Cryptanalysis of two post-quantum authenticated key agreement protocols

Mehdi Abri
University of Isfahan
Isfahan, Iran
m8abri@gmail.com

Hamid Mala
University of Isfahan
Isfahan, Iran
h.mala@eng.ui.ac.ir

Abstract

As the use of the internet and digital devices has grown rapidly, keeping digital communications secure has become very important. Authenticated Key Agreement (AKA) protocols play a vital role in securing digital communications. These protocols enable the communicating parties to mutually authenticate and securely establish a shared secret key. The emergence of quantum computers makes many existing AKA protocols vulnerable to their immense computational power. Consequently, designing new protocols that are resistant to quantum attacks has become essential. Extensive research in this area had led to the design of several post-quantum AKA schemes.

In this paper, we analyze two post-quantum AKA schemes proposed by Dharminder et al. [2022] and Pursharthi and Mishra. [2024] and demonstrate that these schemes are not secure against active adversaries. An adversary can impersonate an authorized user to the server. We then propose reliable solutions to prevent these attacks.

Keywords

Authentication, Key agreement, Lattice-based cryptography, Post-quantum security

1 Introduction

A mutual authenticated key agreement scheme between two entities, A and B, is a protocol that enables parties to authenticate each other as well as establish a shared secret key [15]. Achieving a shared key along with mutual authentication has many applications in the real world [2, 16, 24, 25, 28, 31, 33, 34], two common ones are as follows.

First, the development and expansion of IoT in recent years have increased the data produced by smart devices [23]. This increase has led to challenges such as determining efficient solutions for storing, maintaining, and accessing these data, in a way that if these data are stored and maintained on a cloud server, users can access them when needed. Access to and retrieval of data by users from the cloud server must be such that no one other than the authorized user can gain knowledge of or access the exchanged data. For this purpose, one of the existing solutions is the use of mutual authenticated key agreement schemes. In this case, after mutually authenticating each other, they establish a shared session key and securely exchange data.

Second, another use case for these schemes is in establishing secure communications on mobile devices [32]. Nowadays, people use mobile devices for many daily activities. In many cases, the proper functioning of these devices is contingent upon establishing connections with the outside world and other devices to exchange data and access provided services. As mentioned in the previous section, to ensure security in these communications such that the confidentiality of sensitive data is protected and access to this data and services is restricted to authorized individuals, the use of mutually authenticated key agreement schemes is desirable.

Many of these schemes use classical computational hard assumptions such as integer factorization [22] and discrete logarithm problem [7, 20] to establish their security.

With the advent of quantum computers and consequently quantum algorithms, many cryptographic schemes based on the hardness of classical computational assumptions will be broken and will no longer be secure [19, 30]. Post-quantum cryptography refers to cryptographic systems that remain secure even if an attacker possesses a quantum computer [3]. These cryptographic systems use various approaches to maintain security against quantum attacks. Lattice-based cryptography [21] and problems such as Learning With Errors (LWE) [29] and Ring-LWE [18], which are fundamental lattice-based problems, receiving significant attention in the design of post-quantum schemes [26]. According to the predictions that quantum computers capable of breaking classical schemes will be produced within the next 20 years, there is a need to design schemes resistant to quantum attacks [4]. Consequently, Dharminder et al. [6] have proposed a lattice-based post-quantum mutually authenticated key agreement scheme for the Internet of Things (IoT), and Pursharthi and Mishra [27] have designed a lattice-based post-quantum mutually authenticated key agreement scheme for communications in mobile devices.

In this paper, we analyze the two aforementioned schemes and we demonstrate that [6] is vulnerable to impersonation attacks and [27] is vulnerable to replay attacks and then propose a solution to address the existing flaws.

The rest of the paper is organized as follows. In Section 2, the lattice-based authenticated key agreement schemes are reviewed. In the next section, we present the necessary preliminary concepts to examine the two protocols [6, 27]. In Section 4, scheme [6] is reviewed, followed by a discussion of its vulnerabilities. Then a solution is proposed to fix its vulnerabilities. The next section follows a similar structure as Section 4 and is dedicated to examining the scheme proposed in [27]. Finally, we conclude this article in the last section.

2 Related Work

Katz and Vaikuntanathan [14] in 2009 first proposed a password-authenticated key exchange (PAKE) scheme based on the LWE assumption, allowing two parties to establish a new shared key using a low-entropy password, with security that can be proven under the LWE assumptions. Then in 2012, Ding et al. [10] proposed a scheme based on the RLWE assumptions to agree on a session key. This key exchange protocol was inspired from the Diffie-Hellman key exchange [8]. However, the scheme was vulnerable to man-in-the-middle (MitM) attacks because it lacked mutual authentication. Consequently, an authenticated version of this scheme [35], secure against active adversaries, was subsequently proposed.

In 2017, Ding et al. [9] introduced a new attack called the signal leakage attack (SLA) on [10], unveiling new vulnerabilities in RLWE-based key exchange schemes. Utilizing this idea, Liu et al. [17] were able to find out the vulnerability of scheme [1] against this type of attack. In 2020, Islam [13] proposed a lattice-based authenticated key agreement (AKA) scheme, which was proven secure in the random oracle model. Just like other schemes that used RLWE assumptions, this scheme was also insecure against SLA [5].

