Journal of Information Hiding and Multimedia Signal Processing ©2016 ISSN 2073-4212
Ubiquitous International Volume **7**, Number **4**, July **2016**

A Novel Three-party Authenticated Key Exchange Protocol Based on Secret Sharing

Yanjun Liu and Chin-Chen Chang

Department of Information Engineering and Computer Science
Feng Chia University, Taichung, 407, Taiwan
yjliu104@gmail.com; alan3c@gmail.com

Chin-Yu Sun

Department of Computer Science
National Tsing-Hua University, Hsinchu, 30013, Taiwan
sun.chin.yu@gmail.com

Received December, 2015; revised April, 2016

Abstract. *The most common method to achieve security in multi-party communication is to provide a session key to encrypt messages transmitted among the parties. A three-party authenticated key exchange (3PAKE) protocol can allow two clients to authenticate each other's validity and establish a common session key with the help of a trusted server. In recent years, many 3PAKE protocols have been proposed in the literature, but most of them are either insecure or inefficient. Lv et al. (2013) proposed a new 3PAKE protocol and claimed that it achieved higher efficiency than conventional 3PAKE protocols. However, some researchers pointed out that Lv et al.'s protocol was susceptible to man-in-the-middle attacks and off-line dictionary attacks. In addition, the efficiency of Lv et al.'s protocol still can be increased, since the protocol requires a heavy burden of mathematical operations, such as modulus exponential and public key encryption/decryption operations. In this paper, we proposed a novel 3PAKE protocol based on Shamir's secret sharing scheme. Our proposed protocol can satisfy the essential security requirements and withstand various well-known attacks. Performance analysis showed that our proposed protocol is more efficient than Lv et al.'s protocol.*
Keywords: Three-party; Key agreement; Secret sharing; Security; Efficiency

1. Introduction.

At the present time, one of the most considerable challenges in the field of cryptography is to provide secure communication among multiple clients in an insecure network. Establishing an authenticated session key to encrypt messages transmitted among clients for subsequent communication is a mechanism that is used extensively to achieve security and privacy. Establishing such a key can be accomplished in two ways, i.e., key exchange [1, 2, 3, 4, 5, 6, 7] and key distribution [8, 9, 10, 11]; in our research, we focused only on the issue of key exchange.

In 1992, Bellovin and Merritt [12] introduced the first two-party authenticated key exchange (2PAKE) protocol, and their work led to prolific research on 2PAKE protocols [13, 14, 15]. However, 2PAKE protocols have the problem of being inefficient because one party must share a password with each party with whom he/she wants to communicate. If the number of involved clients increases drastically, too many passwords are needed to accomplish key agreement. To solve this problem, three-party authenticated key exchange (3PAKE) protocols are proposed. In a 3PAKE protocol, each client only needs to share a password with a trusted server who helps establish a session key for the clients. Although the server participates in the establishment of session keys, session keys are not allowed to be disclosed to the server.

In recent years, there have been many literatures that aimed to enhance the security and efficiency of 3PAKE protocols [16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28]. Joux [16] proposed a one-round 3PAKE protocol based on the Diffie-Hellman key exchange scheme [29] and Weil pairing. Unfortunately, Joux's protocol is vulnerable to man-in-the-middle attacks, as is the case for the Diffie-Hellman key exchange scheme. In 2007, Lu and Cao [17] proposed an efficient 3PAKE protocol that did not require public key cryptosystems. However, a series of research works [18, 19, 20] pointed out that Lu and Cao's protocol is susceptible to some kinds of impersonation, man-in-the-middle, and on-line dictionary attacks, and some improvements were made. However, the efficiencies of the improved protocols in [18] and [20] were lower than that of Lu and Cao's protocol. In 2009, inspired by Yeh et al.'s protocol [21], Lee et al. [22] developed a plaintext-equivalent protocol and a verifier-based protocol. Their analyses indicated that their protocols had the same computational cost as Yeh et al.'s protocols, while the communication cost were lower. Yoon and Yoo [23] proposed an enhanced 3PAKE protocol based on Chang and Chang's protocol [24]. They claimed that their protocol overcame the security weakness of the protocol proposed by Chang and Chang with the same communication cost. Nevertheless, Lo and Yeh [25] found that Yoon and Yoo's protocol is insecure against undetectable on-line dictionary attacks. Huang [26] proposed a simple and efficient 3PAKE protocol, but, later, Yoon and Yoo [27] and Liang et al. [2] pointed out that off-line dictionary attacks and undetectable on-line dictionary attacks can be launched successfully in Huang's protocol. Recently, Lv et al. [7] proposed a new 3PAKE protocol with less computational complexity. However, according to Yoon's analysis [28], the protocol cannot resist man-in-the-middle attacks and off-line dictionary attacks.

In this paper, we propose an efficient 3PAKE protocol while keeping security at a high level. The contributions of our proposed protocol are listed below:

(1) To the best of our knowledge, no 3PAKE protocols have used Shamir's secret sharing scheme as their basic building block. Since Shamir's secret sharing scheme is unconditionally secure and the time of constructing it is relatively less than other heavy-burden operations, we innovatively applied it in the design of our 3PAKE protocol. In addition, Shamir's secret sharing scheme can be used to simplify the procedure of 3PAKE protocols.
(2) Our proposed protocol can satisfy basic security requirements and withstand various well-known attacks.
(3) Our proposed protocol can achieve fairness in the session key agreement such that each user has an equal role.
(4) Our proposed protocol does not involve time-consuming operations, such as modulus exponential and public key encryption/decryption operations, thus it is more efficient in terms of computational cost than other related 3PAKE protocols.

