J Supercomput (2014) 70:1002–1022
DOI 10.1007/s11227-014-1273-z

An efficient client–client password-based authentication scheme with provable security

Mohammad Sabzinejad Farash · Mahmoud Ahmadian Attari

Published online: 30 July 2014

Abstract Recently, Tso proposed a three-party password-based authenticated key exchange (3PAKE) protocol. This protocol allows two clients to authenticate each other and establish a secure session key through a server over an insecure channel. The main security goals of such protocols are authentication and privacy. However, we show that Tso's protocol achieves neither authentication goal nor privacy goal. In this paper, we indicate that the privacy and authentication goals of Tso's protocol will be broken by off-line password guessing attack and impersonation attack, respectively. To overcome the weaknesses, we propose an improved 3PAKE protocol to achieve more security and performance than related protocols. The security of the proposed improved protocol is proved in random oracle model.

Keywords Password-based authentication · Key exchange protocol · Off-line password guessing attack · Impersonation attack · Random oracle model

1 Introduction

Authenticated key exchange (AKE) protocols (e.g., [1–10]) help communicating entities, who are communicating over an insecure network, to establish a secret session key to be used for protecting their subsequent communication. Password-based authenticated key exchange (PAKE) protocol is a type of AKE protocols which enables two or more communication entities, who only share a weak, low-entropy and easily memo-

M. S. Farash (✉)
Faculty of Mathematical Sciences and Computer, Kharazmi University, Tehran, Iran
e-mail: m.sabzinejad@gmail.com; sabzinejad@tmu.ac.ir; sabzinejad@khu.ac.ir

M. A. Attari
Faculty of Electrical and Computer Engineering, K.N. Toosi University of Technology, Tehran, Iran

rable passwords, to authenticate each other and establish a high-entropy secret session key.

PAKE protocols were first proposed in the two-party setting (2PAKE) which are quite suitable for the client–server architecture (e.g., [11–18]). However, these protocols are very inconvenient for large-scale client–client communication environments. Since each client needs to remember different password for each partner who communicates with, for a large network, it may strain the storage capacity of the clients. To avoid this problem, PAKE protocols in the three-party setting (3PAKE) are developed. In a 3PAKE protocol, a trusted server mediates between two communication clients and each client only needs to share a password with the server.

1.1 Related works

In order to design a secure and practicable 3PAKE, many protocols have been proposed. The main security threats for the 3PAKE protocols are password guessing attacks. To protect these protocols against dictionary attacks, there are three main approaches: using the server public key (e.g., [19–22]), using symmetric cryptosystems (e.g., [23–25]), and without using server public keys and symmetric cryptosystems (e.g., [26–30,30–42]). The 3PAKE protocols using symmetric cryptosystems requires a stronger assumption "the ideal cipher model" to prove the security [30]. Recently, Xiong et al. [25] demonstrated that all of the 3PAKE protocols without server public keys are not secure against Key Compromise Impersonation (KCI) attack. Therefore, the 3PAKE protocols which uses only server public keys to prevent password guessing attacks are more secure and applicable than the other two approaches.

In 2009, Huang [38] proposed a 3PAKE protocol without server public key and symmetric cryptosystems. However, Yoon and Yoo [39] demonstrated that Huang's 3PAKE protocol is vulnerable to undetectable online password guessing attacks and off-line password guessing attacks by any other user. Based on the Yoon and Yoo's attacks, Wu et al. [40] showed that Huang's protocol is also vulnerable to key compromise impersonation attack and proposed an enhanced protocol which uses server public key. In 2011, Chang et al. [41] applied XOR operations to removed both the server public keys and the symmetric cryptosystem for constructing a communication efficient 3PAKE protocol. However, Wu et al. [30] pointed out that Chang et al.'s 3PAKE is insecure against password guessing attacks and proposed an improved protocol. To overcome the security problems of the Chang et al.'s scheme, Tso [42] also proposed an improved scheme without using the server public key or symmetric cryptosystems. Recently, Xiong et al. [25] demonstrated that the Wu et al.'s 3PAKE protocol is vulnerable to key compromise impersonation (in short, KCI) attack.

1.2 Contribution

The contribution of this paper is twofold. First, the paper indicates that Tso's protocol [42] not only is still vulnerable to off-line password guessing attack, but also is insecure against impersonation attack. Second, it proposes a more secure and efficient 3PAKE protocol to overcome the security flaws of the related protocols.

Table 1 The notations

Notation	Description
A, B	Legitimate users
pw	The password of a legitimate user
id	The unique identity of a legitimate user
S	A remote server
p	A large prime number
G	A multiplicative group of order p
q	A large prime with $q\|(p-1)$
g	A generator of order q
$\mathbb{Z}_q^*$	The non-zero residues mod q
x, Y	Server's private key and public key, respectively
$h(.)$	A Conventional hash function $h : \{0,1\}^* \rightarrow \{0,1\}^l$

1.3 Organization

The rest of this paper is organized as follows. In Sect. 2, we review the Tso's 3PAKE protocol. In Sect. 3, we show the vulnerabilities of Tso's protocol. An enhanced 3PAKE protocol is proposed in Sect. 4. We analyze the security and performance of the proposed scheme in Sects. 5 and 6, respectively. Finally, we conclude our paper in Sect. 7.

2 A brief review of Tso's 3PAKE protocol

This section briefly reviews Tso's 3PAKE protocol [42].

2.1 Notations

The notations used throughout this paper are summarized in Table 1.

2.2 Protocol description

For a detailed analysis, we review Tso's 3PAKE protocol [42]. The details of this protocol, shown in Fig. 1, are as follows:

Step 1: A sends the identities id_A and id_B to S as initial request.
Step 2: Upon receiving the messages $\{id_A, id_B\}$, S chooses two random numbers $e_{S1}, e_{S2} \in_R \mathbb{Z}_q^*$, computes

$$R_{S1} = g^{e_{S1}+pw_A} \bmod p,$$
$$R_{S2} = g^{e_{S2}+pw_B} \bmod p,$$

and sends $\{R_{S1}, R_{S2}\}$ to A.