In 2018, Feng et al. [12] proposed an anonymous ideal lattice-based authenticated key agreement scheme for client-server environments, which was proven secure based on the hardness of the RLWE problem in the random oracle model. In this environment, authorized users can authenticate and utilize services provided by the server without worrying about their identity privacy being compromised. Ding et al. [11], upon reviewing [12], realized that this scheme was insecure against the SLA and consequently proposed a secure anonymous AKA scheme against this attack. Subsequently, Pursharthi and Mishra [27] reviewed [11] and realized the vulnerabilities of this scheme against attacks such as device-stolen attacks, password-guessing attacks, and insider attacks. They proposed a secure scheme against these attacks with improved computational complexity compared to the previous ones. In this paper, we will demonstrate that the scheme [27] is vulnerable to replay attacks. Additionally, by reviewing the scheme proposed by Dharminder et al. [6], which is a lattice-based AKA scheme for the Internet of Things, we will show that this scheme is insecure against active adversaries because an adversary can impersonate an authorized user to the server.

3 Preliminaries

In this section, we will cover the basic concepts required for the schemes[6, 27]. Suppose q is an odd prime number, $\mathbb{Z}$ is the set of all integers, $\mathbb{R}^+$ is the set of all non-negative real numbers, $\mathbb{Z}^+$ is the set of all non-negative integers, and $\mathbb{R}$ is the set of all real numbers. Let χ_β represent a discrete Gaussian distribution where β is a scaling parameter affecting the variance of the distribution, defined over $\mathbb{R}^+$ and $n = 2^t$ where $t \in \mathbb{Z}^+$. Also, let $\mathbb{Z}[X]$ and $\mathbb{Z}_q[X]$ represent the rings of polynomials over $\mathbb{Z}$ and $\mathbb{Z}_q$, respectively. Consider two polynomial rings $R = \mathbb{Z}[X]/(x^n + 1)$ and $R_q = \mathbb{Z}_q[X]/(x^n+1)$. Suppose $M = \{-\lfloor q/4 \rfloor, \ldots, \lfloor q/4 \rfloor\}$ is a subset of $\mathbb{Z}_q = \{-(q-1)/2, \ldots, (q-1)/2\}$.

Definition 1. *For any $x \in \mathbb{Z}_q$, the characteristic function Cha is defined as follows:*

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

Definition 2. *The auxiliary mod function $Mod_2 : \mathbb{Z}_q \times \{0,1\} \to \{0,1\}$ is defined as follows:*

$$Mod_2(r, s) = \big(r + s \cdot (q-1)/2 \quad (mod\ q)\big) \quad (mod\ 2) \tag{2}$$

where $r \in \mathbb{Z}_q$ and $s = Cha(r)$.

This function is denoted by Ψ_2 in [27].

Lemma 1. *If q is an odd prime number, $c, e \in R_q$ such that $| e |< q/8$ and $w = c + 2 \cdot e$, then $Mod_2(c, Cha(c)) = Mod_2(w, Cha(c))$*

Now, we want to extend two functions Cha and Mod_2 on R_q: each element $c = c_0 + c_1 x^1 + \ldots + c_{n-1} x^{n-1} \in R_q$ can be described as $c = (c_0, c_1, \ldots, c_{n-1})$. For any random vector $u = (u_0, u_1, \ldots, u_{n-1}) \in \{0,1\}^n$, the definitions of the above functions are extended as follows:

$$Cha(c) = \big(Cha(c_0), Cha(c_1), \ldots, Cha(c_{n-1})\big) \tag{3}$$

$$Mod_2(c, u) = \big(Mod_2(c_0, u_0), Mod_2(c_1, u_1), \ldots, Mod_2(c_{n-1}, u_{n-1})\big) \tag{4}$$

Definition 3 (*Ring Learning with Errors*). *Suppose A_{s,χ_β} is a distribution over $(c, c \cdot s + e) \in R_q \times R_q$ where $c, s \longleftarrow R_q$ and $e \longleftarrow \chi_\beta$ are chosen randomly. The distinction of A_{s,χ_β} from a uniform distribution over $R_q \times R_q$ for any PPT adversary is computationally infeasible.*

4 Cryptanalysis of Dharminder's protocol

In this section, we first examine protocol [6]. Then, we analyze its security and point out the existing vulnerability. Finally, we conclude this section with a proposed secure approach to address the existing security flaw. The symbols used in this protocol are described in Table 1.