The rest of the paper is organized as follows. Section 2 addresses some background information. Section 3 describes the details of our proposed protocol. Security and performance analyses of our proposed protocol are given in Sections 4 and 5, respectively. Our conclusions are presented in Section 6.

2. **Preliminaries.** In this section, we briefly introduce some fundamental background information related to the 3PAKE protocol. First, we describe the security requirements that most 3PAKE protocols should satisfy, and then we introduce the main building block in the architecture of our proposed protocol.

2.1. **Security requirements.** Most existing 3PAKE protocols satisfy the following security requirements:

(1) **Mutual authentication:** Each user and the server involved in the 3PAKE protocol can authenticate each other's legitimacy. In addition, two users can verify each other's validity successfully.
(2) **Session key agreement:** With the assistance of the trusted server, both users are able to negotiate a common session key that is used to encrypt messages for subsequent, secure communications.
(3) **Security of the session key:** The session key shared between the two users cannot be known by the server or a malicious attacker.
(4) **Perfect forward secrecy:** Perfect forward secrecy ensures that any previously-established session key will not be revealed if the long-term secret keys are compromised.
(5) **Withstanding the impersonation attack:** The impersonation attack refers to two situations: 1) an attacker impersonates a user to cheat the server or the other user and 2) an attacker impersonates the server to cheat users. A good design of the 3PAKE protocol should resist these kinds of attacks.
(6) **Withstanding the man-in-the-middle attack:** Assume that an attacker stands in the middle of any two parties in the 3PAKE protocol and can modify any messages transmitted between these two parties. The man-in-the-middle attack occurs when the attacker lets the two parties believe that they are communicating with each other, when, in fact, each of them is communicating with the attacker rather than the other legitimate party. A good design of the 3PAKE protocol should resist this type of attack.
(7) **Withstanding the replay attack:** In the replay attack, an attacker maliciously repeats or delays valid, transmitted messages to legal entities. A good design of the 3PAKE protocol should resist this type of attack.
(8) **Withstanding the known-key attack:** he known-key attack launches when the compromised session keys can result in the compromise of other session keys. A good design of the 3PAKE protocol should resist this type of attack.
(9) **Withstanding the dictionary attack:** The vast majority of 3PAKE protocols are password-based, thus they should withstand dictionary attacks. The term "dictionary attacks" means that an attacker can guess the passwords of users through a brute-force method, and they are classified into three types, i.e., off-line dictionary attacks; undetectable on-line dictionary attacks; and detectable on-line dictionary attacks.

2.2. **Shamir's secret sharing scheme.** Shamir's secret sharing scheme [8, 9, 10, 11, 30, 31, 32, 33] is the main building block of our proposed protocol. Shamir's secret sharing is a threshold secret sharing mechanism based on the Lagrange interpolating polynomial that can be depicted as follows. There is one dealer D, and there are n users $U = \{u_1, u_2, \cdots, u_n\}$ involved in the scheme, and they execute two phases, i.e., 1) the share generation phase in which the dealer D generates a secret s and divides it into n shares, such that each user gets one share and 2) the secret reconstruction phase in which t or more users can work together by releasing their shares to recover the secret s that was generated by dealer D. Unfortunately, fewer than t shares are unable to recover the correct secret s. The two phases are described in detail below:

Share generation

Step 1: Dealer D randomly selects a polynomial $f(x)$ of degree $t-1$, $f(x) = s + a_1x + a_2x^2 + \cdots + a_{t-1}x^{t-1} \bmod p$, where $s = f(0)$ is the secret, and t coefficients, $s, a_1, a_2, \cdots, a_{t-1}$, are in the finite field $GF(p)$.

Step 2: Dealer D generates n shares $s_i = f(x_i)$ for $i = 1, 2, \cdots, n$, where x_i is the public information of user u_i, such as u_i's ID number.

Step 3: Dealer D sends share s_i to user u_i in a secret channel.

Secret reconstruction

In this process, t users want to use the shares they received to reconstruct the secret s, where $s_{lj} \in \{s_1, s_2, \cdots, s_n\}$ for $j = 1, 2, \cdots, t$ denotes their shares. On the basis of the Lagrange interpolating polynomial, secret s can be reconstructed by calculating $s = f(0) = \sum_{j=1}^{t} s_{lj} \prod_{m=1, m\neq j}^{t} \frac{x_{lm}}{x_{lm} - x_{lj}} \bmod p$.

The distinguished property of Shamir's secret sharing is that it is unconditionally secure, making it feasible and practical for designing a 3PAKE protocol.

3. **Our proposed protocol.** In this section, we propose a novel 3PAKE protocol based on Shamir's secret sharing scheme. The proposed protocol contains three basic entities, i.e., two users (also called clients) and the server. The two users want to communicate with each other over a public channel in an insecure network. To ensure the confidentiality of the transmitted messages, the users must authenticate each other's validity and negotiate a common session key with the assistance of the trusted server. Then, the users can use the session key to encrypt messages, making their subsequent information exchange secure. In addition, our proposed protocol can achieve fairness in the session key agreement, such that the position of each user is fair.

Our proposed protocol consists of two phases, i.e., 1) the initialization phase and 2) the authentication and key agreement phase. According to Table 1, which summarizes the notations used in the proposed protocol, the detailed description of each phase is demonstrated in the following subsections.

