department of Computer Science,
National Chengchi University,
64, Sec. 2, Zhi-nan Rd., Taipei 11605, China

[2]Information and Communication Engineering Technology (ICET),
School of Engineering Technology and Applied Science (SETAS),
Centennial College, Toronto, ON M1G 3T8, Canada

Received November 2020; revised February 2021
(Communicated by: raylin@cs.nccu.edu.tw)

Abstract. *The password-based authenticated key exchange is a technology that allows both parties to perform mutual authentication and generate a shared session key. They through the shared password as the basis for authentication and generate a session key that is only known by both parties. At last, they can use this key to establish a secure channel to transmit secret message. We propose a password-based authenticated key exchange from lattices for Client-Server model. The client only need to remember the password rather than the private key, and the server except keep the password and its own public/private key pair. Both parties execute the mutual authentication via the shared password and accomplish the key exchange within two steps. The security of our protocol is based on LWE problem for lattices, so it is secure even an attacker uses a quantum computer.*
**Keywords:** Client/Server model, Key exchange, Lattice cryptosystem, LWE, Mutual authentication.

## 1. Introduction.

### 1.1. Research Background.

With the booming development of computer technology, people can carry out sundry services via he Internet such as file transfer, on-line shopping, cloud computing and others. The Client-Server model is frequently adopted among above services which clients are served by the great computational power and storage capability of remote central servers. Under such framework, the servers usually have powerful calculate capability and enormous storage space so that the clients can reduce the computational costs and save the memory capacity by outsourcing them to servers. Because the network is an opening environment, in order to ensure that the confidential information is not stolen by malicious attackers during the communication processes, two basic secure requirements must be satisfied: the authentication of identity and the confidentiality of the transmitted data. Identity authentication means that the participants in the communication can verify the legality of each other so that illegal participants will be ruled out. The confidentiality means that except for the legal participants of the communication, other people can't get the secret from the transmitted data.
Several cryptosystems like digital signature and other technologies are necessary in order

to achieve above secure requirements. In general, cryptosystem can be divided into symmetric cryptosystem and public key cryptosystem. Symmetric cryptosystem is that both parties of the communication use the same secret key to perform encryption and decryption, in this case, if the key was stolen then the protected information will be known to the attacker. In order to solve this problem, the public key cryptosystem was proposed. Public cryptosystem means that both parties of the communication have their own public/private key pair, the public key is published while the secret key is kept privately. Assuming Alice wants to send a secret message to Bob, she needs to use Bob's public key to encrypt the message, then Bob decrypts the encrypted message. Because the public key cryptosystem is less efficient in the operation, so the applications usually use it to transmit the session key. The symmetric cryptosystem is mainly adopted while choosing a cryptosystem. Meanwhile, the key exchange protocols play a vital role in the symmetric cryptosystem because they are designed to establish a shared secure session key which will be utilized for further encryptions for both sides of the communication.

In 1976, Whitfield Diffie and Martin Hellman proposed the famous Diffie-Hellman key exchange protocol[15]. It is a well-known pioneer which provides secure and reliable key exchange. However, the user authentication is not included in the original design so that it unavoidably faces the man-in-middle attack. Therefore, authenticated key exchange (AKE) schemes were proposed[5]. In AKE schemes, two parties aim to authenticate each other and establish a shared session key if the authentication successes.

In the current network, most services such as e-mail system, on-line shopping, social networking etc authenticate users through the password-based system. The aforementioned authentication mechanism combined with the key exchange is called the password-based authenticated key exchange (PAKE)[4, 8, 3]. It is different from the AKE using public key certificate, the both parties only need to use the shared password to accomplish the mutual authentication. This method can reduce the additional burden on issuance and management of certificate, the user can freely set the password that is easy to remember by himself/herself and use the password to establish a highly secure mutual authentication channel.

In 2012, Xun et al.[3] proposed a password-based authenticated key exchange in the Client-Server model. The users under the protocol only need to remember the password rather than the private key. Both parties execute mutual authentication via the password and generate a shared session key. The security of their protocol is guaranteed by the discrete logarithm problem in the number theory. However, a mathematician Peter Shor proposed an algorithm in 1994 which can solve the difficult problem from number theory in polynomial time by the quantum computer in the future since the quantum computers are able to effectively execute a great amount of computations in parallel, it means that current cryptosystems and secure protocols will be broken. Many scholars began to study the post-quantum cryptography to find the solutions, it was found that the lattice-based cryptosystems[9, 12] cannot being computed in parallel so that they are secure even in front of the quantum computer.

In 2013, Park et al.[13] proposed a mutual authentication mechanism based on NTRU cryptosystem[12]. NTRU is a lattice-based public key cryptosystem whose security is based on the Shortest vector problem over lattices. This mechanism is built on the NFC mobile payment environment, it make banks and customers perform the mutual authentication to verify each other's identity to ensure the security of the transaction phase. However, Tso et al.[14] proposed an attack method for Park's mechanism. In addition to the registration phase, the other phases are performed in an opening environment. An attacker can firstly eavesdrop the communication and get the message, including the certificate issued by the bank and the authenticated parameters based on NTRU. Then, the

attacker can launch an impersonation attack to pass the verify from legal participants. In this mechanism, in addition to the lack of security, the participants didn't generate a shared session key.
In 2012, Ding et al.[1] proposed a lattice-based key exchange. The security of protocol is based on the LWE problem on lattices, so it can resist the attacks from the quantum computers. They designed a method which can let similar parameters turn into the same value and then use the value to establish a shared session key. Nevertheless, this protocol could not provide mutual authentication, so it only can achieve passive security but still may be attacked by man-in-the-middle or other active attacks.
In order to improved Ding's method, Zhang et al.[10] proposed a new lattice-based key exchange with authentication in 2015. They continuously employed the method of shared parameters to establish the shared session keys and added an authentication mechanism which utilizes participant's public/private key to accomplish implicit key authentication. However, we advocate the clients of Client-Server model do not need to keep his/her own public/private key pair, it can be divided into the following three cases:

1. There is no public/private key pair on both parties: Because both parties have no public/private key that can represent their identity, so they can't perform authentication by encryption, decryption and certificates. Therefore, they need a fair third party to assist authentication between them. This method will add extra steps to reduce the overall efficiency.
2. Both parties have their own public/private key pair: They can use the public key of each other to encrypt the message, and verify each other by the key certificate. However, the client of C/S model in different service needs to generate different key pairs, so he/her needs to spend additional resources on the key storage and management.
3. Only the server has the public/private key pair: The server usually has powerful computational capability and enormous storage space and it must need to communicate with many different users, so the server keeps its own key pair is reasonable. The user just uses the shared password which is registered by the user to perform mutual authentication, he/she doesn't need to keep its own public/private key. This method can reduce the additional burden on management of key pair for client.

Our paper proposes a password-based authenticated key exchange based on lattice in the client/server model. This scheme has the following characteristics:

1. The security of our scheme is based on learning with error(LWE) problem for lattice, which belongs to the application of post-quantum cryptography so it can resist the threat from quantum computers.
2. The key exchange only needs a two-pass communication.
3. We adopt the client/server model so that there is no need for clients to keep his/her own public/private key pair. This method can reduce the additional burden on management of key pair for client.
4. Client can set the password that is easy to remember by himself/herself and use the password to establish a highly secure mutual authentication scheme.
5. We utilize the explicit key authentication so that the participants of communication can perform the mutual authentication.