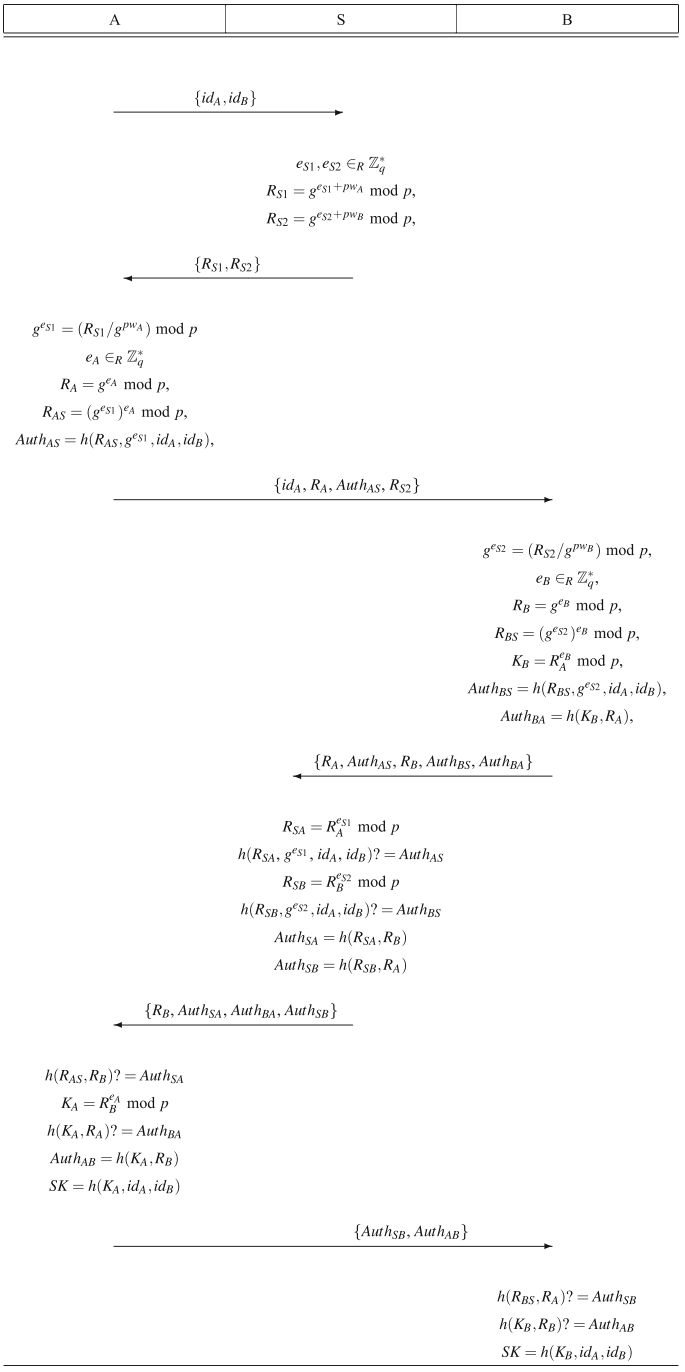

Fig. 1 Tso's 3PAKE protocol [42]

Step 3: After receiving the message $\{R_{S1}, R_{S2}\}$, A firstly retrieves $g^{e_{S1}}$ by computing $(R_{S1}/g^{pw_A}) \bmod p$. A then chooses a random number $e_A \in_R \mathbb{Z}_q^*$ to compute

$$\begin{aligned} R_A &= g^{e_A} \bmod p, \\ R_{AS} &= (g^{e_{S1}})^{e_A} \bmod p, \\ Auth_{AS} &= h(R_{AS}, g^{e_{S1}}, id_A, id_B), \end{aligned}$$

and sends $\{id_A, R_A, Auth_{AS}, R_{S2}\}$ to B.
Step 4: After receiving the message $\{id_A, R_A, Auth_{AS}, R_{S2}\}$, B retrieves $g^{e_{S2}}$ by computing $(R_{S2}/g^{pw_B}) \bmod p$. B then chooses a random number $e_B \in_R \mathbb{Z}_q^*$ to compute

$$\begin{aligned} R_B &= g^{e_B} \bmod p, \\ R_{BS} &= (g^{e_{S2}})^{e_B} \bmod p, \\ K_B &= R_A^{e_B} \bmod p, \\ Auth_{BS} &= h(R_{BS}, g^{e_{S2}}, id_A, id_B), \\ Auth_{BA} &= h(K_B, R_A), \end{aligned}$$

and sends $\{R_A, Auth_{AS}, R_B, Auth_{BS}, Auth_{BA}\}$ to S.
Step 5: After receiving the message $\{R_A, Auth_{AS}, R_B, Auth_{BS}, Auth_{BA}\}$, S authenticates A and B, separately. To authenticate A, S computes $R_{SA} = R_A^{e_{S1}} \bmod p$ and verifies $h(R_{SA}, g^{e_{S1}}, id_A, id_B)? = Auth_{AS}$. To authenticate B, S computes $R_{SB} = R_B^{e_{S2}} \bmod p$ and verifies $h(R_{SB}, g^{e_{S2}}, id_A, id_B)? = Auth_{BS}$. If the two results are positive, S computes $Auth_{SA} = h(R_{SA}, R_B)$ and $Auth_{SB} = h(R_{SB}, R_A)$. Then S sends $\{R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}\}$ to A.
Step 6: After receiving the message $\{R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}\}$, A checks if $h(R_{AS}, R_B)? = Auth_{SA}$. If it holds, S is authenticated by A; then A computes $K_A = R_B^{e_A} \bmod p$, and verifies $h(K_A, R_A)? = Auth_{BA}$. If it holds, B is authenticated by A. Then, A computes $Auth_{AB} = h(K_A, R_B)$ and sends $\{Auth_{SB}, Auth_{AB}\}$ to B. Finally, A computes the session key as $SK = h(K_A, id_A, id_B) = h(g^{e_A e_B}, id_A, id_B)$.
Step 7: After receiving the message $\{Auth_{SB}, Auth_{AB}\}$, B checks if $h(R_{BS}, R_A)? = Auth_{SB}$. If it holds, S is authenticated by B; then B verifies $h(K_B, R_B)? = Auth_{AB}$. If it holds, A is authenticated by B. Finally, B computes the session key as $SK = h(K_B, id_A, id_B) = h(g^{e_A e_B}, id_A, id_B)$.

3 Cryptanalysis of Tso's protocol

The main security goals of 3PAKE protocols are authentication and privacy. However, this section indicates that Tso's protocol [42] does not achieve the security goals. We Show that Tso's protocol is vulnerable to off-line password guessing attack and impersonation attack.