3.1. **Initialization phase.** In the initialization phase, some parameters must be set in advance as follows:

(1) A public collision-free one-way hash function $h(\cdot)$ is selected.

(2) Server S and each user u_i share a long-term secret key that is generated by encrypting information of both server S and user u_i. In particular, server S and user u_1 share the secret key $Key_{Su_1} = h(ID_{u_1}||z)$, where ID_{u_1} is the identity of user u_1, and z is the master secret key of server S. In a similar manner, the secret key $Key_{Su_2} = h(ID_{u_2}||z)$ is shared between server S and user u_2.

(3) Symmetric encryption algorithm $E_{Key_{Su_i}}(\cdot)$ and the corresponding decryption algorithm $D_{Key_{Su_i}}(\cdot)$ are selected with the symmetric key Key_{Su_i}.

Figure 1 illustrates the initialization phase.

TABLE 1. List of notations used in our proposed protocol

u_i	The user i
S	The trusted server
ID_{u_i}	The identity of user u_i
z	The master secret key of server S
x_i	A random number
y_i	A random number
$h(\cdot)$	A public collision-free one-way hash function
Key_{Su_i}	The secret key shared between S and u_i
$E_{Key_{Su_i}}(\cdot)$	secure symmetric encryption algorithm with Key_{Su_i}
$D_{Key_{Su_i}}(\cdot)$	secure symmetric decryption algorithm with Key_{Su_i}
k	The session key shared between u_1 and u_2
$\|\|$	The string concatenation operation

3.2. **Authentication and key agreement phase.** Since users u_1 and u_2 cannot authenticate each other directly in the three-party authentication scenario, server S must take the responsibility of achieving a session key agreement between u_1 and u_2. Based on Shamir's secret sharing scheme, u_1 and u_2 can authenticate each other's validity with the help of server S. Furthermore, u_1 and u_2 are able to establish a common session key for future communication after completing the authentication and key agreement phase. This phase is executed as follows, and it is depicted in Figures 2 (a) and 2 (b).

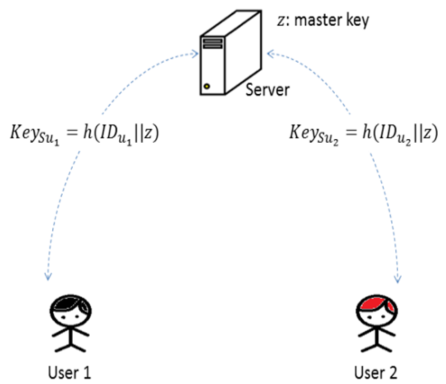
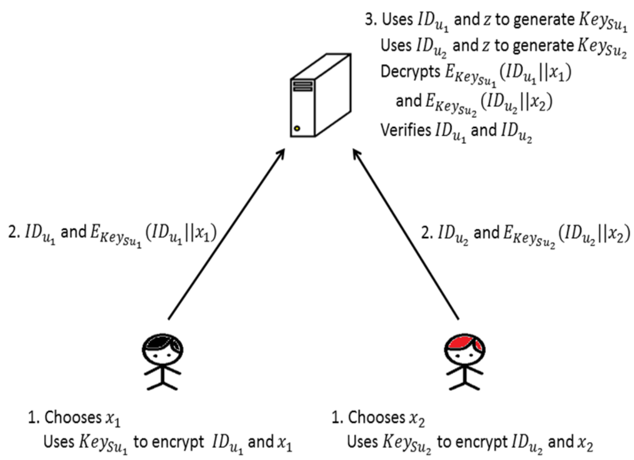
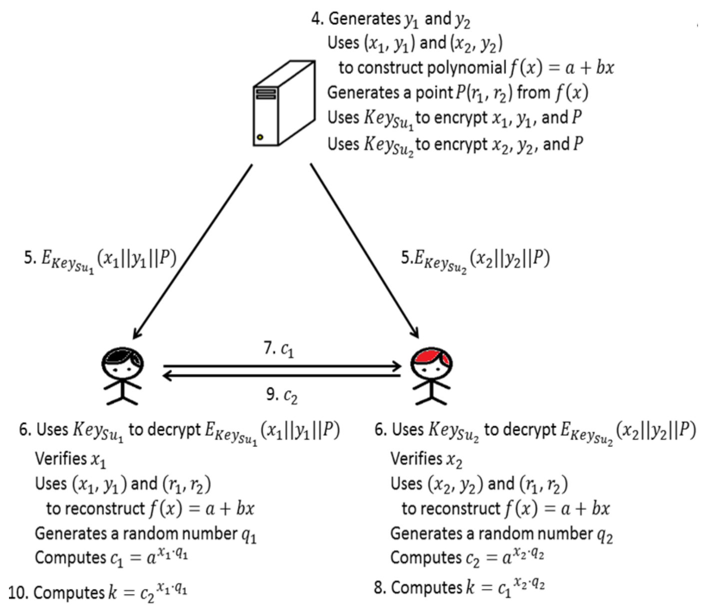


FIGURE 1. Initialization phase in our proposed protocol



(a)



(b)

FIGURE 2. Authentication and key agreement phase in our proposed protocol

Step 1: User u_1 chooses a random number x_1. Then, u_1 encrypts ID_{u_1} and x_1 with key Key_{Su_1} as $E_{Key_{Su_1}}(ID_{u_1}||x_1)$. User u_2 chooses a random number x_2 and then encrypts ID_{u_2} and x_2 with key Key_{Su_2} as $E_{Key_{Su_2}}(ID_{u_2}||x_2)$.