In our scheme, client and server use the password which is registered by the user to perform mutual authentication. The client just uses server's public key to generate the secret authentication message, he/she does not need to keep his/her own public/private key pair. This method belongs to the explicit key authentication, participants can directly

know communicate with whom and ensure both of them have the same shared session key after finish the key exchange. According to the aforementioned characteristics, our scheme is an efficient, secure password-based authenticated key exchange in lattices.

1.2. **Organization.** This paper is divided into several sections, the contents of which are as follow:

1. This section introduces the research background, research motivation and propose of this paper.
2. This section introduces the background knowledge, including lattice, learning with error problem, key exchange and NTRU.
3. This sections introduces several previous related literatures about PAKE.
4. This section introduces our PAKE scheme in lattices in client/server model.
5. This section performs security and feature analyses of the proposed protocol.
6. This section introduces the conclusion.

2. **Preliminaries.** The security of our scheme is based on learning with error problem from lattice, and use the lattice based public key cryptosystem NTRU. In this section, we will introduce the lattice, learning with error problem and NTRU and the definition of implicit key authentication and explicit key authentication.

2.1. **Lattice.** A lattice $\mathcal{L}$ in $\mathbb{R}^n$ is a subgroup of the additive group $\mathbb{R}^n$, or equivalently, $\mathcal{L} = \{a_1v_1 + \cdots + a_nv_n | a_1, \cdots, a_n \in \mathbb{Z}\}$ for some $\mathbb{R}$-basis $v_1, \cdots, v_n$ of $\mathbb{R}^n$. Many lattice based cryptosystem schemes are secure based on some lattice hard problems. These hard problems are shortest vector problem(SVP), closet vector problem(CVP) and learning with error(LWE).
**Shortest Vector Problem(SVP):** Let define $\lambda(\mathcal{L}) = \min_{v \in \mathcal{L} \setminus \{0\}} ||v||_N$ be the length of the shortest non-zero vector in the lattice $\mathcal{L}$, where $\mathcal{L}$ in $\mathbb{R}^n$. The SVP is that of determining the $\lambda(\mathcal{L})$.
**Closet Vector Problem(CVP):** The CVP is that of determining the vector $v \in \mathcal{L}$ that is closet to a given non-zero lattice vector $w \in \mathcal{L}$.
**Learning with Error(LWE):** The LWE problem was introduced by Oded Regev in 2005[11]. Regev showed that the LWE problem is as hard to solve SVP and CVP, therefore, many lattice based cryptosystem are secure assuming as the LWE problem.
Given a parameter $\geq 1$, a modulus $q \geq 2$, and an error probatility distribution $\chi$ on $\mathbb{Z}_q$. Choosing uniformly random a vector $a \in \mathbb{Z}_q^n$ and choosing $e \in \mathbb{Z}_q$ according to $\chi$, then let $a \in \mathbb{Z}_q^n$ $A_{s,\chi}$ on $\mathbb{Z}_q^n \times \mathbb{Z}_q$jbe the probability distribution and outputs $(a, \langle a, s \rangle + e)$, where additions are performed in $\mathbb{Z}_q$. The LWE problem is that given several arbitrary number of independent sample from $A_{s,\chi}$. The LWE problem has two type, search method and decision method. The search method is finding the secret $s$. And the decision method is distinguishing the sample where from $A_{s,\chi}$ or uniformly random in $\mathbb{Z}_q^n \times \mathbb{Z}_q$. **Ring learning with Error(RLWE):** The RLWE problem is similar the LWE problem except the RLWE problem is defined on polynomial ring. Given an integer coefficients polynomial ring $\mathcal{R}_q = \mathbb{Z}_q/f(x)$, $q$ is prime integer, $f(x)$ is a irreducible polynomial and $\chi$ is the distribution on $\mathcal{R}_q$. Then we can define the RLWE problem like LWE problem by following formula: $(a_i(x), \langle a_i(x), s_i(x) \rangle + e)$.

2.2. **NTRU Cryptosystem.** NTRU (N-th degree truncated polynomial ring) is a public key cryptosystem[12], which selected integer coefficient polynomial from polynomial ring $\mathcal{R} = \mathbb{Z}[x]/(x^N - 1)$ to perform addition, multiplication and modulo operation of the polynomial coefficients, so it has a very fast speed. The security of NTRU is based

on SVP(Shortest Vector Problem) of lattices, so it can resistant to attack by quantum computers.

We assume that Alice wants to send encrypted message to Bob, the entire encryption and decryption process can be divided into four steps:

1. **Initial Parameter Setting:**
   In this phase, Bob sets up three integers $(N, p, q)$ and four integer coefficients polynomial sets $L_f, L_g, L_r, L_m$ from $\mathcal{R} = \mathbb{Z}[x]/(x^N - 1)$.
   - $N$ is prime and define the degree of polynomial $N - 1$.
   - $p, q$ need not be prime but need be coprime.
   - $q > p$.
2. **Key Generation:**
   To generate key pair, Bob dose the following step
   (a) Bob chooses two random polynomials $f \in L_f, g \in L_g$, $f$ must satisfy the requirement that it have inverse modulo $q$ and $p$ such that $f_q \cot f = \mod q$ and $f_p \cdot f = 1 \mod q$.
   (b) Bob computes its public key $h = f_q \cdots g \mod q$ and the private key is $(f, f_p)$.
3. **Encryption:**
   Alice use Bob's public $h$ to encrypt a message $m \in L_m$. Alice chooses randomly a polynomial $r$ with small coefficients. Then encrypting the message m to computes the ciphertext $c = r \cdot h + m \mod q$
4. **Decryption:**
   Bob uses its private key to decrypt the ciphertext $c$ to recover the message $m$.
   (a) Compute $a = f \cdot c \mod q$.
   (b) Compute $b = a \mod p$.
   (c) Compute $m = f_p \cdot b \mod p$.

2.3. **Key Exchange.** Key exchange is a protocol which can make both parties in communication share a secret session key. This session key cannot be known to anyone other than the participants. The famous key exchange was proposed by Whitfield Diffie and Martin Hellman in 1976(Diffie-Hellmen key exchange), it became the basis of many key exchange protocol. However, the authentication mechanism was not provided in the D-H key exchange, so it may be attacked by man-in-the-middle or other active attacks. Therefore, a secure key agreement must ensure that people other than the participants are unable to know the parameters of the generated key during the exchange process, and that all participants have the same key after finish the key exchange. The following basic security requirements must be satisfied:

- **Known Session Key Security:** Even if an attacker can intercept the current session key, he can't figure out the key used in the past and the key that will be generated in the future.
- **Forward and Backward Secrecy:** Even if an attacker can intercept the password, the keys that are produced before are remained secure.
- **Off-line dictionary attack resistance:** It does not leak any information of password from the public messages.
- **Impersonation attack:** An attacker could not fake a legal user to communicate with another person.
- **Man-in-the-middle attack resistance:** An attacker can't relays and alters the communication between two parties who believe they are directly communicating with each other.

2.4. **Implicit Key Authentication v.s. Explicit Key Authentication.** The implicit key exchange is to assume that the identity of participants in communication are legal, they only need to follow the execution of protocol then they can get a shared session key. The famous DH that is a kind of implicit key exchange, but it doesn't provide authentication mechanism, so the participants can't verify the identity of each other and they may face the Man-in-Middle attack. In the above related work, Zhang's[10] protocol also belongs to the implicit key exchange, but it has the authentication. Their protocol didn't transmit the information about identity, but in session key generation, the key parameter $K_j = (p_i c + X_i)(s_j d + rj) + 2cg_j$ and $K_i = (p_j d + Y_j)(s_i c + r_i) + 2dg_i$ including the public key of each other $p = as + 2e$, and s is the corresponding private key. An attacker intercepts the public key to forge a message to perform the impersonation attack, but it didn't know the private key s, so it can't get the correct key parameter and the session key. However, the transmitted message didn't have the information about identity of participants, so they can't directly know communicate with whom. This method requires additional steps to ensure both of participants have the same shared session key after finish the key exchange.
The explicit key authentication means that the participants of communication has transmit the information about identity, so they can use this information to verify the identity of each other. Compared to the implicit key authentication, the explicit authentication can directly know communication with whom. Our scheme belongs to explicit key authentication; the participants transmit the authenticated message including the shared password.