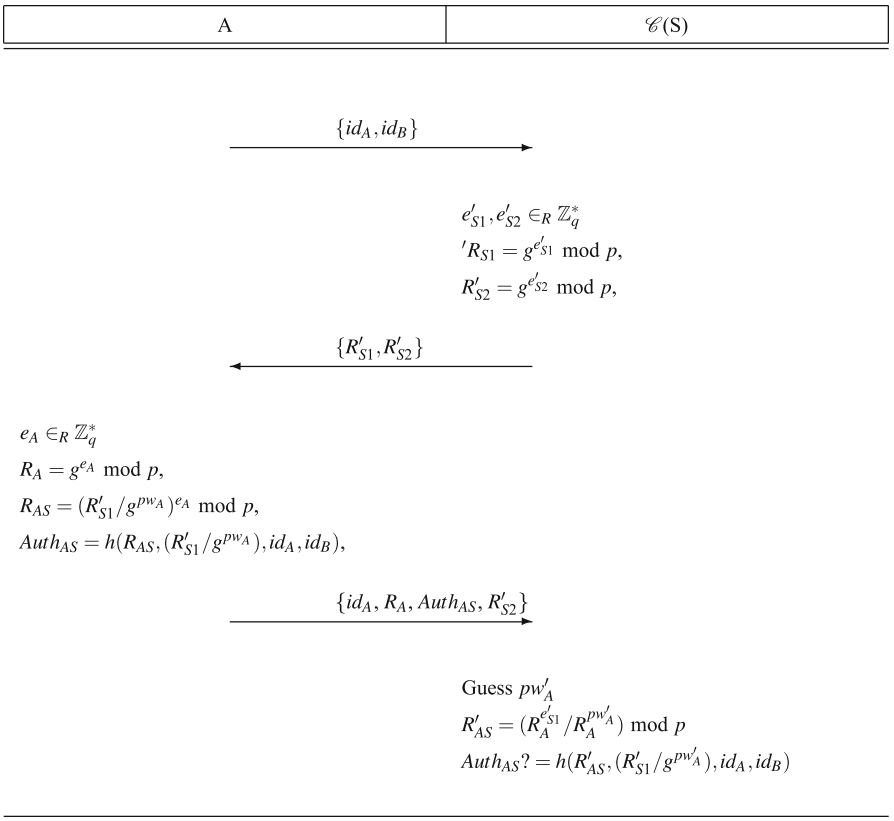

Fig. 2 Off-line password guessing attack on Tso's 3PAKE protocol

3.1 Off-line password guessing attack

Here, we show that an active adversary can successfully guess the users' correct passwords which leads to completely breaking Tso's protocol [42]. The details of the proposed off-line password guessing attack, outlined in Fig. 2, are as follows:

Step 1: When the user A wishes to communicate with the user B through the server S, and sends the initial message $\{id_A, id_B\}$ to S, the active adversary $\mathscr{C}$ intercepts the message and chooses two random numbers $e'_{S1}, e'_{S2} \in_R \mathbb{Z}_q^*$ to computes

$$R'_{S1} = g^{e'_{S1}} \bmod p,$$
$$R'_{S2} = g^{e'_{S2}} \bmod p.$$

$\mathscr{C}$ then returns $\{R'_{S1}, R'_{S2}\}$ to A.
Step 2: After receiving the message $\{R'_{S1}, R'_{S2}\}$, A chooses a random number $e_A \in_R \mathbb{Z}_q^*$ to compute

$$\begin{aligned} R_A &= g^{e_A} \bmod p, \\ R_{AS} &= (R'_{S1}/g^{pw_A})^{e_A} \bmod p, \\ Auth_{AS} &= h(R_{AS}, (R'_{S1}/g^{pw_A}), id_A, id_B), \end{aligned} \tag{1}$$

and sends $\{id_A, R_A, Auth_{AS}, R'_{S2}\}$ to B.

Step 3: $\mathscr{C}$ intercepts the message $\{id_A, R_A, Auth_{AS}, R'_{S2}\}$ and tries to guess A's correct password pw_A. To do so, $\mathscr{C}$ performs the following steps

1. Guess a password pw'_A.
2. Compute
$$R'_{AS} = (R_A^{e'_{S1}}/R_A^{pw'_A}) \bmod p. \tag{2}$$
3. Check if
$$Auth_{AS} = h(R'_{AS}, (R'_{S1}/g^{pw'_A}), id_A, id_B). \tag{3}$$
4. If Eq. (3) holds, $\mathscr{C}$ confirms that the guessed password pw'_A is the correct one. Otherwise, $\mathscr{C}$ chooses another password pw'_A and repeatedly performs above steps to obtain the correct password.

Proposition 1 *Equation* (3) *holds for a correct guess of* $pw'_A = pw_A$.

Proof Firstly, we show that the Eqs. (1) and (2) are equal for $pw'_A = pw_A$ as follows:

$$\begin{aligned} R'_{AS} &= (R_A^{e'_{S1}}/R_A^{pw'_A}) \bmod p \\ &= ((g^{e_A})^{e'_{S1}}/(g^{e_A})^{pw_A}) \bmod p \\ &= ((g^{e'_{S1}})^{e_A}/(g^{pw_A})^{e_A}) \bmod p \\ &= (g^{e'_{S1}}/g^{pw_A})^{e_A} \bmod p \\ &= (R'_{S1}/g^{pw_A})^{e_A} \bmod p \\ &= R_{AS}. \end{aligned}$$

Thus,

$$\begin{aligned} Auth_{AS} &= h(R_{AS}, (R'_{S1}/g^{pw_A}), id_A, id_B) \\ &= h(R'_{AS}, (R'_{S1}/g^{pw'_A}), id_A, id_B). \end{aligned}$$

It fulfills the proof. □

3.2 Impersonation attack

Here, we show that an adversary who obtained the correct password of a special user can easily masquerade as the legitimate server and another user. Suppose the adversary $\mathscr{C}$ obtained A's password pw_A. To impersonate the server S and B, the adversary $\mathscr{C}$ performs as follows (outlined in Fig. 3):