Step 2: u_1 sends ID_{u_1} and $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ to server S. Correspondingly, u_2 sends ID_{u_2} and $E_{Key_{Su_2}}(ID_{u_2}||x_2)$ to server S.

Step 3: Upon receiving the transmitted messages, S uses the received ID_{u_1} and her/his master secret key z to generate key $Key_{Su_1} = h(ID_{u_1}||z)$. S also generates key $Key_{Su_2} = h(ID_{u_2}||z)$. Afterwards, S decrypts $E_{Key_{su_1}}(ID_{u_1}||x_1)$ and $E_{Key_{su_2}}(ID_{u_2}||x_2)$ with Key_{Su_1} and Key_{Su_2}, respectively. Then, S verifies ID_{u_1} and ID_{u_2}.

Step 4: S chooses two random numbers, y_1 and y_2. Then, S constructs a first-degree, interpolated polynomial as $f(x) = a + bx \bmod n$ to pass through two points, i.e., (x_1, y_1) and (x_2, y_2). S also generates an additional point $P = (r_1, r_2)$ from $f(x)$. After that, S uses key Key_{Su_1} to encrypt x_1, y_1, and P as $E_{Key_{Su_1}}(x_1||y_1||P)$ and uses key Key_{Su_2} to encrypt x_2, y_2, and P as $E_{Key_{Su_2}}(x_2||y_2||P)$.

Step 5: S simultaneously sends $E_{Key_{Su_1}}(x_1||y_1||P)$ to u_1 and $E_{Key_{Su_2}}(x_2||y_2||P)$ to u_2.

Step 6: u_1 uses key Key_{Su_1} to decrypt $E_{Key_{Su_1}}(x_1||y_1||P)$ and verifies x_1. Then, u_1 reconstructs the polynomial $f(x)$ by using two points (x_1, y_1) and P. After that, u_1 chooses a random number q_1 and computes $c_1 = a^{x_1 q_1}$. Also, u_2 uses key Key_{Su_2} to decrypt $E_{Key_{Su_2}}(x_2||y_2||P)$ and verifies x_2. Then, u_2 reconstructs the polynomial $f(x)$ by using two points (x_2, y_2) and P. After that, u_2 chooses a random number q_2 and computes $c_2 = a^{x_2 q_2}$.

Step 7: u_1 sends c_1 to u_2.

Step 8: u_2 computes the session key $k = c_1^{x_2 q_2} = a^{x_1 x_2 q_1 q_2}$.

Step 9: u_2 sends c_2 to u_1.

Step 10: u_1 computes the session key $k = c_2^{x_1 q_1} = a^{x_1 x_2 q_1 q_2}$.

According to this phase, server S uses two shadows (points), (x_1, y_1) and (x_2, y_2), to construct a polynomial $f(x) = a + bx \bmod n$. Then, u_1/u_2 can reconstruct $f(x)$ by using the shadows $(x_1, y_1)/(x_2, y_2)$ and (r_1, r_2) he/she holds. Therefore, only u_1 and u_2 know the correct secret a, which will be used later to establish the session key.

4. **Functionality and Security analyses.** In this section, we will demonstrate that our proposed protocol can satisfy the fundamental requirements mentioned in Subsection 2.1. In particular, our proposed protocol can possess multiple functionalities, such as achieving mutual authentication, session key agreement, security of the session key, fairness in the session key agreement, and perfect forward secrecy. In addition, our proposed protocol is secure against various attacks, including impersonation attacks, man-in-the-middle attacks, replay attacks, and known-key attacks.

4.1. **Mutual authentication.** Our proposed protocol can achieve mutual authentication as defined in Subsection 2.1. In Step 2 of the authentication and key agreement phase, u_1 sends ID_{u_1} and $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ to server S, and u_2 sends ID_{u_2} and $E_{Key_{Su_2}}(ID_{u_2}||x_2)$ to server S. Then, S uses $Key_{Su_1}$to decrypt $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ and Key_{Su_2} to decrypt $E_{Key_{Su_2}}(ID_{u_2}||x_2)$, respectively. S checks whether the decrypted ID_{u_1} is equal to the received ID_{u_1} and whether the decrypted ID_{u_2} is equal to the received ID_{u_2}. If they hold, S can be convinced that both u_1 and u_2 are legal. Then, S sends $E_{Key_{Su_1}}(x_1||y_1||P)$ to u_1 and $E_{Key_{Su_2}}(x_2||y_2||P)$ to u_2 at the same time in Step 5. Afterwards, u_1 uses key Key_{Su_1} to decrypt $E_{Key_{Su_1}}(x_1||y_1||P)$, and u_2 uses key Key_{Su_2} to decrypt $E_{Key_{Su_2}}(x_2||y_2||P)$. To verify the integrity of the transmitted message, u_1 and u_2 checks whether the decrypted x_1 and x_2 are identical to the value he/she selected in Step 1. If they are identical, the server is authenticated by both u_1 and u_2. Therefore, each user and the server can authenticate each other's legitimacy. In addition, u_1 sends $c_1 = a^{x_1 q_1}$ to u_2 and u_2 computes the session key $k = c_1^{x_2 q_2}$. Accordingly, u_2 sends $c_2 = a^{x_2 q_2}$ to u_1 and u_1 computes the session key $k = c_2^{x_1 q_1}$. If u_1 and u_2 are legal, they can obtain the same session key k. Thus, they can verify each other's validity successfully.