Table 1: Symbols and Their Descriptions

Notations	Descriptions
U_i	i^{th} user
MD_i	U_i's smart (mobile) device
S	cloud server
Cha	Characteristic mapping
Mod_2	Auxiliary modular mapping
ID_{U_i}	Identity of U_i
SK	Session key shared between U_i and S
PW_i	U_i's chosen (sufficiently strong) password
x	Master secret key of server S
α_{U_i}, rn_{U_i}	Random secrets generated by U_i
T_1, T_2	Current timestamps
ΔT	Maximum allowable transmission delay
q	A sufficiently large prime
R_q	Finite ring
$x \longleftarrow A$	x is randomly chosen from the set A
χ_β	Discrete Guassian distribution
$h(\cdot)$	Collision-resistance hash function
c	c is randomly chosen from R_q
n	Positive integer of the form $2^k, k > 0$
P	Public key of server
$\oplus$	Bitwise excelusive-OR (XOR)
$\|\|$	String concatenation

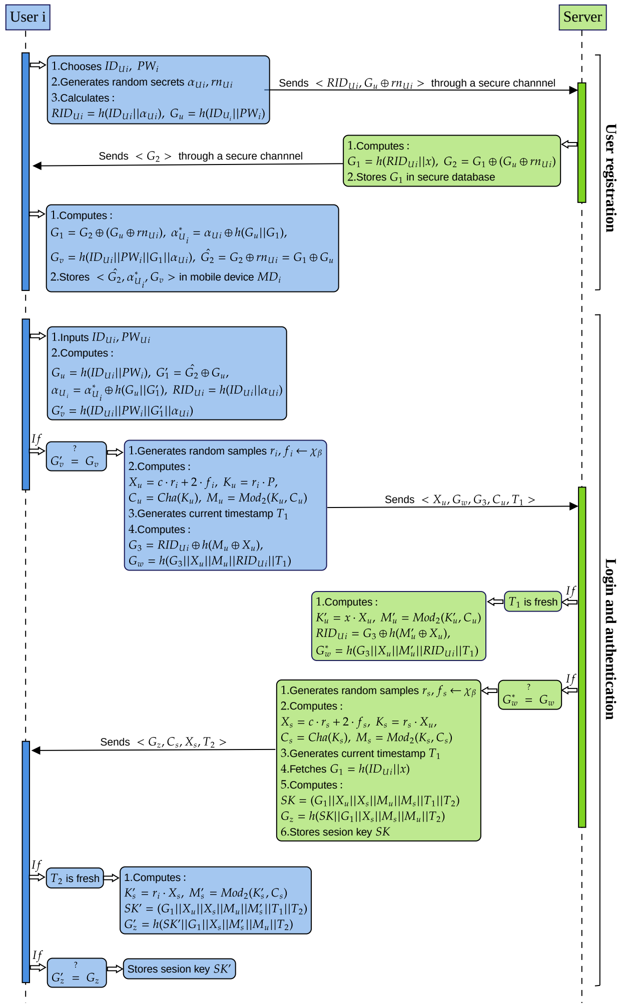
User i
Server
1.Chooses $ID_{Ui},\ PW_i$
2.Generates random secrets α_{Ui}, rn_{Ui}
3.Calculates :
$RID_{Ui} = h(ID_{Ui}||\alpha_{Ui}),\ G_u = h(ID_{U_i}||PW_i)$
Sends $< RID_{Ui}, G_u \oplus rn_{Ui} >$ through a secure channnel
1.Computes :
$G_1 = h(RID_{Ui}||x),\ G_2 = G_1 \oplus (G_u \oplus rn_{Ui})$
2.Stores G_1 in secure database
Sends $< G_2 >$ through a secure channnel
1.Computes :
$G_1 = G_2 \oplus (G_u \oplus rn_{Ui}),\ \alpha^*_{U_i} = \alpha_{Ui} \oplus h(G_u||G_1),$
$G_v = h(ID_{Ui}||PW_i||G_1||\alpha_{Ui}),\ \hat{G_2} = G_2 \oplus rn_{Ui} = G_1 \oplus G_u$
2.Stores $< \hat{G_2}, \alpha^*_{U_i}, G_v >$ in mobile device MD_i
User registration
1.Inputs ID_{Ui}, PW_{Ui}
2.Computes :
$G_u = h(ID_{Ui}||PW_i),\ G'_1 = \hat{G_2} \oplus G_u,$
$\alpha_{U_i} = \alpha^*_{U_i} \oplus h(G_u||G'_1),\ RID_{Ui} = h(ID_{Ui}||\alpha_{Ui})$
$G'_v = h(ID_{Ui}||PW_i||G'_1||\alpha_{Ui})$
If
$G'_v \overset{?}{=} G_v$
1.Generates random samples $r_i, f_i \leftarrow \chi_\beta$
2.Computes :
$X_u = c \cdot r_i + 2 \cdot f_i,\ K_u = r_i \cdot P,$
$C_u = Cha(K_u),\ M_u = Mod_2(K_u, C_u)$
3.Generates current timestamp T_1
4.Computes :
$G_3 = RID_{Ui} \oplus h(M_u \oplus X_u),$
$G_w = h(G_3||X_u||M_u||RID_{Ui}||T_1)$
Sends $< X_u, G_w, G_3, C_u, T_1 >$
If
T_1 is fresh
1.Computes :
$K'_u = x \cdot X_u,\ M'_u = Mod_2(K'_u, C_u)$
$RID_{Ui} = G_3 \oplus h(M'_u \oplus X_u),$
$G^*_w = h(G_3||X_u||M'_u||RID_{Ui}||T_1)$
If
$G^*_w \overset{?}{=} G_w$
1.Generates random samples $r_s, f_s \leftarrow \chi_\beta$
2.Computes :
$X_s = c \cdot r_s + 2 \cdot f_s,\ K_s = r_s \cdot X_u,$
$C_s = Cha(K_s),\ M_s = Mod_2(K_s, C_s)$
3.Generates current timestamp T_1
4.Fetches $G_1 = h(ID_{Ui}||x)$
5.Computes :
$SK = (G_1||X_u||X_s||M_u||M_s||T_1||T_2)$
$G_z = h(SK||G_1||X_s||M_s||M_u||T_2)$
6.Stores sesion key SK
Sends $< G_z, C_s, X_s, T_2 >$
If
T_2 is fresh
1.Computes :
$K'_s = r_i \cdot X_s,\ M'_s = Mod_2(K'_s, C_s)$
$SK' = (G_1||X_u||X_s||M_u||M'_s||T_1||T_2)$
$G'_z = h(SK'||G_1||X_s||M'_s||M_u||T_2)$
If
$G'_z \overset{?}{=} G_z$
Stores sesion key SK'
Login and authentication