3. **Related Works.** In this section, we will introduce the lattice-based applications, several key exchange studies.

3.1. **Identity-based Password-Authenticated Key Exchange for Client/Server Model.** In 2012, Xun et al.[3] proposed an identity-based password-authenticated key exchange for client/server mode. The user under the protocol is only required to remember the password and the server except keep the password and its own public/private key. Both parties execute mutual authentication via the password and generate a shared session key in the last. The security of their protocol is guaranteed by the discrete logarithm problem in number theory, which is easily attacked by a quantum computer in future.

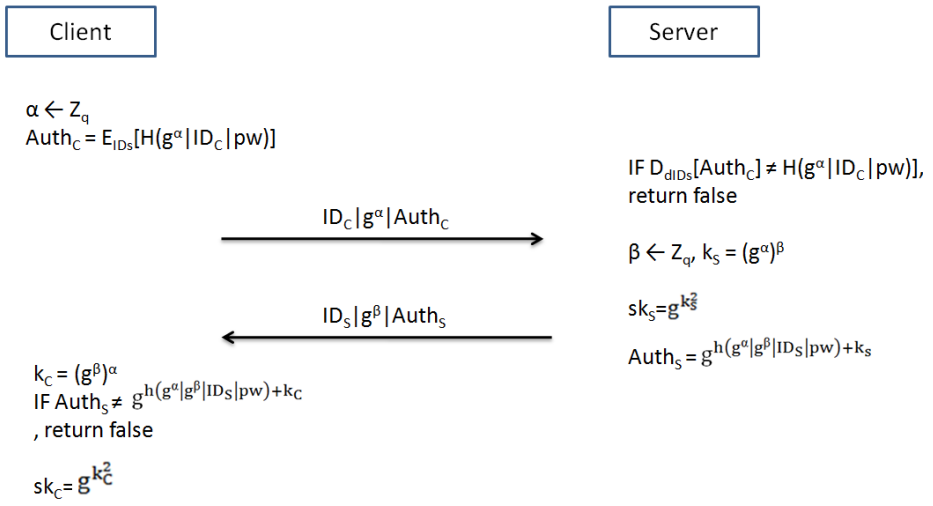


FIGURE 1. Xun's Protocol

Before the protocol execution, both parties share a public parameter $g$.

1. The client randomly select secret parameters $\alpha$ to compute the $g^{\alpha}$. After that, it employs server's public key $pk_{\mathrm{S}}$ to make authentication message $\mathrm{Auth_C} = E_{\mathrm{ID_s}}[\mathrm{H}(g^{\alpha} \mid \mathrm{ID_C} \mid \mathrm{pw})]$. Then send $\mathrm{Auth_C}$, $\mathrm{ID_C}$ and $g^{\alpha}$ together to the server.
2. After receiving $\mathrm{Auth_C}$, the server shall employ its won private key $sk_{\mathrm{S}}$ for decryption to verify the hash value. After passing the verification, the server shall randomly select secret parameter $\beta$ to compute the $g^{\beta}$, then it can compute the shared session key $sk_{\mathrm{S}} = g^{k_{\mathrm{S}}^2}$. Finally, the server make a authentication message $\mathrm{Auth_S} = g^{h(g^{\alpha}|g^{\beta}|\mathrm{ID_S})|\mathrm{pw})+k_{\mathrm{S}}}$ then sedn it back the client with $\mathrm{ID_S}, g^{\beta}$.
3. After receiving $\mathrm{Auth_S}$, the client firstly verifies it. If the verification is successful, the client can compute the shared session key $sk_{\mathrm{C}} = g^{k_{\mathrm{C}}^2} = g^{k_{\mathrm{S}}^2} = sk_{\mathrm{S}}$ to complete the authenticated key exchange of both parties.

3.2. **Anonymous Authentication Scheme based on NTRU for the Protection of Payment Information in NFC Mobile Environment.** In 2013, Park et al.[13] proposed a mutual authentication mechanism based on NTRU cryptosystem. This mechanism is built on the NFC mobile payment environment, it makes bank and customer perform the mutual authentication to verify each other identity to ensure the security of the transaction phase. In addition to verify the identity of customers, but also to allow customers to verify the bank. The authentication process can be divided into four steps:

1. **System Parameters Setting:**

   - $A$: NFC mobile payment user.
   - $B$: Bank.
   - $\mathcal{R}, L_f, L_g, f, g, f_p, f_q, p, q, g_p, g_p$: NTRU parameters.
   - $v$: Public key polynomial.
   - $I$: User identity.
   - Cert: Certificate generated by the certificate authority.
   - H:Secure hash function.
2. **User Registration Phase:**
   If a user wants to use the NFC mobile payment, he needs to register with bank.
   (a) The user picks $f_A, g_A$ from $L_f, L_g$, then computes inverse $f_{Aq}, g_{Aq}$ and public key $v_A = p \cdot f_{Ap} \cdot g_A \in \mathbb{Z}_q[x]/(x^N - 1)$.
   (b) User sends its identity informations $I_A$ and public key $v_A$ to the bank. Bank verifies the identity of the user, if the verification succeed, then bank will generate a certificate $\mathrm{Cert}(I_A, v_A)$. At last, bank returns $\mathrm{Cert}(I_A, v_A)$ to the user and stores the user information in the bank.
3. **User Identity Proof Phase:**

   (a) User picks a random polynomial $r_A$ and computes $x_A = g_A \cdot r_A$, then user transmitted $I_A$, $v_A$, $\mathrm{Cert}(I_A, v_A)$ and $x_A$ to the bank for authentication.
   (b) After receiving the data from step a, the bank verifies the integrity of $\mathrm{Cert}(I_A, v_A)$. If passed, the Bank picks a random polynomial $e_B$ from Le and sends it back to the user.
   (c) User computes $y_A = f_A \cdot r_A \cdots e_B$ and returns $y_A$ to the bank.
   (d) With $(v_A, x_A, y_A, e_B)$, the bank verifies the user by checking $y_A \cdot v_A = x_A \cdot e_B$ or not.

   If the user is legal, then the operation results are equal. Indicating that the user through the verification of the bank.
4. **Bank Identity Proof Phase:**

(a) Bank picks a random polynomial $r_B$ and computes $x_B = g_B \cdot r_B$, then he transmitted $e_B$, $I_B$, $v_B$, Cert$(I_B, v_B)$ and $x_B$ to the user for authentication.
(b) After receiving the data from step a, the user verifies the integrity of Cert$(I_B, v_B)$. If passed, the user picks a random polynomial $e_A$ from $L_e$ and sends it back to the bank.
(c) The bank computes $y_B = f_B \cdot r_B \cdot e_A$ and returns $y_B$ to the user.
(d) With $(v_B, x_B, y_B, e_A)$, the user can verifies the bank by checking $y_B \cdot v_B = x_B \cdot e_A$ or not.

If the bank is legal, then the operation results are equal. Indicating that the bank through the verification of the user and finish the mutual authentication.