4.2. **Session key agreement.** Both users establish a common session key with the help of the server S by the following steps. S uses two points, (x_1, y_1) and (x_2, y_2), to construct a polynomial $f(x) = a + bx \bmod n$, where x_1 and x_2 are sent from u_1 and u_2, respectively, while y_1 and y_2 are chosen by S. Then, S generates a point $P = (r_1, r_2)$ from $f(x)$. After receiving $E_{Key_{Su_1}}(x_1||y_1||P)$ sent from S, u_1 uses key Key_{Su_1} to decrypt it and reconstructs $f(x)$ by two using points (x_1, y_1) and P based on Shamir's secret sharing scheme. In a similar way, u_2 can reconstruct $f(x)$ by using two points (x_2, y_2) and P. Since u_1 and u_2 can retrieve the same a from $f(x)$, they agree on a common session key $k = a^{x_1 x_2 q_1 q_2}$.

4.3. **Security of the session key.** Our proposed protocol can ensure the security of the session key such that it can be computed only by users who participated in the protocol. Although the server helps users establish the session key, he/she has no way to obtain this session key. S cannot obtain q_1 from c_1, x_1, and a due to the difficulty associated with solving the discrete logarithm problem. For the same reason, S cannot obtain q_2 from c_2, x_2, and a. Consequently, S cannot get the session key $k = a^{x_1 x_2 q_1 q_2}$. Next, we explain why an attacker cannot obtain the session key k. Even if an attacker intercepts messages $E_{Key_{Su_1}}(ID_{u_1}||x_1)$, $E_{Key_{Su_2}}(ID_{u_2}||x_2)$, $E_{Key_{Su_1}}(x_1||y_1||P)$, $E_{Key_{Su_2}}(x_2||y_2||P)$, c_1, and c_2, he/she cannot utilize the correct key $E_{Key_{Su_1}}$ or $E_{Key_{Su_2}}$ to derive x_1 or x_2. Moreover, the attacker cannot get a, q_1, and q_2, which are kept secret by u_1 and u_2, so the attacker cannot obtain session key k.

4.4. **Fairness in the session key agreement.** Our proposed protocol can achieve fairness in the session key agreement. In most 3PAKE protocols, one user cannot communicate with the server directly but must transmit some information to the other user who then forwards this information to the server. This indicates that one user is placed in a more important position than the other. In contrast, our proposed protocol allows each user to convey messages to the server directly, and, later, the server can help users to establish a common session key with these messages. As a result, each user has an equal role in our proposed protocol.

4.5. **Perfect forward secrecy.** We assume that the master secret key z of server S is compromised in our proposed protocol. Throughout the security analysis, we assume that Evan is an attacker who is able to eavesdrop and intercept the valid data transmission in the communication channel. Therefore, Evan can easily compute long-term secret keys $Key_{Su_1} = h(ID_{u_1}||z)$ and $Key_{Su_2} = h(ID_{u_2}||z)$. Evan also can derive (x_1, y_1) and (x_2, y_2) and obtain a by using these long-term secret keys. However, since Evan cannot obtain randomly chosen numbers q_1 and q_2 that are essential components of the session key, the disclosure of long-term secret keys cannot lead to the compromise of any previously established session key. As a result, our proposed protocol can provide perfect forward secrecy.

4.6. **Withstanding the impersonation attack.** In this subsection, we show that our proposed protocol can withstand impersonation attacks in the authentication and key agreement phase. More specifically, two scenarios for the impersonation attack are described, i.e., (1) Evan's impersonating server S and (2) Evan's impersonating user u_1 or u_2.

Scenario 1. The attacker, Evan, is impersonating server S

Here, as an example, we only take the situation in which Evan impersonates server S to cheat user u_1. The process of cheating user u_2 can be analyzed in a similar way. If Evan wants to act as server S, he intercepts ID_{u_1} and $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ sent by u_1 and ID_{u_2} and $E_{Key_{Su_2}}(ID_{u_2}||x_2)$ sent by u_2. Then, without the correct key Key_{Su_1}/Key_{Su_2} that is shared between S and u_1/u_2, Evan cannot decrypt $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ and $E_{Key_{Su_2}}(ID_{u_2}||x_2)$. Thus, he must forge ${x_1}^*$ and ${x_2}^*$ and generate two random numbers ${y_1}^*$ and ${y_2}^*$ to construct an $f^*(x) = a^* + b^*x \bmod n^*$ by using points $({x_1}^*, {y_1}^*)$ and $({x_2}^*, {y_2}^*)$. S also generates a point $P^* = ({r_1}^*, {r_2}^*)$ from $f^*(x)$. By generating fake keys ${Key_{Su_1}}^*$ and ${Key_{Su_2}}^*$, S sends $E_{{Key_{Su_1}}^*}({x_1}^*||{y_1}^*||P^*)$ to u_1 to cheat her/him. Upon receiving the transmitted message, u_1 uses key Key_{Su_1} to decrypt $E_{{Key_{Su_1}}^*}({x_1}^*||{y_1}^*||P^*)$. However, u_1 finds out that the decrypted ${x_1}^*$ is not equal to x_1, which he or she selected previously. Therefore, u_1 terminates the procedure immediately.

Scenario 2. The attacker Evan is impersonating one user

If Evan attempts to impersonate u_1, he must choose a random number ${x_1}^*$ due to not knowing the real x_1 and forge key ${Key_{Su_1}}^*$ to encrypt ID_{u_1} and ${x_1}^*$. Then, Evan acts as u_1 to send ID_{u_1} and $E_{{Key_{Su_1}}^*}(ID_{u_1}||{x_1}^*)$ to S. However, when S uses the key Key_{Su_1} to decrypt $E_{{Key_{Su_1}}^*}(ID_{u_1}||{x_1}^*)$, he/she observes that the decrypted ID_{u_1} is illegal by comparing it with the correct ID_{u_1}. Consequently, S terminates the procedure. Similarly, Evan fails to impersonate u_2 to pass the authentication process.