Figure 1: Sequence Diagram of Dharminder's Protocol

4.1 Dharminder's protocol

Protocol [6] is a post-quantum mutual authenticated key agreement protocol in which two entities, a user and a server, intend to establish a session key after mutual authentication. This protocol is divided into four main phases:

- **Setup Phase:** In this phase, the server, after executing the setup algorithm, makes the parameters $\{n, q, c, \chi_\beta, P, h(.)\}$ publicly available to the users and keeps x private.
- **Registration Phase:** As shown in Figure 1, the user U_i initially sends a registration request to the respective server after determining ID_{U_i}, PW_i and calculating RID_{U_i}, G_u by sending $< RID_{U_i}, G_u \oplus rn_{U_i} >$ through a secure channel. The server then uses the received values and x to calculate G_1, G_2, and after storing G_1, sends G_1 back to the user through a secure channel. Now, the user calculates the values $< \hat{G}_2, \alpha^*_{U_i}, G_v >$ in such a way that by storing these values on his/her mobile device, he/she can later use them along with input, after user authentication by the mobile device, to calculate G_1.
- **Login and Mutual Authentication Phase:** After a successful registration in the previous phase, as shown in Figure 1, the user inputs ID_{U_i}, PW_{U_i} to create a new session key. Subsequently, the mobile device calculates G'_v using these values along with those stored during the registration phase and compares it with G_v. If $G'_v = G_v$ holds, the user is authenticated by the mobile device, and the login phase is successfully completed. In the next phase, MD_i by calculating and sending $< X_u, G_w, G_3, C_u, T_1 >$ through a public channel to the server, requests authentication and the creation of a new session key. Upon receiving the user's message at time T_1^*, the server verifies the validity of the received timestamp by checking the condition $|T_1 - T_1^*| < \Delta T$. If T_1 is valid, the server uses the received values and x to calculate G_W^* and compares it with G_W. If $G_W^* = G_W$ holds, the user is authenticated by the server. Then, the server calculates X_s, K_s, C_s, M_s, G_z, and the shared session key SK. The server sends $< G_z, C_s, X_s, T_2 >$ to the user, enabling the user to authenticate the server and compute the shared session key. Finally, upon receiving the server's message, the user first verifies the validity of the received timestamp by checking the condition $|T_2 - T_2^*| < \Delta T$. If T_2 is valid, the user uses the received values and G_1 to calculate SK' and G'_z. Then, the value G'_z is compared with G_z. If $G'_z = G_z$ holds, the server is authenticated by the user, and the key SK' is stored as the new session key.
- **User Password Change Phase:** In this phase, if the registered user wishes, their password can be updated to a new one locally without the need to establish a connection with the server.

4.2 Attack on Dharminder's protocol

We want to show that in Protocol [6], an adversary A can authenticate themselves as user U_i to the server S. The adversary A can bypass the login phase on the user's mobile device and send a message as U_i in the public channel. Now, to authenticate U_i to the server using the received message, it is sufficient for the adversary to calculate and send valid $< \hat{X}_u, \hat{G}_w, \hat{G}_3, \hat{C}_u >$ to the server. For this purpose, the adversary A first randomly selects r', f' from χ_β and calculates the following values:

$$\hat{X}_u = c \cdot r' + 2 \cdot f' \tag{5}$$

$$\hat{K}_u = r' \cdot P \tag{6}$$

$$\hat{C}_u = Cha(\hat{K}_u) \tag{7}$$

$$\hat{M}_u = Mod_2(\hat{K}_u, \hat{C}_u) \tag{8}$$

Then, in addition to generating the timestamp T_1 at the same time, the adversary produces a random value $R\hat{I}D_{U_i}$ with a length equal to the hash function's range $h(\cdot)$. The adversary calculates the values $\hat{G}_3, \hat{G}_w$ as follows:

$$\hat{G}_3 = R\hat{I}D_{U_i} \oplus h(\hat{M}_u \oplus \hat{X}_u) \tag{9}$$

$$\hat{G}_w = h(\hat{G}_3||\hat{X}_u||\hat{M}_u||R\hat{I}D_{U_i}||T_1) \tag{10}$$