3.3. **Security Analysis of a NTRU-based Mutual Authentication Scheme.** The mutual authentication mechanism proposed by the Park is based on NTRU cryptosystem, which allows mutual authentication between the user and the bank by the public/private key pair, certificate, authentication parameters and random polynomials. In the verify phase, both parties through the certificate to verify each other. The verification side checks the certificate in the first step, if passed then they continue to transmit the authentication parameters and do the final verification operation. Because the general certificate is not limited to use only once, which may intercept by attacker and perform the impersonation attack.

Tso et al.[14] proposed an attack method for Park's mechanism. Assume an attacker A' who can eavesdrop the communication between the user and the bank, then A' can have the following information $v$, $I$, Cert$(I,v)$, $x$, $e$, $y$. With these information, the attacker can launch an impersonation attack.

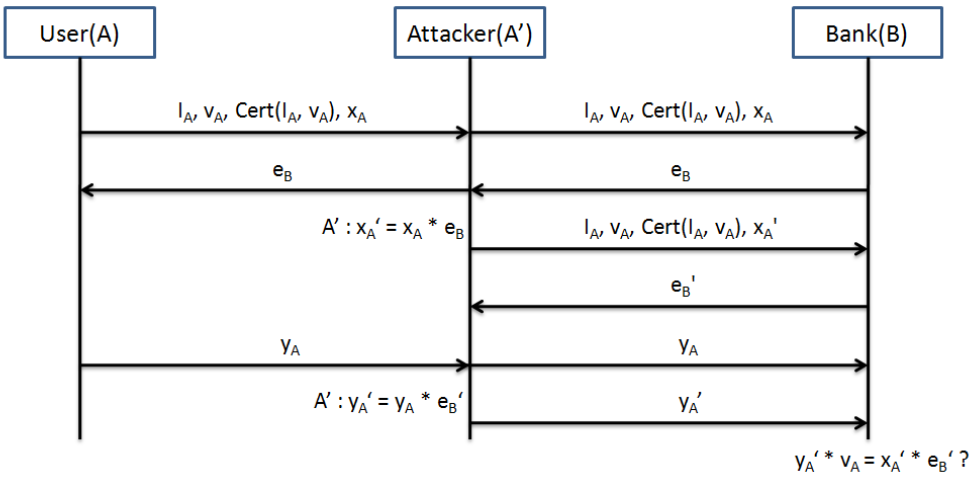


FIGURE 2. Impersonation Attack

1. The attacker A' computers $x_{A'} = x_A \cdot e_B$ and then sends $I_A, v_A$ Cert$(I_A, v_A)$ with $x_{A'}$ to the bank.
2. After receiving the data from step a, the bank will verify the integrity of Cert$(I_A, v_A)$. If passed, the bank will select a random polynomial $e_{B'}$ from $L_e$ and sends it back to the attacker.
3. The attacker A' computes $y_{A'} = y_A \cdot e_{B'}$ and returns it back to the bank.
4. With $(v_A, x'_A, y_{A'}, e_{B'})$, the bank authenticates the attacker by checking $y'_A \cdot v_A = x_{A'} \cdot e_{B'}$ or not.

In this way, the attack successfully passes the authentication from the bank and can impersonate as the victim user A. This may cause many serious problems when the scheme is used for mobile e-commerce.

3.4. **A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem.** In 2012, Ding et al.[1] proposed a lattice-based key exchange. The security of protocol is guaranteed on the LWE problem on lattices, so it can against the attack from the quantum computers. They also designed a method which can let two similar parameters turn into the same value and then use the value to establish a shared session key. Nevertheless, this protocol could not provide mutual authentication, so it may be attacked by man-in-the-middle or other active attacks.

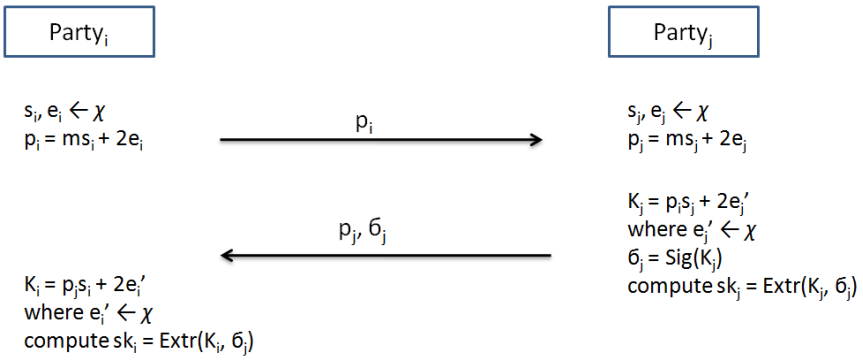


FIGURE 3. Ding's Protocol

1. Both parties share a public parameter $m$. Then Party$_i$ randomly selects secret parameters including $s_i, e_i$. After all, they compute $p_i = ms_i + 2e_i$ and sends $p_i$ to the other party.
2. After receiving the message from step a, Party$_j$ first compute $p_j = ms_j + 2e_j$by randomly selects secret parameters including $s_j$, $e_j$. Then it chooses a random parameter $e_j'$ to compute the key element $K_j = p_i s_j + 2e_j'$ with received $p_i$ and secret $s_j$.
3. The Party$_j$ inputs $K_j$ into signal function Sig() to get $\sigma_j$ =Sig$(K_j)$, and then inputs $K_j, \sigma_j$ into function Extr() to get the shared session key $sk_j$ =Extr$(K_j, \sigma_j)$. Finally, Party$_j$ sends $p_j, \sigma_j$ back to the Party$_i$.
4. Party$_i$ first chooses a random parameter $e_i'$ to compute the key element $K_i = p_j s_i + 2e_i'$ with received $p_j$ and secret $s_i$, then input $K_i$ and $\sigma_j$ into functions Extr() to get the shared key $sk_i$=Extr$(K_i, \sigma_j)$=Extr$(K_j, \sigma_j$=$sk_j)$ to complete the authenticated key exchange of both parties.

1. Both parties share a public parameter $a$. Party$_i$ randomly selects secret parameters including $r_i, f_i$, then compute $X_i = ar_j + 2f_i$ and sends $X_i$ to the Party$_j$.
2. After receiving the message from step a, Party$_j$ first computes $Y_j = ar_j + 2f_j$ by randomly selects secret parameters $r_j, f_j$. Then it chooses a random parameter $g_j$ to compute the key element $K_j = (p_i c + X_i)(s_j d + r_j) + 2cg_j$, where $c = H(i, j, X_i), d = H(i, j, X_i, Y_j)$.
3. The Party$_j$ inputs $K_j$ into signal function Cha() to get $w_j$=Cha$(K_j)$, and then inputs $K_j, w_j$ into function Mod2()to get the key parameter $\sigma_j$ =Mod2$(K_j, w_j)$. Finally , Party$_j$ computes the shared session key $sk_j = H(i, j, X_i, Y_J, w_j, \sigma_j)$ and sends $Y_j, w_j$ back to the Party$_i$.