Scenarios 1 and 2 indicate that it is impossible for an attacker to launch an impersonation attack successfully in our proposed protocol.

4.7. **Withstanding the man-in-the-middle attack.** In the following, we show how our proposed protocol can prevent a man-in-the-middle attack. According to where the attacker Evan stands, this attack can be classified into two scenarios listed below:

Scenario 1. The attacker, Evan, stands between one user and server S

Let us consider the situation in which Evan stands between u_1 and S. Evan modifies the message $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ to $E_{Key_{Su_1}}(ID_{u_1}||x_1)^*$ and then sends ID_{u_1} and $E_{Key_{Su_1}}(ID_{u_1}||x_1)^*$ to server S, masquerading as u_1 to cheat S. S decrypts $E_{Key_{Su_1}}(ID_{u_1}||x_1)^*$ by using $E_{Key_{Su_1}}$ to extract ID_{u_1}. However, S detects that ID_{u_1} is invalid and terminates the procedure. Alternatively, Evan intercepts $E_{Key_{Su_1}}(x_1||y_1||P)$ sent from S and modifies it as $E_{Key_{Su_1}}(x_1||y_1||P)^*$. Then, Evan transmits the modified message to u_1. u_1 decrypts $E_{Key_{Su_1}}(x_1||y_1||P)^*$ with $E_{Key_{Su_1}}$ to retrieve the parameters inside it. Fortunately, Evan is unable to deceive u_1, and the procedure is terminated when he/she verifies that the decrypted x_1 is different from the one that he/she had chosen before. Thus, the attacker can cheat neither u_1 nor S when standing between them. The case is same for an attacker standing between u_2 and S.

Scenario 2. The attacker, Evan, stands between two users

Evan replaces c_1 with $c_1{}^* = e^{x_e q_e}$ and then sends it to u_2. u_2 computes $k^* = (c_1{}^*)^{x_2 q_2} = e^{x_e q_e{}^{x_2 q_2}}$ and sends c_2 to u_1. Evan intercepts c_2, replaces it with $c_2{}^* = e^{x_e q_e}$, and then sends $c_2{}^*$ back to u_1. u_1 computes $k^{**} = (c_2{}^*)^{x_1 q_1} = e^{x_e q_e{}^{x_1 q_1}}$. Since a can only be recovered by u_1 and u_2 and x_1, x_2, q_1, and q_2 were chosen by u_1 or u_2, Evan cannot share the session key k^* with u_2 and k^{**} with u_1. In other words, u_1 and u_2 can detect the Evan's existence, and, thus, a man-in-the-middle attack cannot be initiated successfully.

4.8. **Withstanding the replay attack.** Our proposed protocol can resist the replay attack by choosing random numbers x_1, x_2, q_1, and q_2 in different sessions. For instance, let us consider the scenario in which Evan replays $E_{Key_{Su_1}}(ID_{u_1}||x_1)$ and $E_{Key_{Su_2}}(ID_{u_2}||x_2)$ in Step 2. S decrypts these two messages to extract x_1 and x_2 and then conveys $E_{Key_{Su_1}}(x_1||y_1||P)$ to u_1 and $E_{Key_{Su_2}}(x_2||y_2||P)$ to u_2. When u_1 obtains x_1 by decrypting $E_{Key_{Su_1}}(x_1||y_1||P)$, he/she can immediately verify that the decrypted x_1 is not fresh. Similarly, u_1 and u_2 can detect the illegitimacy of x_1 and x_2 if Evan replays $E_{Key_{Su_1}}(x_1||y_1||P)$ and $E_{Key_{Su_2}}(x_2||y_2||P)$. Also, if Evan replays c_1 and c_2, u_1 and u_2 are incapable of sharing a common session key, and the procedure is terminated. Consequently, the replay attack will fail.

4.9. **Withstanding the known-key attack.** Our proposed protocol can withstand the known-key attack due to the fact that keys in different sessions are independent of each other. Assume that Evan knows the session key $k = a^{x_1 x_2 q_1 q_2}$ and that he wants to derive another session key $k' = (a')^{x_1{}' x_2{}' q_1{}' q_2{}'}$. Since $\{x_1, x_2, q_1, q_2\}$ and $\{x_1{}', x_2{}', q_1{}', q_2{}'\}$ are randomly selected by u_1 and u_2 in different sessions, Evan cannot mount the known-key attack by computing k' from k.

Table 2 compares the functionalities and security capabilities of our proposed protocol and other, related 3PAKE protocols [7, 17, 22, 23, 26]. The results of the comparisons show that our proposed protocol is superior to the others. In addition, our proposed protocol does not have to resist the dictionary attack because it is based on Shamir's secret sharing scheme without depending on the use of passwords.

TABLE 2. Comparison of functionalities and security capabilities

Protocols	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
Lu and Cao [17]	Yes	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes	No	No
Lee et al. [22]	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yoon and Yoo [23]	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Huang [26]	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Lv et al. [7]	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes
Ours	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	-	-	-

A1: mutual authentication; A2: session key agreement; A3: security of the session key; A4: fairness in the session key agreement; A5: providing perfect forward secrecy; A6: withstanding impersonation attacks; A7: withstanding man-in-the-middle attacks; A8: withstanding replay attacks; A9: withstanding known-key attacks; A10: withstanding off-line dictionary attacks; A11: withstanding undetectable on-line dictionary attacks; A12: withstanding detectable on-line dictionary attacks.