And finally, the adversary sends $< \hat{X}_u, \hat{G}_w, \hat{G}_3, \hat{C}_u, T_1 >$ to the server S on behalf of user U_i. Upon receiving the message, if the timestamp T_1 is verified to be fresh, the server uses the received data and x to calculate the following values:

$$K_{u'} = x \cdot \hat{X}_u \tag{11}$$

$$M_{u'} = Mod_2(K_{u'}, \hat{C}_u) = \hat{M}_u \tag{12}$$

$$RID_{U_i} = \hat{G}_3 \oplus h(M_{u'} \oplus \hat{X}_u) = \widehat{RID}_{U_i} \oplus h(\hat{M}_u \oplus \hat{X}_u) \oplus h(M_{u'} \oplus \hat{X}_u) = \widehat{RID}_{U_i} \tag{13}$$

$$G^*_w = h(\hat{G}_3 || \hat{X}_u || M_{u'} || RID_{U_i} || T_1) \tag{14}$$

Since $M_{u'} = \hat{M}_u, RID_{U_i} = \widehat{RID}_{U_i}$, then $G^*_w = \hat{G}_w$, and the sender's identity is authenticated as user U_i by the server. As a result, the adversary A successfully authenticated herself as user U_i to the server S. Note that although the adversary A is not able to compute the session key, she managed to impersonate U_i to the server by executing this attack.

4.3 Suggestions

To prevent the attack described in the previous section, it is sufficient to change the structure of G_w, where $G_w = h(G_3 || X_u || M_u || RID_{U_i} || T_1)$, as follows:

$$G_w = h(G_1 || G_3 || X_u || M_u || RID_{U_i} || T_1) \tag{15}$$

where G_1 is $G'_2 \oplus G_u$, which the user's mobile device calculates during the login phase. Additionally, the server securely stores this value during the user U_i's registration phase. Now, if the adversary A wants to authenticate herself as user U_i to the server, she must be able to obtain G_1, which is the long-term private key shared between the server and the user, in addition to the values described in the previous section.

5 Cryptanalysis of Pursharthi's protocol

In this section, similar to the previous section, we first examine the main protocol, then describe the existing security flaw, and finally conclude with a proposed solution. Additionally, the symbols used in this protocol are described in Table 2.

Table 2: Symbols and Their Descriptions

Notations	Descriptions
U_i	i^{th} user
S	Server
Bio_i	Boimetric of U_i
T_i	Current timestamp
$H_b(\cdot)$	Biohashing function
$H_0(\cdot)$	Hash function
$H_1(\cdot)$	Maps $\{0,1\}^* \longrightarrow \chi_a$
χa	Guassian distribution
a	$a \in R_q$ public parameter
n	Positive integer of the form $2^k, k > 0$
ρ	Prime number
$s_j \in \chi_a$	Server's private key
$p_j = a \cdot s_j + 2 \cdot e_j$	Server's public key where $e_j \in \chi_a$

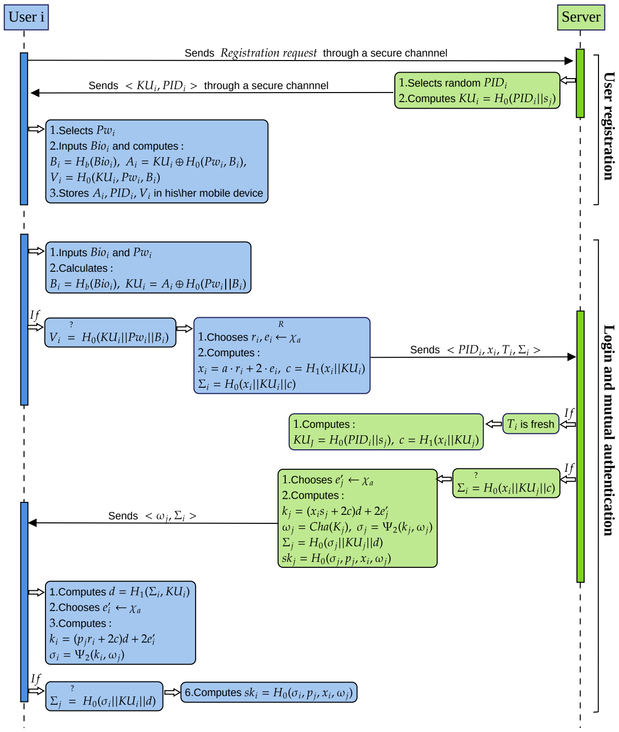
User i
Server
Sends *Registration request* through a secure channnel
1.Selects random PID_i
2.Computes $KU_i = H_0(PID_i||s_j)$
Sends $< KU_i, PID_i >$ through a secure channnel
User registration
1.Selects Pw_i
2.Inputs Bio_i and computes :
$B_i = H_b(Bio_i),\ A_i = KU_i \oplus H_0(Pw_i, B_i),$
$V_i = H_0(KU_i, Pw_i, B_i)$
3.Stores A_i, PID_i, V_i in his\her mobile device
1.Inputs Bio_i and Pw_i
2.Calculates :
$B_i = H_b(Bio_i),\ KU_i = A_i \oplus H_0(Pw_i||B_i)$
If
$V_i \overset{?}{=} H_0(KU_i||Pw_i||B_i)$
1.Chooses $r_i, e_i \overset{R}{\longleftarrow} \chi_a$
2.Computes :
$x_i = a \cdot r_i + 2 \cdot e_i,\ c = H_1(x_i||KU_i)$
$\Sigma_i = H_0(x_i||KU_i||c)$
Sends $< PID_i, x_i, T_i, \Sigma_i >$
Login and mutual authentication
If
T_i is fresh
1.Computes :
$KU_J = H_0(PID_i||s_j),\ c = H_1(x_i||KU_j)$
If
$\Sigma_i \overset{?}{=} H_0(x_i||KU_j||c)$
1.Chooses $e'_j \longleftarrow \chi_a$
2.Computes :
$k_j = (x_i s_j + 2c)d + 2e'_j$
$\omega_j = Cha(K_j),\ \sigma_j = \Psi_2(k_j, \omega_j)$
$\Sigma_j = H_0(\sigma_j||KU_j||d)$
$sk_j = H_0(\sigma_j, p_j, x_i, \omega_j)$
Sends $< \omega_j, \Sigma_i >$
1.Computes $d = H_1(\Sigma_i, KU_i)$
2.Chooses $e'_i \longleftarrow \chi_a$
3.Computes :
$k_i = (p_j r_i + 2c)d + 2e'_i$
$\sigma_i = \Psi_2(k_i, \omega_j)$
If
$\Sigma_j \overset{?}{=} H_0(\sigma_i||KU_i||d)$
6.Computes $sk_i = H_0(\sigma_i, p_j, x_i, \omega_j)$