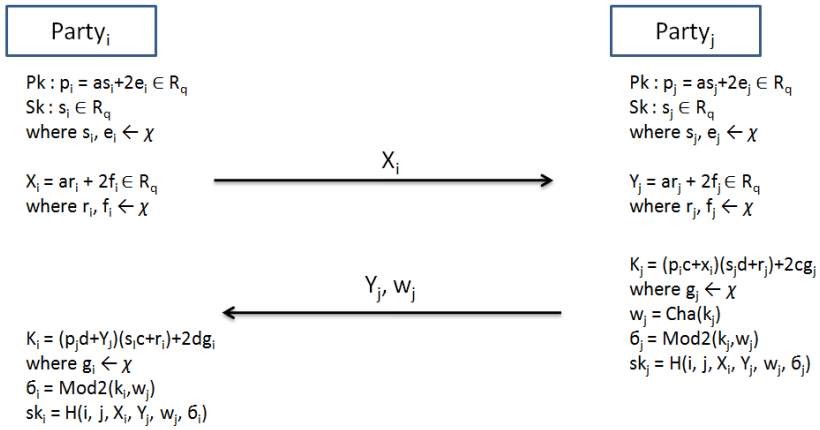


FIGURE 4. Zhang's Protocol

4. Party$_i$ first chooses a random parameter $g_i$ to compute the key element $K_i = (p_jd + Y_j)(s_ic + r_i) + 2dg_i$. Then inputs $K_i, w_j$ into Extr() function to get the key $sk_i = H(i, j, X_i, Y_j, w_j, \sigma_i) = H(i, j, X_i, Y_j, w_j, \sigma_j) = sk_j$ to complete the authenticated key exchange of both parties.

3.5. **Authenticated Key Exchange from Ideal Lattices.** In order to improved Ding's method, Zhang et al.[10] proposed a new lattice-based key exchange with authentication mechanism in 2015. Their protocol refers to the parameters extraction method proposed by the Ding and used participant's public/private key to accomplish implicit authentication. The implicit authentication refers to the achievement of authentication without the assistance with encryption or certificates. The security of Zhang's protocol is also based on LWE problem like Ding and added the authentication mechanism, so it not only can resistant to attack by quantum computer, but also resistant to passive and active attack. However, we think the clients of client-server model do not need to keep his/her own public/private key pair. They just use the shared password to execute the mutual authentication with Servers, this method can reduce the additional burden on management of key pair for client.

4. **Proposed Password-based Authenticated Key Exchange from Lattices.** In this section, we will introduce our scheme, which can be divided into the system parameters setting phase, protocol execution phase and the parameters correctness.

4.1. **Architecture.** We propose a password-based authenticated key exchange from lattices for Client-Server model. The client only has to remember the password shared with the server, and the server records the password in addition to its own public/private key pair. They just use the shared password to execute the mutual authentication and accomplish the key exchange within two steps. Our security is based on LWE from lattice and perform the explicit authentication with server's public key, so it can resist attacks from quantum computers.

4.1.1. *System Parameters Setting.*

- $n$ is a power number of 2.
- $q$ is a prime number greater than **8**, and satisfy $q \mod 2n = 1$.
- $\mathcal{R}_q$ is a polynomial ring with modular $q$ and $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$.
- $\chi$ is the Gaussian discrete distribution over $\mathcal{R}_q$.
- $\gamma$ is the standard deviation of $\chi$.

- pw is the password shared by Client/Server.
- g is the public parameters shared by both parties.
- $(\text{Pubkey}_s, \text{PriKey}_s)$are the public/private key pair of the Server.
- $\text{ID}_\text{C}, \text{ID}_\text{S}$ are the identity information of the Client and the Server.
- $\text{T}_\text{cur}$ is current time.
- TS is timestamp.

4.1.2. *Protocol Execution.*

**Step 1:** The client randomly selects secret parameters including $\text{f}_\text{C}$, $\alpha$ and Nonce from the Gaussian distribution $\chi$, then compute the authentication parameter $X = g\alpha + 2\text{f}_\text{C}$. After that, it employs $\text{pk}_\text{S}$ to make $\text{Auth}_\text{C} = \text{E}_{\text{pks}}[\text{H}(\text{X} \mid \text{ID}_\text{C} \mid \text{pw} \mid \text{Nonce} \mid \text{TS}_1), \text{Nonce}]$ . Then sends $\text{Auth}_\text{C}$, $\text{ID}_\text{C}$, X and $\text{TS}_1$ together to the server.
**Step 2.1:** The 2.1 to 2.3 is done by the server. After receiving $\text{Auth}_\text{C}$, the server firstly compares the timestamp $\text{TS}_1$ with current time $\text{T}_\text{cur}$, if the time gap is more than limit $\Delta\text{T}$, then the server rejects this request. After that , the sever employ its own private key $\text{sk}_\text{s}$ for decryption to verify the hash value and get Nonce. After passing the verification, the server shall randomly selects secret parameters $\beta$ and $\text{f}_\text{S}$ to compute the authentication parameter $Y = g\beta + 2\text{f}_\text{S}$.
**Step 2.2:** The server shall employ the received X, $\beta$ and the random number $r_\text{s}$ to compute $K_\text{S} = \text{X}\beta + 2r_\text{s}$. Then, the server inputs $K_\text{S}$ to get $w_\text{S} = Sig(K_\text{S})$ and in addition computes the key element $\rho_\text{S} = \text{Extr}(K_\text{S}, w_\text{S})$. Finally, the shared key is $sk_\text{S} = \text{H}(\text{ID}_\text{C} \mid \text{ID}_\text{S} \mid \text{X}|\text{Y} \mid w_s \mid \text{Nonce} \mid \rho_\text{S})$.
**Step 2.3:** The sever generates
$\text{Auth}_\text{S} = \text{H}(\text{Y} \mid \text{ID}_\text{S} \mid \text{pw} \mid w_\text{S} \mid \text{Nonce}+1 \mid \text{TS}_2)$ and then sends it together with $\text{ID}_\text{S}$, Y, $w_\text{S}$ and $\text{TS}_2$ back to the client.
**Setp 3:** After receiving $\text{Auth}_\text{S}$, the client firstly compares the timestamp $\text{Ts}_2$ with current time $\text{T}_\text{cur}$, if the time gap is more than limit $\Delta\text{T}$, then the client rejects this request. After that, the client verifies the $\text{Auth}_\text{S}$, if the verification is successful, the client shall select a random number $r_\text{C}$ to compute $K_\text{C} = \text{Y}\alpha + 2r_\text{C}$, then uses $K_\text{C}$, $w_\text{S}$ to get the key element $\rho_\text{C} = \text{Extr}(K_\text{C}, w_\text{S})$. In the end, they could get the key $sk_\text{C} = \text{H}(\text{ID}_\text{C} \mid \text{ID}_\text{S} \mid \text{X—Y} \mid w_s \mid \text{Nonce} \mid \rho_\text{C}) = sk_\text{S}$ to complete the authenticated key exchange of both parties.

4.1.3. *Password Update.* If the client wants to update the shared password, he/she can use the established session key to encrypt the new password then send to server. After receiving the encrypted message, the server uses shared session key for decryption to obtain the new password and record it, then both parties complete the update of password and need to revoke the current session key.

4.2. **Correctness.** Our protocol refers to the parameters extraction method proposed by the Ding et al. [1] to establish the shared secrete session key. This will be achieved via two functions including Sig and Extr. The text below will introduce the frameworks of Sig and Extr in detail.