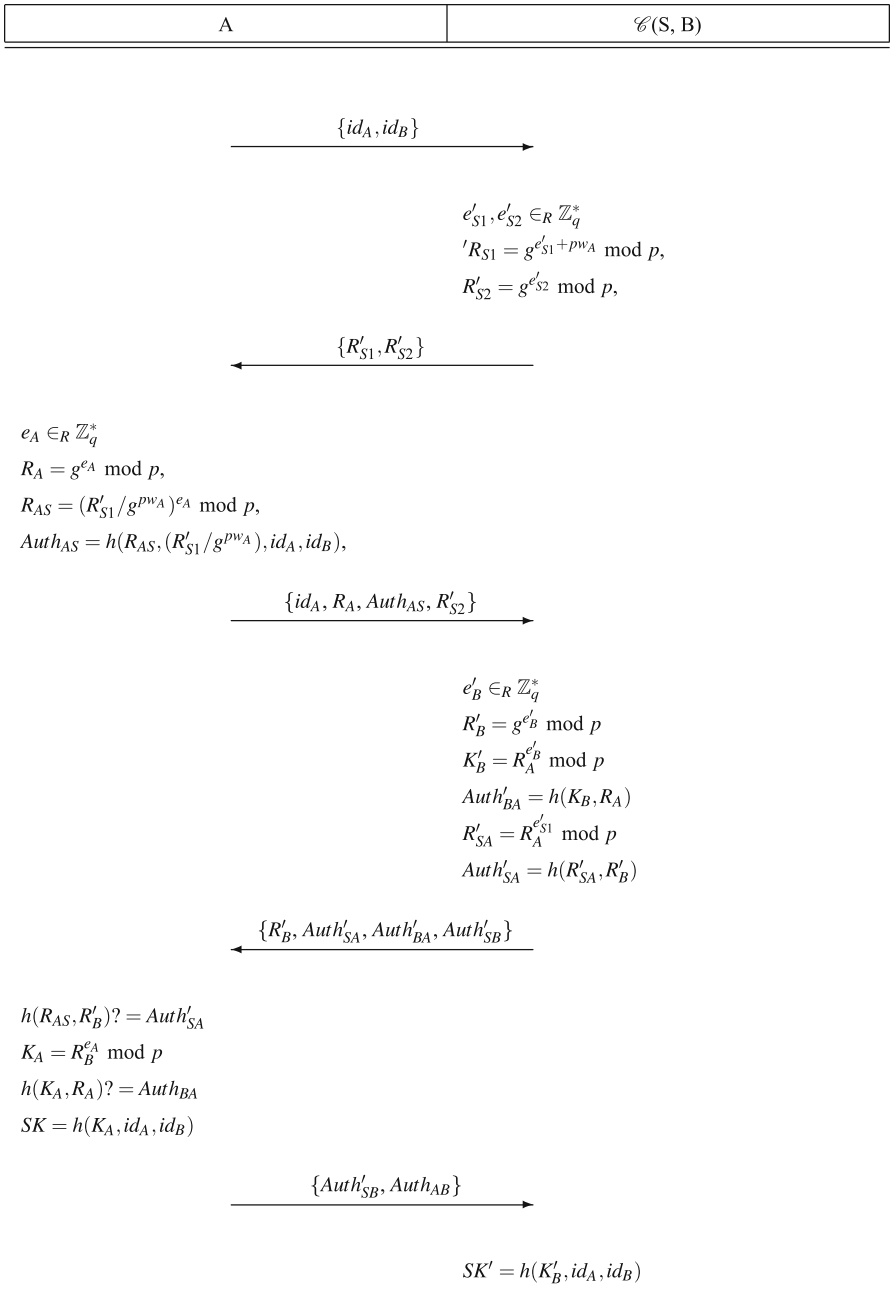

Fig. 3 Impersonation attack on Tso's 3PAKE protocol

Step 1: When the user A wishes to communicate with the user B through the server S, and sends the initial message $\{id_A, id_B\}$ to S, the active adversary $\mathscr{C}$ intercepts the message and chooses two random numbers $e'_{S1}, e'_{S2} \in_R \mathbb{Z}_q^*$ to computes

$$R'_{S1} = g^{e'_{S1}+pw_A} \bmod p,$$
$$R'_{S2} = g^{e'_{S2}} \bmod p.$$

$\mathscr{C}$ then returns $\{R'_{S1}, R'_{S2}\}$ to A.
Step 2: After receiving the message $\{R'_{S1}, R'_{S2}\}$, A chooses a random number $e_A \in_R \mathbb{Z}_q^*$ to compute

$$\begin{aligned} R_A &= g^{e_A} \bmod p, \\ R_{AS} &= (R'_{S1}/g^{pw_A})^{e_A} \bmod p, \\ Auth_{AS} &= h(R_{AS}, (R'_{S1}/g^{pw_A}), id_A, id_B), \end{aligned} \tag{4}$$

and sends $\{id_A, R_A, Auth_{AS}, R'_{S2}\}$ to B.
Step 3: $\mathscr{C}$ intercepts the message $\{id_A, R_A, Auth_{AS}, R'_{S2}\}$ and tries to masquerade as both B and S. To masquerade as B, $\mathscr{C}$ chooses a random number $e'_B \in_R \mathbb{Z}_q^*$, and computes

$$R'_B = g^{e'_B} \bmod p,$$
$$K'_B = R_A^{e'_B} \bmod p,$$
$$Auth'_{BA} = h(K_B, R_A).$$

To masquerade as S, $\mathscr{C}$ computes

$$R'_{SA} = R_A^{e'_{S1}} \bmod p,$$
$$Auth'_{SA} = h(R'_{SA}, R'_B),$$

and chooses random string $Auth'_{SB}$. $\mathscr{C}$ then sends $\{R'_B, Auth'_{SA}, Auth'_{BA}, Auth'_{SB}\}$ to A.
Step 4: After receiving the message $\{R'_B, Auth'_{SA}, Auth'_{BA}, Auth'_{SB}\}$, A checks if

$$h(R_{AS}, R'_B)? = Auth'_{SA}. \tag{5}$$

If it holds, A ensures that the received message was sent by S; then A computes

$$K_A = R_B^{e_A} \bmod p, \tag{6}$$

and verifies

$$h(K_A, R_A)? = Auth'_{BA}. \tag{7}$$

If it holds, A ensures that $Auth'_{BA}$ was generated by B, and computes $Auth_{AB} = h(K_A, R_B)$ and sends $\{Auth'_{SB}, Auth_{AB}\}$ to B. Finally, A computes the session key as $SK = h(K_A, id_A, id_B)$.

Step 5: $\mathscr{C}$ intercepts the message $\{Auth'_{SB}, Auth_{AB}\}$ and computes the session key as $SK' = h(K'_B, id_A, id_B)$.

Proposition 2 *By performing the above steps, the adversary $\mathscr{C}$ can successfully masquerade as the server S.*

Proof The user A accepts the adversary $\mathscr{C}$ as the server S if Eq. 5 holds. We show that it holds as follows:

$$\begin{aligned} h(R_{AS}, R'_B) &= h((R'_{S1}/g^{pw_A})^{e_A}), R'_B) \\ &= h(((g^{e'_{S1}+pw_A})/g^{pw_A})^{e_A}), R'_B) \\ &= h((g^{e'_{S1}e_A}), R'_B) \\ &= h(R_A^{e'_{S1}}, R'_B) \\ &= h(R'_{SA}, R'_B) \\ &= Auth'_{SA}. \end{aligned}$$

□

Proposition 3 *By performing the above steps, the adversary $\mathscr{C}$ can successfully masquerade as the user B.*

Proof The user A accepts the adversary $\mathscr{C}$ as the user B if Eq. 7 holds. We show that it holds as follows:

$$\begin{aligned} h(K_A, R_A) &= h((R_B^{'e_A}), R_A) \\ &= h((g^{e'_B e_A}), R_A) \\ &= h((R_A^{e'_B}), R_A) \\ &= h(K'_B, R_A) \\ &= Auth'_{BA}. \end{aligned}$$

□

Proposition 4 *By performing the above steps, the adversary $\mathscr{C}$ can successfully establish a common session key with the user A.*

Proof The session key SK computed by A is equal to the session key SK' computed by the adversary $\mathscr{C}$, since

$$\begin{aligned} SK &= h(K_A, id_A, id_B) \\ &= h((R_B^{'e_A}), id_A, id_B) \\ &= h((g^{e'_B e_A}), id_A, id_B) \\ &= h((R_A^{e'_B}), id_A, id_B) \end{aligned}$$

$$= h(K'_B, id_A, id_B)$$
$$= SK'.$$

□

According to the Propositions 2, 3 and 4, Tso's protocol is vulnerable to impersonation attacks.