Figure 2: Sequence Diagram of Pursharthi's Protocol

5.1 Pursharthi's protocol

Protocol [27] is also a post-quantum mutual authenticated key agreement protocol in which two entities, the user and the server, intend to establish a session key after mutual authentication. This protocol is divided into three main phases:

- **Setup Phase:** In this phase, after executing the setup algorithm, the server sends the parameters $\{n, \rho, \chi_a, a, p_j, H_0, H_1\}$ publicly to the users and keeps s_j private.
- **Registration Phase:** As shown in Figure 2, user U_i initially sends his/her registration request to the server through a secure channel to access the server's services. Upon receiving the user's registration request, the server randomly selects a PID_i and, after calculating the value KU_i, sends the values $< KU_i, PID_i >$ to the user through a secure channel. Now, the user calculates the values $< A_i, V_i >$ using his/her chosen password Pw_i and Bio_i in such a way that by storing these values and PID_i on his/her mobile device, he/she can later use them and the input information to calculate KU_i after authenticating the user through the mobile device.
- **Login and Mutual Authentication Phase:** After successful registration in the previous step, as shown in Figure 2, the user initially enters Bio_i, Pw_i to establish a new session key. The mobile device then uses these values and the values stored during the registration phase to calculate $H_0(KU_i||Pw_i||B_i)$ and compares it with V_i. If $V_i = H_0(KU_i||Pw_i||B_i)$ holds, the user is authenticated by the mobile device, and the login phase is completed successfully.

In the next step, the user's mobile device calculates x_i and Σ_i and sends $< PID_i, x_i, T_i, \Sigma_i >$ through a public channel to the server, requesting authentication and consequently the establishment of a new session key. Upon receiving the user's message, the server checks the freshness of the received timestamp T_i. If T_i is valid, the server uses the received values and s_j to calculate $H_0(x_i||KU_j||c)$ and compares it with Σ_i. If $\Sigma_i = H_0(x_i||KU_j||c)$ holds, the user is authenticated by the server. The server then calculates $k_j, \omega_j, \sigma_j, \Sigma_j$ and the shared session key sk_j and sends the values $< \omega_j, \Sigma_j >$ to the user so that the user can authenticate the server and calculate the shared key. Finally, upon receiving the server's message, the user calculates k_i, σ_i using the received values and KU_i, and then compares $H_0(\sigma_i, KU_i, d)$ with Σ_j. If $\Sigma_j = H_0(\sigma_i, KU_i, d)$ holds, the server is authenticated by the user, and the session key Sk_i is established as the new session key.

5.2 Attack on Pursharthi's protocol

We want to demonstrate that Protocol [27] is vulnerable to a replay attack. An adversary A can bypass the login phase on the user's mobile device and send a message in the public channel as user U_i. To authenticate U_i to the server using the received message, the adversary only needs to send a valid $< PID_i, x_i, T_i, \Sigma_i >$ to the server. Suppose the adversary has already eavesdropped on the public channel and obtained at least one valid $< PID_i, x_i, T_i, \Sigma_i >$. In this case, the adversary can send a message $< PID_i, x_i, T_i^{Current}, \Sigma_i >$ where $T_i^{Current}$ is the timestamp at the moment the adversary wants to send the message to the server. Upon receiving the message, the server first verifies $T_i^{Current}$ and, after confirming its freshness, uses s_j and the received message to calculate c and KU_j, and then uses them to calculate $H_0(x_i||KU_j||c)$ and compare it with Σ_i. Since it is assumed that the adversary has previously eavesdropped on a valid $< PID_i, x_i, \Sigma_i >$ and because T_i is not used in the structure of any of them, it results in $\Sigma_i = H_0(x_i||KU_j||c)$, and the adversary A is authenticated as user U_i to the server.