4.2.1. *Signal Functions.* Suppose $q$ ia a prime number greater than 2, $\mathbb{Z}_q = \{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$, $E = \{-\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{q}{4} \rfloor\}$, then the Signal Function Sig(x) shall be defined as below:

$$Sig(x) = \begin{cases} 0, & x \in E \\ 1, & otherwise. \end{cases}$$

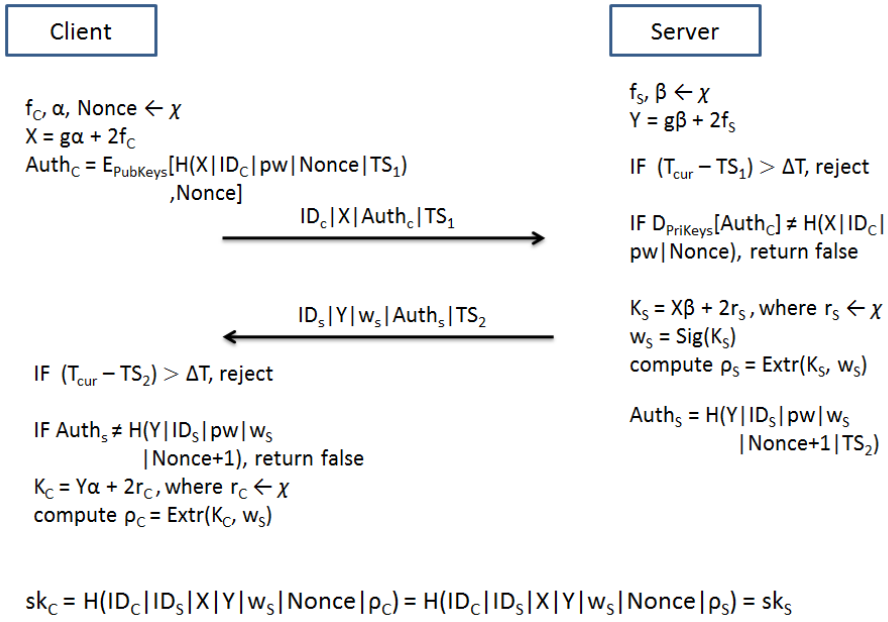


FIGURE 5. Our Scheme

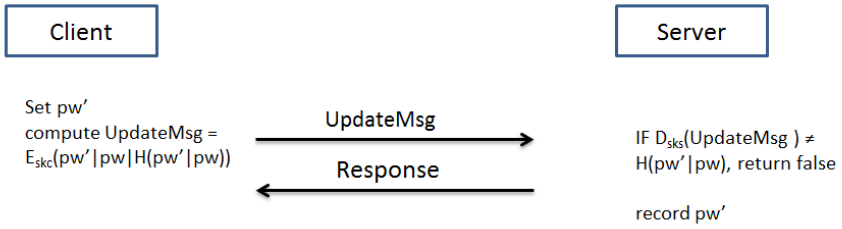


FIGURE 6. Password Update Phase

4.2.2. *Robust Extractors.* Suppose $q$ is an odd number greater than 8, then the function Extr() shall be defined as below:

$$\mathrm{Extr}(x,\sigma) = (x + \sigma \cdot \frac{q-1}{2} \mod q) \mod 2, \text{ where } \sigma = Sig(x)$$

Given two number $x, y \in \mathbb{Z}_q$, $x - y = 2\epsilon$ and $\mid 2\epsilon \mid \leq \frac{q}{4} - 2$, then the following equations shall be satisfied:

$$\begin{aligned}
\mathrm{Extr}(x,\sigma) = & (x + \sigma \cdot \tfrac{q-1}{2} \mod q) \mod 2 \\
& , \text{ where } \sigma = Sig(y) \\
= & (y + \sigma \cdot \tfrac{q-1}{2} + 2\epsilon \mod q) \mod 2 \\
= & (y + \sigma \cdot \tfrac{q-1}{2} \mod q) + 2\epsilon \mod 2 \\
= & (y + \sigma \cdot \tfrac{q-1}{2} \mod q) \mod 2 \\
= & \mathrm{Extr}(y,\sigma)
\end{aligned}$$

Now, we will proof two facts:

1. Why two similar values with error tolerance $2\epsilon \leq \frac{q}{4} - 2$ can trun into a same value.
2. How to select parameters to satisfies the error tolerance.

**Proof of the fact 1:**[1]
First, we need to limit $\mid y + \sigma \cdot \frac{q-1}{2} \mod q + 2\epsilon \mid \leq \frac{q}{2} - 1$, it means $y$ pluses $2\epsilon$ is still in the $\mathbb{Z}_q$. Second, it can follow the Sig definition to see that $\mid y + \sigma \cdot \frac{q}{2} \mod q \mid \leq \frac{q}{4} + 1$. According

to the above two inequalities to deduce $\mid y+\sigma\cdot\frac{q-1}{2} \mod q+2\epsilon \mid\leq \frac{q}{4}+1+\mid 2\epsilon \mid\leq \frac{q}{2}-1$, so we can proof $2\epsilon\leq\frac{q}{4}-2$.

**Lemma 1.**[2]
For any $s\geq\omega\sqrt{\log n}$, we have

$$\underset{x\leftarrow D_{\mathbb{Z}^n,s}}{Pr}[||x||>s\sqrt{n}]\leq 2^{-n}$$

**Proof of the fact 2:**
This proof is to show the correctness of the proposed scheme. Firstly, the key parameters $K_{\mathrm{C}}, K_{\mathrm{S}}$ of our scheme can be expanded as follows:

$$K_{\mathrm{C}}=\mathrm{Y}\alpha+2r_{\mathrm{C}}=(g\beta+2f_{\mathrm{S}})\alpha+2r_{\mathrm{C}}=g\alpha\beta=2(\alpha f_{\mathrm{S}}+r_{\mathrm{C}})$$
$$K_{\mathrm{s}}=\mathrm{X}\beta+2r_{\mathrm{S}}=(g\alpha+2f_{\mathrm{C}})\beta+2r_{\mathrm{S}}=g\alpha\beta=2(\beta f_{\mathrm{C}}+r_{\mathrm{S}})$$

Then we can see the error tolerance of $K_{\mathrm{C}}, K_{\mathrm{S}}$ is:

$$K_{\mathrm{C}}=K_{\mathrm{S}}+2(\alpha f_{\mathrm{S}}+r_{\mathrm{C}}-\beta f_{\mathrm{C}}-r_{\mathrm{S}})$$

Where the parameters $\alpha,\beta,f,r$ are selected from the distribution $\chi_\gamma$, so we can obtain the following inequality from Lemma 1:

$$||2(\alpha f_{\mathrm{S}}+r_{\mathrm{C}}-\beta f_{\mathrm{C}}-r_{\mathrm{S}})||\leq 8n\gamma^2<\frac{q}{4}-2$$

by selecting the parameters $n=\lambda, q=\lambda^4, \gamma=\lambda$, it will satisfy the above inequality.

5. **Security Analysis.** In this section, we refer to the password authenticated key exchange security model proposed by Xun et al[3], Bellare et al.[5] and Katz et al.[4] to analysis the security of our scheme. We first define some oracle which can be used by adversary to attack the communication, and then we can analysis whether vulnerabilities exist in our scheme. Next, we define the advantage of the adversary. Finally, we set up several different experiments to simulate and prove the security of our scheme.
The password of each session and the public/private key pairs of each server are independent and randomly generated in our scheme. Without lose the generality, we only proof for the communication between one user and a server. $C_i$ and $S_i$ are referred to the client and server in the i-th communication, that is, C and S are in state i.

5.1. **Oracle Definition.** Suppose that there is an adversary A can control all communication channels of the protocol and query following oracles:

**Send($U^i, M$):** Adversary can send the specified message to U, and intercept the message return message returned from U. (U can be client or server.)
**Execute($C^i, S^i$):** Adversary can let the user execute the i-th communication with server, then he can intercept the message from it.
**Corrupt($C$):** Adversary can get the previous password from client
**Corrupt($S$):** Adversary can get the private key from Server, it is equivalent to get the previous password from server.
**Reveal($U^i$)** : Adversary can get the session key from any side of communication parties.
**Test($U^i$):** Adversary can test the security of U by this oracle's output b, b is a random bit 0 or 1. If b = 0, the adversary gets the session key sk, and if b = 1 the adversary gets a random session key sk'.