5. **Performance evaluation.** In this section, we evaluate the performance of our 3PAKE protocol. As shown in Table 3, we compared our 3PAKE protocol with other related works [7, 17, 22, 23, 26] based on the dominant operations executed in each protocol, i.e., modulus exponential, hash, pseudo-random, public key encryption/decryption, symmetric encryption/decryption, and Shamir's secret sharing operation. As Table 3 indicates, our proposed protocol eliminates the usage of modulus exponential and public key encryption/decryption operations, which are the most time-consuming tasks. Instead, Shamir's secret sharing and symmetric encryption/decryption operations are utilized to reduce the computational cost significantly.

TABLE 3. Performance Comparison

Protocols	B1 $u_1/u_2/S$	B2 $u_1/u_2/S$	B3 $u_1/u_2/S$	B4 $u_1/u_2/S$	B5 $u_1/u_2/S$	B6 $u_1/u_2/S$
Lu and Cao [17]	3/3/6	3/3/2	1/1/1	0/0/0	0/0/0	0/0/0
Lee et al. [22]	2/2/0	1/1/0	2/3/0	1/2/1	4/4/4	0/0/0
Yoon and Yoo [23]	3/3/4	5/5/6	2/2/1	0/0/0	1/1/2	0/0/0
Huang [26]	2/2/2	3/3/2	1/1/1	0/0/0	0/0/0	0/0/0
Lv et al. [7]	2/2/2	1/1/2	1/2/1	0/0/0	3/4/3	0/0/0
Ours	0/0/0	0/0/2	2/2/2	0/0/0	2/2/4	1/1/1

B1: modulus exponential operation; B2: hash operation; B3: pseudo-random operation; B4: public key encryption/decryption operation; B5: symmetric encryption/decryption operation; B6: Shamir's secret sharing operation.

In Table 4, we show the specifications of our workstation. With this workstation, we used C++ language to implement (1) the code of hash function, and (2) the code of Shamir's secret sharing scheme. After inputting 64 letters for a 512-bit random string and testing 10,000 times, the average time we calculate are about (1) 6.4 ms, and (2) 2.64 ms, respectively. Furthermore, Schneier [34] mentioned that a hash function (MD5/SHA) was about 1000 times faster than an asymmetric cryptosystem (RSA-1024) and that one symmetric cryptosystem (DES) was about 100 times faster than one asymmetric cryptosystem. According to [35] and our experimental results, we can conjecture the computational cost of each encryption/decryption operation in our protocol in Table 5. Besides, we calculate the execution time of each party in our proposed protocol and show it in Table 6.

TABLE 4. Experimental platform in our paper

Device of our workstation	
OS	Windows 7 SP1
CPU	Intel®Core™i7-3770 processor running at 3.40 GHz
RAM	8,192 MB
Else	Western Digital WD5000AAKX-08U6AA0 ATA drive
Language	C++

TABLE 5. Computational cost

Computational cost (sec.)	
Hash operation	6.92×10^{-3}
Public key encryption/decryption operation	6.92
Symmetric encryption/decryption operation	6.92×10^{-2}
Shamir's secret sharing operation	2.64×10^{-3}

TABLE 6. Execution time of our proposed protocol

Execution time (ms)	
User 1	141.04 ms
User 2	141.04 ms
Server	293.28 ms

6. **Conclusions.** In this paper, we proposed a novel 3PAKE protocol based on Shamir's secret sharing scheme. Our proposed protocol possesses multiple functionalities, such as achieving mutual authentication, session key agreement, the security of session key, fairness in the session key agreement, and perfect forward secrecy. In addition, our proposed protocol is secure against various attacks, including impersonation attacks, man-in-the-middle attacks, replay attacks, and known-key attacks. Our performance analysis showed that the computational cost of our proposed protocol is lower than that of Lv et al.'s protocol.

REFERENCES

[1] J. J. Zhao and D. W. Gu, Provably secure three-party password-based authenticated key exchange protocol, *Information Sciences*, vol.184, no.1, pp.310-323, 2012.
[2] H. Q. Liang, J. T. Hu and S. H. Wu, Re-attack on a three-party password-based authenticated key exchange protocol, *Mathematical and Computer Modelling*,vol.57, no.5-6, pp.1175-1183, 2013.
[3] T. Y. Chang, M. S. Hwang and W. P. Yang, A communication-efficient three-party password authenticated key exchange protocol, *Information Sciences*, vol.181, no.1, pp.217-226, 2011.
[4] X.Hu, Z. Chen and F. Q. Li, New identity-based three-party authenticated key agreement protocol with provable security, *Journal of Network and Computer Applications*, vol.36, no.2, pp.927-932, 2012.
[5] K. A. Shim, A round-optimal three-party ID-based authenticated key agreement protocol, *Information Sciences*, vol.186, no.1, pp.239-248, 2012.

[6] H. C. Tsai and C. C. Chang, Provably secure three party encrypted key exchange scheme with explicit authentication, *Information Sciences*, vol.238, pp.242-249, 2013.

[7] C. Lv, M. D. Ma, H. Li, J. F. Ma and Y. V. Zhang, An novel three-party authenticated key exchange protocol using one-time key, *Journal of Network and Computer Applications*, vol.36, no.1, pp.498-503, 2013.