5.3 Suggestions

5.3.1 Prevent Replay Attack. To prevent the described replay attack, it is sufficient to change the structure of Σ_i, which is currently in the form $H_0(x_i||KU_j||c)$, to the following form:

$$\Sigma_i = H_0(x_i||KU_j||c||T_i) \tag{16}$$

In this case, even if the adversary eavesdrops on the sent messages and replaces $T_i^{Current}$ with T_i by sending a fresh message, considering the new structure of Σ_i, her identity will ultimately not be authenticated by the server.

5.3.2 Password Change Phase. One of the main limitations of Protocol [27] is the lack of a mechanism for updating the user's password to a new one. In this protocol, once the user sets a password, he/she is no longer able to change it. By using the following algorithm, if the registered user wishes, his/her password can be updated to a new one locally without the need to communicate with the server.

```
function PasswordChange($Bio_i, Pw_i, Pw_i^{new}$)
    $B_i = H_b(Bio_i)$
    $KU_i = A_i \oplus H_0(B_i||Pw_i)$
    if $V_i \overset{?}{=} H_0(KU_i||Pw_i||B_i)$ then
        $A_i^{new} = A_i \oplus H_0(B_i||Pw_i) \oplus H_0(B_i||Pw_i^{new})$
        $V_i^{new} = H_0(KU_i, Pw_i^{new}, B_i)$
        Update $(A_i, PID_i, V_i)$ by computed $(A_i^{new}, PID_i, V_i^{new})$
    end if
end function
```

Figure 3: User password change phase (execute by user's mobile device)

In the pseudocode described in Figure 3, in the first step, the user's authenticity is verified by computing $H_0(KU_i||Pw_i||B_i)$ using the input Bio_i, Pw_i and A_i stored on the user's mobile device. Then, if the user's authenticity is confirmed, the new values A_i^{new}, V_i^{new} are calculated using the new password Pw_i^{new} and the existing values on the user's mobile device. These new values replace the previous ones so that, in addition to the expiration of the previous password, the user is able to log into the application with the updated password.

6 Conclusion

In this paper, we conducted security analysis of two quantum-safe authenticated key exchange schemes [6, 27]. We demonstrated that an active adversary, after bypassing the login phase in [6], by generating the necessary values using the described method can impersonate the user to the server. In protocol [27], by eavesdropping one successful session, the adversary can manage a replay attack to impersonate an authorized user to the server. Thus, both schemes fail to provide user authentication. Then, by proposing effective countermeasures, we fortified both schemes [6, 27] against the described attacks. Moreover, by providing a secure mechanism for updating the user's password locally, we eliminated the limitation of [27] in this regard.

References

[1] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. 2016. Post-quantum key {Exchange—A} new hope. In *25th USENIX Security Symposium (USENIX Security 16)*. 327–343.

[2] Akhtar Badshah, Ghulam Abbas, Muhammad Waqas, Fazal Muhammad, Ziaul Haq Abbas, Muhammad Bilal, and Houbing Song. 2024. Blockchain-assisted lightweight authenticated key agreement security framework for smart vehicles-enabled Intelligent Transportation System. *IEEE Transactions on Automation Science and Engineering* (2024).

[3] Daniel J Bernstein and Tanja Lange. 2017. Post-quantum cryptography. *Nature* 549, 7671 (2017), 188–194.

[4] Lily Chen, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray A Perlner, and Daniel Smith-Tone. 2016. *Report on post-quantum cryptography*. Vol. 12. US Department of Commerce, National Institute of Standards and Technology ....

[5] Vivek Dabra, Anju Bala, and Saru Kumari. 2021. Flaw and amendment of a two-party authenticated key agreement protocol for post-quantum environments. *Journal of Information Security and Applications* 61 (2021), 102889.

[6] Dharminder Dharminder, Challa Bhageeratha Reddy, Ashok Kumar Das, Youngho Park, and Sajjad Shaukat Jamal. 2022. Post-quantum lattice-based secure reconciliation enabled key agreement protocol for IoT. *IEEE Internet of Things Journal* 10, 3 (2022), 2680–2692.

[7] Claus Diem. 2011. On the discrete logarithm problem in elliptic curves. *Compositio Mathematica* 147, 1 (2011), 75–104.

[8] Whitfield Diffie and Martin E Hellman. 2022. New directions in cryptography. In *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*. 365–390.

[9] Jintai Ding, Saed Alsaygih, RV Saraswathy, Scott Fluhrer, and Xiaodong Lin. 2017. Leakage of signal function with reused keys in RLWE key exchange. In *2017 IEEE international conference on communications (ICC)*. IEEE, 1–6.

[10] Jintai Ding, Xiang Xie, and Xiaodong Lin. 2012. A simple provably secure key exchange scheme based on the learning with errors problem. *Cryptology ePrint Archive* (2012).

[11] Ruoyu Ding, Chi Cheng, and Yue Qin. 2022. Further analysis and improvements of a lattice-based anonymous PAKE scheme. *IEEE Systems Journal* 16, 3 (2022), 5035–5043.