5.2. **Advantage of the Adversary.** We can use Test($U^i$) output b to determine whether the adversary successfully attacks the i-th protocol, but U's state must be fresh, it means U will not be queried by certain specific oracle. So, we need to define the fresh state before experiments.

5.2.1. *Fresh Definition.* We define the state is fresh if two following conditions hold:

1. The adversary never queried Reveal(C) or Reveal(S) to get the session key.
2. The adversary never queried Reveal Corrupt(C) or Corrupt(S) to get the password from client or the private key from sever.

5.2.2. *Succ Definition.* In this section, we define the successful attack state of the adversary. Suppose that there is an adversary who tries to attack the protocol, it wins if all following rules are satisfied:

1. The adversary only queried Test($U^i$) once time in protocol.
2. When adversary queried Test($U^i$), the i-th protocol was done and the session key has been generated.
3. U is fresh.
4. The adversary queried Test($U^i$) output b' = b, it means adversary get the right session key.

If all rules above are satisfied, it is defined as Succ, and the success probability is described as $\mathrm{Pr}_A^p$[Succ].

5.2.3. *Experiment Definition.* We analyse the security of our protocol by providing oracle accesses to the adversary in the different experiment. In the following, we define several experiments to discuss, each experiment will follow the setting of the previous experiment and make some changes.

In each experiment, the advantage of the adversary can be expressed as $Adv_A^p$ (k) = $2Pr_A^p$ [Succ]-1. In general, the probability $Pr_A^p$ [Succ] equivalent to throwing a coin is $\frac{1}{2}$, it means no leakage any secret message in the communication and the key exchange is safe.

**Experiment** $P_0$: we define the real execution of the protocol as experiment $P_0$. The adversary can use any oracle to query our protocol.

**Experiment** $P_1$: In experiment $P_1$, an oracle generated message $\mathrm{msg_C} = (\mathrm{ID_C}, \mathrm{X}, \mathrm{Auth_C})$ or $\mathrm{msg_S} = (\mathrm{ID_S}, \mathrm{Y}, \mathrm{Auth_S})$ and the hash function needs to satisfy the following two properties:

1. The message $\mathrm{msg_C}$ or $\mathrm{msg_S}$ aren't repeated. This is when $\imath \neq j$, $\mathrm{msg_C^i} \neq \mathrm{msg_C^j}$ and $\mathrm{msg_S^i} \neq \mathrm{msg_S^j}$.
2. The hash function doesn't occur collision, it means that is a cryptographically collision-resistant hash function.

In $P_0$, there are no such properties but there is still very small chance that the message will be repeated and the collision of the hash function occurs. The parameters contained in msg were generated by $\mathrm{X} = g\alpha + 2f_{\mathrm{C}}$, $\mathrm{Y} = g\beta + 2f_{\mathrm{S}}$ where $\alpha, \beta, f$ are randomly selected in $\chi$, so the probability of repeated message is also close to 0, and the probability of a collision in the general hash function is also close to 0. So the adversary can't determine the difference between $P_0$ and $P_1$, then the advantage $|\mathrm{Adv}_A^{P_0}(k) - \mathrm{Adv}_A^{P_1}(k)|$ is negligible.

**Experiment** $P_2$: In experiment $P_2$, the adversary A queried oracle Execute($\mathrm{C}^i, \mathrm{S}^i$) to get the parameter $K_{\mathrm{S}} = \mathrm{X}\beta + 2r_{\mathrm{S}}$ is replaced with a random value $K_{\mathrm{S}'}$ from $\mathcal{R}_q$. If $K_{\mathrm{S}}$ is an LWE sample, then what A obtains exactly the same as in $P_1$; if $K_{\mathrm{S}}$ is uniformly random in $\mathcal{R}_q$, then what A obtains exactly the same as in $P_2$. This implies that if A can distinguish $P_1$ and $P_2$, then he can also solve the decision LWE. So we can define the advantage $|\mathrm{Adv}_A^{P_1}(k) - \mathrm{Adv}_A^{P_2}(k)|$ between $P_1$ and $P_2$ is negligible. Therefore, since $K_{\mathrm{S}'}$ is a random value, the $\rho_{\mathrm{S}'}$=Extr($K_{\mathrm{S}'}, w_{\mathrm{S}'}$) is also randomly. The session key sk' is generated by wrong $\rho_{\mathrm{S}'}$, that is, the generation of sk is not related to the correct $K_{\mathrm{S}}$.

**Claim:** If the adversary A queried oracle Corrupt($U^i$) to get the previous password $pw^{i-1}$, according to the above proof we can see the session key generation and password are independent, so the adversary is unable to obtain the current session key ski as well as to obtain the new password $pw^{i+1}$ during the password update phase. In conclusion, the oracle Corrput($U^i$) gives no advantage to the adversary.

**Experiment** $P_3$: In experiment $P_3$, the adversary A queried oracle Execute($\mathrm{C}^i, \mathrm{S}^i$) to get the session key $sk_{\mathrm{C}} = \mathrm{H}(\mathrm{ID_C} \mid \mathrm{ID_S} \mid \mathrm{X}|\mathrm{Y}| \ w_{\mathrm{S}}$—Nonce—$\rho_{\mathrm{C}}$) =H($\mathrm{ID_C} \mid \mathrm{ID_S}|\mathrm{X}|\mathrm{Y}|w_{\mathrm{S}}|\mathrm{Nonce}|\rho_{\mathrm{S}}$) $= sk_{\mathrm{S}}$, among the Nonce value is replaced by a random value Nonce'. If A can distinguish sk and sk' by $msg_{\mathrm{C}} = (ID_{\mathrm{C}}|\mathrm{X}|E_{pks}[\mathrm{H}(\mathrm{X}| \ ID_{\mathrm{C}}|\mathrm{pw}|\mathrm{Nonce}),\mathrm{Nonce}])$, it means he can break the semantic security of cryptosystem. So, we can define the advantage $|Adv_A^{P_2}(k) - Adv_A^{P_3}(k)|$ between P2 and P3 is negligible.

**Experiment** $P_4$ : In experiment $P_4$, we redefine the successful attack state of the adversary. We first define the message type which is from adversary.

1. Oracle generated: A message is called oracle generated if ti was answered by oracles and it is generated in accordance with the specification of protocol.
2. Adversary generated: A message is called adversary generated if it was randomly selected by the adversary.

The experiment $P_4$ is only redefined for the (2).

Then we divide the $Send_1(\mathrm{S}, msg_{\mathrm{C}})$, $Send_2(\mathrm{C}, msg_{\mathrm{S}})$ oracles according to the two transmissions in the protocol. If $msg_{\mathrm{C}}$ is adversary generated and it passed the authentication, then it si said to be valid and it means he has the correct password; if $msg_{\mathrm{S}}$ is adversary generated and pass the authentication, then it said to be valid. Both of these can be considered successful attacks, on the contrary, it is necessary to refer to the definition in $P_3$ to determine whether the attack is successful. We can see the advantage in $P_4$ is more than in $P_3$, this means $|Adv_A^{P_3}(k) \leq Adv_A^{P_4}(K)|$.

**Experiment** $P_5$: in experiment $P_5$, it continued the definition of a successful attack from $P_4$. The adversary A queried oracle $Send_1(\mathrm{S}, msg_{\mathrm{C}})$ to get the $msg_{\mathrm{C}} = (ID_{\mathrm{C}}|\mathrm{X}|E_{pks}[\mathrm{H}(\mathrm{X}|ID_{\mathrm{C}}|\mathrm{pw}|\mathrm{Nonce}),\mathrm{Nonce}])$, among the password pw is replaced with a random value pw'. If the cryptosystem which we use can satisfy the semantic security, then the adversary can't reveal the correct pw from $\mathrm{Auth_C} = \mathrm{E_{pks}}[\mathrm{H}(\mathrm{X}|\mathrm{ID_C}|\mathrm{pw}|\mathrm{Nonce}|\mathrm{TS}_1), \mathrm{Nonce}]$. So we can define the advantage $|Adv_A^{P_4}(k) - Adv_A^{P_5}(k)|$ between $P_4$ and $P_5$ is negligible.