[8] C. Guo and C. C. Chang, An authenticated group key distribution protocol based on the generalized Chinese remainder theorem, *International Journal of Communication Systems*, vol.27, no.1, pp.126-134, 2014.

[9] Y. J. Liu, C. C. Chang and S. C. Chang, A residual number system oriented group key distribution mechanism, *International Journal of Information Processing and Management*, vol.4, no.3, pp.146-155, 2013.

[10] L. Harn and C. L. Lin, Authenticated group key transfer protocol based on secret sharing, *IEEE Transactions on Computers*, vol.59, no.6, pp.842-846, 2010.

[11] Y. J. Liu, L. Harn and C. C. Chang, An authenticated group key distribution mechanism using theory of numbers, *International Journal of Communication Systems*, vol.27, no.11, pp.3502-3512, 2014.

[12] S. M. Bellovin and M. Michael, Encrypted key exchange: Password-based protocols secure against dictionary attacks, *roceedings of IEEE Computer Society Symposium on Research in Security and Privacy*, USA, pp.72-84, 1992.

[13] K. Kazukuni, Pretty-simple password-authenticated key-exchange protocol proven to be secure in the standard model, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol.85, no.10, pp.2229-2237, 2002.

[14] M. Abdalla and D. Pointcheval, Simple password-based encrypted key exchange protocols, *Proceedings of Topics in Cryptology CT-RSA' 05, Lecture Notes in Computer Science, Communications and Computer Sciences*, vol.3376, pp.191-208, 2005.

[15] J. N. Katz, P. L. MacKenzieb , G. Tabana and V. Gligorc, Two-server password-only authenticated key exchange, *Journal of Computer and System Sciences*, vol.78, no.2, pp.651-669, 2012.

[16] A. Joux, A one round protocol for tripartite DiffieHellman, *Proceedings of 4th International Symposium on Algorithmic Number Theory, Lecture Notes in Computer Science*, vol.1838, pp.385-394, 2000.

[17] R. X. Lu and Z. F. Cao, Simple three-party key exchange protocol, *Computers & Security*, vol.26, no.1, pp.94-97, 2007.

[18] H. Guo, Z. J. Li, Y. Mu and X. Y. Zhang, Cryptanalysis of simple three-party key exchange protocol, *Computers & Security*, vol.27, no.1, pp.16-21, 2008.

[19] H. R. Chung and W. C. Ku, Three weaknesses in a simple three-party key exchange protocol, *Information Sciences*, vol.178, no.1, pp.220-229, 2008.

[20] H. S. Kim and J. Y. Choi, Enhanced password-based simple three-party key exchange protocol, *Computers & Electrical Engineering*, vol.35, no.1, pp.107-114, 2009.

[21] H. T. Yeh, H. M. Sun and T. L. Hwang, Efficient three-party authentication and key agreement protocols resistant to password guessing attacks, *Journal of Information Science and Engineering*, vol.19, no.6, pp.1059-1070, 2003.

[22] T. F. Lee, J. L. Liu, M. J. Sung, S. B. Yang and C. M. Chen, Communication-efficient three-party protocols for authentication and key agreement, *Computers & Mathematics with Applications*, vol.58, no.4, pp.641-648, 2009.

[23] F. J. Yoon and K. Y. Yoo, Improving the novel three-party encrypted key exchange protocol, *Computer Standards & Interfaces*, vol.30, no.5, pp.309-314, 2008.

[24] C. C. Chang and Y. F. Chang, A novel three-party encrypted key exchange protocol, *Computer Standards & Interfaces*, vol.26, no.5, pp.471-476, 2004.

[25] N. W. Lo and K. H. Yeh, Cryptanalysis of two three-party encrypted key exchange protocols, *Computer Standards & Interfaces*, vol.31, no.6, pp.1167-1174, 2009.

[26] H. F. Huang, A simple threeparty passwordbased key exchange protocol, *International Journal of Communication Systems*, vol.22, no.7, pp.857-862, 2009.

[27] E. J. Yoon and K. Y. Yoo, Cryptanalysis of a simple threeparty passworbased key exchange protocol, *International Journal of Communication Systems*, vol.24, no.4, pp.532-542, 2011.

[28] E. J. Yoon, On the security of Lv et al.'s three-party authenticated key exchange protocol using one-time key, *Advanced Infocomm Technology*, vol.7593, pp.191-198, 2013.

[29] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, vol.22, no.6, pp.644-654, 1976.

[30] A. Shamir, How to share a secret, *Communications of the ACM*, vol.22, no.11, pp.612-613, 1979.
[31] G. R. Blakley, Safeguarding cryptographic keys, *Proceedings of American Federation of Information Processing Societies National Computer Conference*, New York, USA, vol.48, pp.313-317, Nov. 1979.
[32] L. Harn and C. L. Lin, Strong (n, t, n) verifiable secret sharing scheme, *Information Sciences*, vol.180, no.16, pp.3059-3064, 2010.
[33] L. Harn, Group Authentication, *IEEE Transactions on Computers*, vol.62, no.9, pp.1893-1898, 2013.
[34] B. Schneier, Applied cryptography, protocols, algorithms, and source code in C, *John Wiley and Sons Inc., 2nd Edition*, New York, USA, 1996.
[35] C. C. Lee, A simple key agreement scheme based on chaotic maps for VSAT satellite communications, *International Journal of Satellite Communications and Networking*, vol.31, no.4, pp.177-186, 2013.