4 The proposed improved 3PAKE protocol

As can be clearly seen in Sect. 3, the vulnerabilities of Tso's scheme is due to the computation of the parameters R_{S1} and R_{S2}. To overcome the vulnerabilities, we change the computation of these parameters as $R_{S1} = g^{e_{S1}} + g^{pw_A}$ and $R_{S2} = g^{e_{S2}} + g^{pw_B}$. The details of the proposed improved scheme, shown in Fig. 4, are described in the following steps:

Step 1: A sends the identities id_A and id_B to S as initial request.
Step 2: Upon receiving the messages $\{id_A, id_B\}$, S chooses two random numbers $e_{S1}, e_{S2} \in_R \mathbb{Z}_q^*$, computes

$$R_{S1} = g^{e_{S1}} + g^{pw_A} \bmod p,$$
$$R_{S2} = g^{e_{S2}} + g^{pw_B} \bmod p,$$

and sends $\{R_{S1}, R_{S2}\}$ to A.
Step 3: After receiving the message $\{R_{S1}, R_{S2}\}$, A firstly retrieves $g^{e_{S1}}$ by computing $(R_{S1} - g^{pw_A}) \bmod p$. A then chooses a random number $e_A \in_R \mathbb{Z}_q^*$ to compute

$$R_A = g^{e_A} \bmod p,$$
$$R_{AS} = (g^{e_{S1}})^{e_A} \bmod p,$$
$$Auth_{AS} = h(R_{AS}, g^{e_{S1}}, id_A, id_B),$$

and sends $\{id_A, R_A, Auth_{AS}, R_{S2}\}$ to B.
Step 4: After receiving the message $\{id_A, R_A, Auth_{AS}, R_{S2}\}$, B retrieves $g^{e_{S2}}$ by computing $(R_{S2} - g^{pw_B}) \bmod p$. B then chooses a random number $e_B \in_R \mathbb{Z}_q^*$ to compute

$$R_B = g^{e_B} \bmod p,$$
$$R_{BS} = (g^{e_{S2}})^{e_B} \bmod p,$$
$$K_B = R_A^{e_B} \bmod p,$$
$$Auth_{BS} = h(R_{BS}, g^{e_{S2}}, id_A, id_B),$$
$$Auth_{BA} = h(K_B, R_A),$$

and sends $\{R_A, Auth_{AS}, R_B, Auth_{BS}, Auth_{BA}\}$ to S.