**Experiment** $P_6$: In experiment $P_6$, the adversary A queried oracle Execute($\mathrm{C}^i, \mathrm{S}^i$) to get the authentication message $\mathrm{Auth_S} = \mathrm{H}(\mathrm{Y}|\mathrm{ID_S}|\mathrm{pw}|w_{\mathrm{S}}|\mathrm{Nonce}+1|)$, among the password pw is replace with a random value pw". If the crypto system which we use can satisfy the semantic security, then the adversary can't reveal the correct session key sk from $\mathrm{Auth_S}$ and $msg_{\mathrm{C}}$= $(ID_{\mathrm{C}}|\mathrm{X}|E_{pks}[\mathrm{H}(\mathrm{X}|ID_{\mathrm{C}}|\mathrm{pw}|\mathrm{Nonce}),\mathrm{Nonce}])$. So we can define the advantage $|Adv_A^{P_5}(k) - Adv_A^{P_6}(k)|$ between $P_4$ and $P_5$ is negligible.
Now we can know the relationship of adversary's advantage from $P_0$ to $P_6$ is $|Adv_A^{P_i}(k) \leq Adv_A^{P_6}(k)|$, where $i = 0, \cdots, 5$. It means if the advantage in last experiment $P_6$ is negligible, then the advantage in all previous experiment is negligible too. Finally, we will prove that $Adv_A^{P_6}(k)$ is negligible threat to the security of our protocol. From the above proof, we can know the probability of successful attack in $P_6$ is $Pr_A^6[Succ] = \frac{1}{2}$, it means even if the adversary use the oracle to query the secret parameter in our protocol, he can only achieve the same probability with the random guess. Therefore, we can define the advantage of adversary in last experiment $P_6$ is $Adv_A^{P_6} = 2pr_A^6[Succ] - 1 = 0$, this

means our protocol didn't disclose any confidential information to the adversary in the communication and it is saft.

5.3. **NTRU.** The cryptssytem used in our protocol is NTRU public key cryptosystem, which is designated as standard IEEE P1263.1 in 2008[6]. Stehl é et al.[7] proposed a modified version of NTRU in 2011, theri protocol replaced the SVP of original security basis to the LWE and modify the key generation. In original NTUR, the private key consists of two polynomial functions with degree less than n and coefficients in $\{0, 1, -1\}$, and the public key is theri quotient. Stehlé believes that the key is limited to a small range so that it can't achieve the randomness. To achieve the public/private key pair distribution statistically close to uninform, they sample the private key polynomials according to discrete Gaussian with standard deviation in LWE. Thus, it can provide the higher secure level than original NTRU.

5.4. **Comparison with Related Works.** In this subsection, we will compare our scheme with other related works in different items, these papers [3, 1, 10] are the main reference in our scheme, the paper[9] is also a lattice based PAKE, and the paper[8] is a secure PAKE based on discrete logarithm problem in number theory, so we refer to these papers as a comparison object. We have defined several comparison items including:

1. Security based on which hard problem.
2. Authentication.
3. The number of communcications.
4. Public/private key pair.

Table 1. Comparison with Related Works

| | **J-PAKE**[8] | **Xun**[3] | **Katz**[9] | **Ding**[1] | **Zhang**[10] | **Our Scheme** |
|---|---|---|---|---|---|---|
| Hardness Problem | Discrete Logarithm | Discrete Logarithm | LWE | LWE | LWE | LWE |
| Authentication | Implicit | Explicit | Explicit | None | Implicit | Explicit |
| Communication (Pass) | 4 | 2 | 3 | 2 | 2 | 2 |
| Public/ private key pair | None | Server only | Both parties | None | both parties | Server only |

6. **Conclusion.** In this paper, we proposed a PAKE scheme in the client/server model. The new scheme is a lattice based PAKE so it is more secure than scheme which were proved secure based on discrete logarithm problem or factoring problem in number theory. Our scheme is different from previous agreements that users keep their own public/private key pairs, but the only secret in our scheme is registered for authentication is user's password. This method can reduce the burden on key management for the client side. Importantly, it is secure against the adversaries who contains quantum computes and others powerful computers. And, it could be finished within just two steps. This is exactly a safe and effective password based authenticated key exchange agreement and we believe the new scheme can be beneficial to the real world.

## REFERENCES


[1] J. Ding, X. Xie, and X. Lin, A simple provably secure key exchange scheme based on the learning with errors problem, *IACR Cryptol.* ePrint Arch., vol. 2012, pp. 688, 2012.

[2] D.Micciancio and O.Regev, Worst-case to average-case reductions based on gaussian measures, *SIAM Journal on Computing*, vol. 37, no. 1, pp. 267–302, 2007.

[3] X. Yi, R. Tso, and E. Okamoto, Identity-based password-authenticated key exchange for client/server model, *SECRYPT*, vol. 12, pp. 45–54, 2012.

[4] J. Katz, R. Ostrovsky, and M. Yung, Efficient password-authenticated key exchange using human-memorable passwords, *in International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 475–494, Springer, Berlin-Heidelberg, Germany, 2001.

[5] M. Bellare, D. Pointcheval, and P. Rogaway, Authenticated key exchange secure against dictionary attacks, *in International conference on the theory and applications of cryptographic techniques*, pp. 139–155, Springer, Berlin-Heidelberg, Germany, 2000.

[6] IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices, *in IEEE Std 1363.1-2008*, pp.1–81, 10 March 2009, doi: 10.1109/IEEESTD.2009.4800404.

[7] D. Stehlé and R. Steinfeld, Making ntru as secure as worst-case problems over ideal lattices, *in Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 27–47, Springer, Berlin-Heidelberg, Germany, 2011.

[8] F. Hao and P. Ryan, J-pake: authenticated key exchange without pki, *in Transactions on computational science XI*, pp. 192–206, Springer, Berlin-Heidelberg, Germany, 2010.

[9] J. Katz and V. Vaikuntanathan, Smooth projective hashing and password-based authenticated key exchange from lattices, *in International Conference on the Theory and Application of Cryptology and Information Security*, pp. 636–652, Springer, Berlin-Heidelberg, Germany, 2009.

[10] J. Zhang, Z. Zhang, J. Ding, M. Snook, and Ö. Dagdelen, Authenticated key exchange from ideal lattices, *in Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 719–751, Springer, Berlin-Heidelberg, Germany, 2015.

[11] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, *Journal of the ACM (JACM)*, vol. 56, no. 6, pp. 1–40, 2009.

[12] J. Hoffstein, J. Pipher, and J. H. Silverman, Ntru: A ring-based public key cryptosystem, *in International Algorithmic Number Theory Symposium*, pp. 267–288, Springer, Berlin-Heidelberg, Germany, 1998.

[13] S.-W. Park and I.-Y. Lee, Anonymous authentication scheme based on ntru for the protection of payment information in nfc mobile environment, *Journal of Information Processing Systems*, vol. 9, no. 3, pp. 461–476, 2013.

[14] R. Tso and Y.-S. Jheng, Security analysis of a ntru-based mutual authentication scheme, *in 2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 1–3, IEEE, 2016.

[15] W. Diffie and M. Hellman, New directions in cryptography, *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.