[12] Qi Feng, Debiao He, Sherali Zeadally, Neeraj Kumar, and Kaitai Liang. 2018. Ideal lattice-based anonymous authentication protocol for mobile devices. *IEEE Systems Journal* 13, 3 (2018), 2775–2785.

[13] SK Hafizul Islam. 2020. Provably secure two-party authenticated key agreement protocol for post-quantum environments. *Journal of Information Security and Applications* 52 (2020), 102468.

[14] Jonathan Katz and Vinod Vaikuntanathan. 2009. 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, 636–652.

[15] Laurie Law, Alfred Menezes, Minghua Qu, Jerry Solinas, and Scott Vanstone. 2003. An efficient protocol for authenticated key agreement. *Designs, Codes and Cryptography* 28 (2003), 119–134.

[16] Hui-Tang Lin and Wei-Li Jhuang. 2024. Blockchain-Based Lightweight Certificateless Authenticated Key Agreement Protocol for V2V Communications in IoV. *IEEE Internet of Things Journal* (2024).

[17] Chao Liu, Zhongxiang Zheng, and Guangnan Zou. 2019. Key reuse attack on newhope key exchange protocol. In *Information Security and Cryptology–ICISC 2018: 21st International Conference, Seoul, South Korea, November 28–30, 2018, Revised Selected Papers 21*. Springer, 163–176.

[18] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. 2010. On ideal lattices and learning with errors over rings. In *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29*. Springer, 1–23.

[19] Vasileios Mavroeidis, Kamer Vishi, Mateusz D Zych, and Audun Jøsang. 2018. The impact of quantum computing on present cryptography. *arXiv preprint arXiv:1804.00200* (2018).

[20] Kevin S McCurley. 1990. The discrete logarithm problem. In *Proc. of Symp. in Applied Math*, Vol. 42. USA, 49–74.

[21] Daniele Micciancio and Oded Regev. 2009. Lattice-based cryptography. In *Post-quantum cryptography*. Springer, 147–191.

[22] Peter L Montgomery. 1994. A survey of modern integer factorization algorithms. *CWI quarterly* 7, 4 (1994), 337–366.

[23] Dimitris Mourtzis, Ekaterini Vlachou, and NJPC Milas. 2016. Industrial big data as a result of IoT adoption in manufacturing. *Procedia cirp* 55 (2016), 290–295.

[24] Vanga Odelu, Ashok Kumar Das, Saru Kumari, Xinyi Huang, and Mohammad Wazid. 2017. Provably secure authenticated key agreement scheme for distributed mobile cloud computing services. *Future Generation Computer Systems* 68 (2017), 74–88.

[25] Vanga Odelu, Ashok Kumar Das, Mohammad Wazid, and Mauro Conti. 2016. Provably secure authenticated key agreement scheme for smart grid. *IEEE Transactions on Smart Grid* 9, 3 (2016), 1900–1910.

[26] Stefano Pirandola, Ulrik L Andersen, Leonardo Banchi, Mario Berta, Darius Bunandar, Roger Colbeck, Dirk Englund, Tobias Gehring, Cosmo Lupo, Carlo Ottaviani, et al. 2020. Advances in quantum cryptography. *Advances in optics and photonics* 12, 4 (2020), 1012–1236.

[27] Komal Pursharthi and Dheerendra Mishra. 2024. Towards post-quantum authenticated key agreement scheme for mobile devices. *Journal of Information Security and Applications* 82 (2024), 103754.

[28] Alavalapati Goutham Reddy, Ashok Kumar Das, Vanga Odelu, Awais Ahmad, and Ji Sun Shin. 2019. A privacy preserving three-factor authenticated key agreement protocol for client–server environment. *Journal of ambient intelligence and humanized computing* 10 (2019), 661–680.

[29] Oded Regev. 2009. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)* 56, 6 (2009), 1–40.

[30] Peter W Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 124–134.

[31] Anil Kumar Sutrala, Mohammad S Obaidat, Sourav Saha, Ashok Kumar Das, Mamoun Alazab, and Youngho Park. 2021. Authenticated key agreement scheme with user anonymity and untraceability for 5G-enabled softwarized industrial cyber-physical systems. *IEEE Transactions on Intelligent Transportation Systems* 23, 3 (2021), 2316–2330.

[32] Loïc D Tsobdjou, Samuel Pierre, and Alejandro Quintero. 2021. A new mutual authentication and key agreement protocol for mobile client—server environment. *IEEE Transactions on Network and Service Management* 18, 2 (2021), 1275–1286.

[33] Yapeng Wu, Hua Guo, Yiran Han, Sijia Li, and Jianwei Liu. 2024. A Security-Enhanced Authentication and Key Agreement Protocol in Smart Grid. *IEEE Transactions on Industrial Informatics* (2024).

[34] Zheng Yang, Junyu Lai, Yingbing Sun, and Jianying Zhou. 2019. A novel authenticated key agreement protocol with dynamic credential for WSNs. *ACM Transactions on Sensor Networks (TOSN)* 15, 2 (2019), 1–27.

[35] Jiang Zhang, Zhenfeng Zhang, Jintai Ding, Michael Snook, and Özgür Dagdelen. 2015. Authenticated key exchange from ideal lattices. In *Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II 34*. Springer, 719–751.