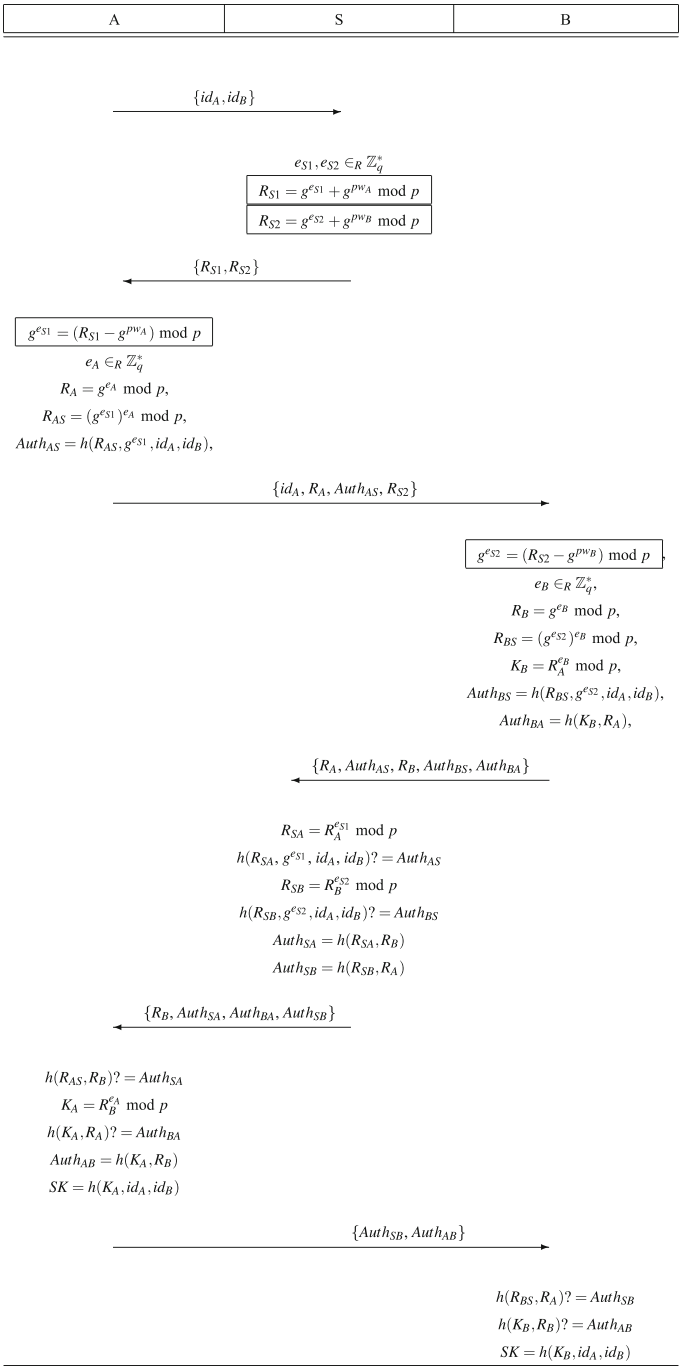

Fig. 4 The improved 3PAKE protocol

Step 5: After receiving the message $\{R_A, Auth_{AS}, R_B, Auth_{BS}, Auth_{BA}\}$, S authenticates A and B, separately. To authenticate A, S computes $R_{SA} = R_A^{e_{S1}} \bmod p$ and verifies $h(R_{SA}, g^{e_{S1}}, id_A, id_B)? = Auth_{AS}$. To authenticate B, S computes $R_{SB} = R_B^{e_{S2}} \bmod p$ and verifies $h(R_{SB}, g^{e_{S2}}, id_A, id_B)? = Auth_{BS}$. If the two results are positive, S computes $Auth_{SA} = h(R_{SA}, R_B)$ and $Auth_{SB} = h(R_{SB}, R_A)$. Then S sends $\{R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}\}$ to A.
Step 6: After receiving the message $\{R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}\}$, A checks if $h(R_{AS}, R_B)? = Auth_{SA}$. If it holds, S is authenticated by A; then A computes $K_A = R_B^{e_A} \bmod p$, and verifies $h(K_A, R_A)? = Auth_{BA}$. If it holds, B is authenticated by A. Then, A computes $Auth_{AB} = h(K_A, R_B)$ and sends $\{Auth_{SB}, Auth_{AB}\}$ to B. Finally, A computes the session key as $SK = h(K_A, id_A, id_B) = h(g^{e_A e_B}, id_A, id_B)$.
Step 7: After receiving the message $\{Auth_{SB}, Auth_{AB}\}$, B checks if $h(R_{BS}, R_A)? = Auth_{SB}$. If it holds, S is authenticated by B; then B verifies $h(K_B, R_B)? = Auth_{AB}$. If it holds, A is authenticated by B. Finally, B computes the session key as $SK = h(K_B, id_A, id_B) = h(g^{e_A e_B}, id_A, id_B)$.

5 Security analysis of the improved protocol

In this section, we show that our improved protocol is secure in the random oracle model. We start with the formal security model and the algorithm assumption that will be used in our proof.

5.1 Security model

In order to make our scheme resist the known attacks to 3PAKE protocol, we use the method of provable security. The security proof is based on the model proposed by Abdalla and Pointcheval [43]. The model that we use is as follows.

5.1.1 Participants

A 3PAKE protocol Π runs in a network of a number of interconnected participants where each participant is either a client $U \in \mathscr{U}$ or a trusted server $S \in \mathscr{S}$. The set $\mathscr{S}$ is assumed to involve only a single server for simplicity. Each of the participants may have several instances called oracles involved in distinct executions of the protocol Π. We refer to ith instance of U (resp. S) in a session as Π_U^i (resp. Π_S^i). Every instance Π_U^i (resp. Π_S^j) has a partner ID pid_U^i (resp:pid_S^j), a session ID sid_U^i (resp:sid_S^j), and a session key sk_U^i (resp:sk_S^j). pid_U^i (resp:pid_S^j) denotes the set of the identities that are involved in this instance. sid_U^i (resp:sid_S^j) denotes the flows that are sent and received by the instance Π_U^i (resp. Π_S^j). An instance Π_U^i (resp. Π_S^i) is said to be *accepted* if it holds a session key sk_U^i (resp:sk_S^j), a session identifier sid_U^i (resp:sid_S^j), and a partner identifier pid_U^i (resp:pid_S^j). Two instances $\Pi_{U_1}^i$ and $\Pi_{U_2}^j$ are considered *partnered* if and only if (1) both of them have accepted, (2) $pid_{U_1}^i = pid_{U_2}^j$, (3) $sid_{U_1}^i = sid_{U_2}^j$, (4) $sk_{U_1}^i = sk_{U_2}^j$.

5.1.2 Long-lived keys

Each client $U \in \mathscr{U}$ holds a password pw_U. Each server $S \in \mathscr{S}$ holds a vector $pw_S = \langle pw_U \rangle_{U \in \mathscr{U}}$ with an entry for each client.

5.1.3 Adversary model

The communication network is assumed to be fully controlled by an adversary $\mathscr{M}$, which schedules and mediates the sessions among all the parties. The adversary $\mathscr{M}$ is allowed to issue the following queries in any order:

Execute($\Pi^i_{U_1}, \Pi^j_{U_2}, \Pi^k_S$): This query models passive attacks in which the attacker eavesdrops on honest executions among the client instances $\Pi^i_{U_1}$ and $\Pi^j_{U_2}$ and trusted server instance Π^k_S. The output of this query consists of the messages that were exchanged during the honest execution of the protocol Π.
SendClient(Π^i_U, m): The adversary makes this query to intercept a message and then modify it, create a new one, or simply forward it to the client instance Π^i_U. The output of this query is the message that the client instance Π^i_U would generate upon receipt of message m. Additionally, the adversary is allowed to initiate the protocol by invoking SendClient($\Pi^i_{U_1}, (U_1, Start)$).
SendServer(Π^i_S, m): This query models an active attack against a server. The adversary makes this query to obtain the message that the server instance Π^i_S would generate on receipt of the message m.
Reveal(Π^i_U): This query models the known session key attack. The adversary makes this query to obtain the session key of the instance Π^i_U.
Corrupt(U): This query returns to the adversary the long-lived key pw_U for participant U.
Test(Π^i_U): Only one query of this form is allowed to be made by the adversary to a fresh oracle. To respond to this query, a random bit $b \in \{0, 1\}$ is selected. If $b = 1$, then the session key held by Π^i_U is returned. Otherwise, a uniformly chosen random value is returned.

5.1.4 Fresh oracle

An oracle Π^i_U is called fresh if and only if the following conditions hold: (1) Π^i_U has accepted; (2) Π^i_U or its partner (if exists) has not been asked a Reveal query after their acceptance; and (3) the client who has a partner instance with Π^i_U, has not been issued a Corrupt query.

5.1.5 3PAKE security

The security of a 3PAKE protocol Π is modeled by the game $Game^{3pake}(\Pi, \mathscr{M})$. When playing this game, $\mathscr{M}$ can make many queries mentioned earlier to Π^i_U and Π^j_S. If $\mathscr{M}$ asks a single test query, Test(Π^i_U), where Π^i_U has accepted and is fresh,

then $\mathscr{M}$ outputs a single bit b'. The aim of $\mathscr{M}$ is correctly guessing the bit b in the test session. More precisely, we define the advantage of $\mathscr{M}$ as follows:

$$Adv_{\Pi,D}^{3pake}(\mathscr{M}) = |2Pr[b' = b] - 1|. \tag{8}$$

The protocol Π is said to be 3PAKE-secure if $Adv_{\Pi,D}^{3pake}(\mathscr{M})$ only negligibly larger than $O(q_{send})/|D|$, where q_{send} is the number of the Send queries, and $|D|$ is the size of the password dictionary.

5.2 Computational assumption

We define the decisional Diffie–Hellman (DDH) assumption which we use in the security proof of our scheme.

5.2.1 Decisional Diffie–Hellman (DDH)

The DDH assumption can be precisely defined by two experiments, $Exp_{\alpha,p}^{cddh-real}(W)$ and $Exp_{\alpha,p}^{cddh-rand}(W)$. An adversary W is provided with $g^u \bmod p$, $g^v \bmod p$ and $g^{uv} \bmod p$ in the experiment $Exp_{\alpha,p}^{cddh-real}(W)$, and $g^u \bmod p$, $g^v \bmod p$ and $g^w \bmod p$ in the experiment $Exp_{\alpha,p}^{cddh-rand}(W)$, where u, v and w are drawn at random from $\mathbb{Z}_q^*$. Define the advantage of W in violating the DDH assumption, $Adv_{\alpha,p}^{cddh}(W)$, as follows:

$$\begin{aligned} Adv_{\alpha,p}^{ddh}(W) = \max\{|Pr[Exp_{\alpha,p}^{ddh-real}(W) = 1] \\ - Pr[Exp_{\alpha,p}^{cddh-rand}(W) = 1]|\}. \end{aligned}$$

5.3 Security proof

Theorem 1 *Let D be a uniformly distributed dictionary of size $|D|$. Let Π describes the 3PAKE protocol defined in Fig. 4. Suppose that DDH assumption holds, then,*

$$\begin{aligned} Adv_{\Pi,D}^{3pake}(\mathscr{M}) \leq \frac{q_h^2}{2^l} + \frac{(q_s + q_e)^2}{p^2} + 2q_e \cdot Adv^{DDH}(W) \\ + 2\max\left\{\frac{q_h}{p}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\} \end{aligned}$$

where q_s denotes the number of Send queries; q_e denotes the number of Execute queries; q_h denotes the number of hash queries to h.

Proof This proof consists of a sequence of hybrid games, starting at the real attack G_0 and ending up at game G_4 where the adversary has no advantage. For each game $G_i (0 \leq i \leq 4)$, we define $Succ_i$ as the event that $\mathscr{M}$ correctly guesses the bit b in the test session.

Game G_0. This game is the real protocol, in the random-oracle model. In this game, all the instances of A and B and the trusted server S are modeled as the real execution in the random oracle. By definition of event $Succ_i$, which means that the adversary correctly guesses the bit b involved in the Test-query, we have

$$Adv^{3pake}_{\Pi,D}(\mathscr{M}) = 2|\Pr[Succ_0] - \frac{1}{2}|. \tag{9}$$

Game G_1. This game is as the same as the game G_0 except that we simulate the hash oracle h as usual by maintaining hash list h_{List} with entries of the form (*Inp*, *Outp*). On hash query h(*Inp*) for which there exists a record (*Inp*, *Outp*) in the list h_{List}, return *Outp*. Otherwise, randomly choose $Outp \in \{0, 1\}^l$, send it to $\mathscr{M}$ and store the new tuple (*Inp*, *Outp*) into h_{List}. We also simulate all the instances, as the real players would do, for the Send-query and for the Execute, SendClient, SendServer, Reveal, Corrupt and Test queries. From the viewpoint of the adversary, we easily see that the game is perfectly indistinguishable from the real attack. Hence,

$$\Pr[Succ_1] = \Pr[Succ_0]. \tag{10}$$

Game G_2. In this game, we simulate all the oracles in game G_1, except we cancel the game in which some collisions appear on the partial transcripts (R_A, R_{S1}), (R_B, R_{S2}) or (R_A, R_B) and on hash values. According to the birthday paradox, the probability of collisions in output of h oracle is at most $q_h^2/2^{l+1}$, where q_h denotes the maximum number of queries to h. Similarly, the probability of collisions in the transcripts is at most $(q_s + q_e)^2/(2p^2)$, where q_s represents the number of queries to the SendClient and SendServer oracles and q_e represents the number of queries to the Execute oracle. So we have

$$|\Pr[Succ_2] - \Pr[Succ_1]| \leq \frac{q_h^2}{2^{l+1}} + \frac{(q_s + q_e)^2}{2p^2}. \tag{11}$$

Game G_3. In this game, we change the simulation of queries to the SendClient oracle. First, we randomly select a session executed by some honest clients A and B for partner instances Π^i_A and Π^j_B.

- When SendClient(Π^i_A, $(B, Start)$) is asked, we return $\{id_A, id_B\}$ to $\mathscr{M}$.
- When SendClient(Π^i_A, (R_{S1}, R_{S2})) is asked, we randomly select $u \in \mathbb{Z}^*_q$, and compute $R_A = g^u \bmod p$, $R_{AS} = (R_{S1} - g^{pw_A})^u \bmod p$ and $Auth_{AS}$ as the real protocol. Then, we return $\{id_A, R_A, Auth_{AS}, R_{S2}\}$ to $\mathscr{M}$.
- When SendClient(Π^j_B, $(id_A, R_A, Auth_{AS}, R_{S2})$) is asked, we randomly choose $v \in \mathbb{Z}^*_q$, compute $R_B = g^v \bmod p$, $K_B = R_A^v = g^{uv} \bmod p$, R_{BS}, $Auth_{BS}$ and $Auth_{BA}$ like the real protocol, and return $\{R_A, Auth_{AS}, R_B, Auth_{BS}, Auth_{BA}\}$ to $\mathscr{M}$.
- When SendClient(Π^i_A, $(R_B, Auth_{SA}, Auth_{BA}, Auth_{SB})$) is asked, we compute $K_A = R_B^u = g^{uv} \bmod p$, $Auth_{AB}$ and the session key SK like the real protocol, and return $\{Auth_{SB}, Auth_{AB}\}$ to $\mathscr{M}$.

So, it can be easily seen that this game is perfectly indistinguishable from the previous game G_2. Hence,

$$\Pr[Succ_3] = \Pr[Succ_2]. \tag{12}$$

Game G_4. In this game, we once again change the simulation of queries to the SendClient oracle for the selected session in game G_3. This time, we change the way we compute the values K_A and K_B so that they become independent of passwords and ephemeral keys. When SendClient $(\Pi_B^j, (id_A, R_A, Auth_{AS}, R_{S2}))$ and SendClient $(\Pi_A^i, (R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}))$ are asked, we set $K_A = K_B = T_w(\alpha)$, where w is selected from $\mathbb{Z}_p^*$ at random. The difference between the game G_4 and the game G_3 is as follows:

$$|\Pr[Succ_4] - \Pr[Succ_3]| \leq q_{exe} \cdot Adv_{\alpha,p}^{DDH}(W). \tag{13}$$

□

Proof By assuming a successful adversary $\mathscr{M}$ to distinguish G_3 and G_4, we construct a DDH solver W. The only difference between G_3 and G_4 is in the computation of K_A and K_B for the selected session. First time, W obtains a DDH tuple (g^u, g^v, Z). As G_3 and G_4, the solver W selects a matching session for Π_A^i and Π_B^j executed by the honest clients A and B. Then, when SendClient$(\Pi_A^i, (R_{S1}, R_{S2}))$ is asked, the solver W sets $R_A = g^u \bmod p$. Also, when SendClient$(\Pi_B^j, (id_A, R_A, Auth_{AS}, R_{S2}))$ and SendClient$(\Pi_A^i, (R_B, Auth_{SA}, Auth_{BA}, Auth_{SB}))$ are asked, the solver W sets $R_B = g^v \bmod p$ and $K_A = K_B = Z$. For all other queries, the solver W treats as G_3 and G_4.

The probability that the distinguisher $\mathscr{M}$ picks the selected session as the test session, (i.e., the adversary asks Test (Π_A^i) or Test(Π_B^j)) is $1/q_{exe}$. So, the solver W simulates all oracle queries without knowing u and v. From the obtained information, the distinguisher $\mathscr{M}$ can compute $(R_A = g^u \bmod p, R_B = g^v \bmod p)$ but cannot compute $K_A = K_B$. In the case of $Z = g^{uv} \bmod p$, this environment for the distinguisher is equivalent to G_3. In the case of $Z = g^w \bmod p$, this environment for the distinguisher is equivalent to G_4.

Finally, if the distinguisher decides that he interacted with G_3, then the solver W decides that $Z = g^{uv} \bmod p$. And, if the distinguisher decides that he interacted with G_4, then the solver W decides that $Z \neq g^{uv} \bmod p$. Hence,

$$|\Pr[Succ_4] - \Pr[Succ_3]| \leq q_{exe} \cdot Adv_{\alpha,p}^{DDH}(W).$$

□

In game G_4, Diffie–Hellman keys K_A and K_B are random and independent with passwords and ephemeral keys. So, there are three possible cases where the adversary distinguishes the real session key and the random key as follows:

Case 1. The adversary queries (g^w, id_A, id_B) to h. The probability that this event occurs is q_h/l.

Case 2. The adversary asks SendClient query except SendClient(Π_B^j, m) and successfully impersonates A to B. The adversary is allowed to reveal the static key pw_B of B but it is not allowed to reveal static key pw_A of A. Thus, in order to impersonate A, the adversary has to obtain some information of the password pw_A of A. The probability is $1/|D|$. If the adversary just makes an attempt at random to impersonate A by computing R_{AS} and succeeds, it will make the difference but the probability is less than $1/2^l$. Since there are at most q_s sessions of this kind, the probability that this event occurs is lower than $q_s/|D| + q_s/2^l$

Case 3. The adversary asks SendClient query except SendClient(Π_A^i, m) and successfully impersonates B to A. Similar to Case 1, the probability that this event occurs is lower than $q_s/|D| + q_s/2^l$.

As a conclusion,

$$\Pr[Succ_4] = \frac{1}{2} + \max\left\{\frac{q_H}{p}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\}. \tag{14}$$

Combining all the above equations, one gets the announced result as follows:

$$\begin{aligned}
Adv_{\Pi,D}^{3pake}(\mathscr{M}) &= 2|\Pr[Succ_0] - \frac{1}{2}| \\
&= 2\left|\Pr[Succ_0] - \Pr[Succ_4] + \max\left\{\frac{q_h}{l}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\}\right| \\
&\le 2\left(|\Pr[Succ_0] - \Pr[Succ_4]| + \max\left\{\frac{q_h}{l}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\}\right) \\
&\le 2\left(|\Pr[Succ_1] - \Pr[Succ_2]| + |\Pr[Succ_3] - \Pr[Succ_4]|\right. \\
&\quad \left. + \max\left\{\frac{q_h}{p}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\}\right) \\
&\le \frac{q_h^2}{2^l} + \frac{(q_s+q_e)^2}{p^2} + 2q_e \cdot Adv^{DDH}(W) \\
&\quad + 2\max\left\{\frac{q_h}{p}, \frac{q_s}{|D|} + \frac{q_s}{2^l}\right\}
\end{aligned}$$

6 Performance and security comparison

We evaluate the performance and security of our proposed protocol and make comparisons with some 3PAKE protocols which use neither symmetric key cryptography nor server public key. Table 2 shows the performance comparisons of our protocol and Chang et al.'s protocol [41] and Tso's protocol [42]. Two main operations are adopted in this analysis and they are defined as follows. H: the hash function operation; $_M$: the modular exponentiation operation. In the proposed protocol, we assume that the server computes and stores $g^{pw_U} \bmod p$ for each user U in the registration phase. Thus, when the server wants to use g^{pw_U}, it is not needed to compute a modular exponentiation. According to this assumption, the computational cost of our protocol is same as Tso's protocol and equal to 12M + 14H.

Table 2 Computation comparison

	User A	Server	User B	Total
Pu et al.'s [32]	5M + 4H	7M + 2H	5M + 4H	17M + 10H
Yang et al.'s [36]	4M + 3H	4M + 8H	4M + 3H	12M + 14H
Youn et al.'s [37]	6M + 6H	4M + 6H	6M + 6H	16M + 18H
Chang et al.'s [41]	3M + 5H	4M + 4H	3M + 5H	10M + 14H
Tso's [42]	4M + 5H	4M + 4H	4M + 5H	12M + 14H
Ours	4M + 5H	4M + 4H	4M + 5H	12M + 14H

Table 3 Security comparison

	Pu et al.'s scheme [32]	Yang et al.'s scheme [36]	Youn et al.'s scheme [37]	Chang et al.'s scheme [41]	Tso's scheme [42]	Our scheme
Reply attack	Secure	Secure	Secure	Secure	Secure	Secure
Impersonation attack	Insecure	Secure	Insecure	Insecure	Insecure	Secure
Guessing attacks	Secure	Secure	Secure	Insecure	Insecure	Secure
Denning–Sacco attack	Secure	Secure	Secure	Secure	Secure	Secure
Modification attack	Secure	Secure	Secure	Secure	Secure	Secure
Known-key attack	Secure	Secure	Secure	Secure	Secure	Secure
Forward secrecy	Yes	Yes	Yes	Yes	Yes	Yes
Provably secure	Yes	Yes	No	No	No	Yes

Table 3 lists the security comparisons among our proposed protocol and other related protocols. It demonstrates that our protocol has many excellent features and is more secure than other related protocols.

7 Conclusions

In this paper, we briefly reviewed Tso's 3PAKE protocol. We demonstrated that Tso's scheme is vulnerable to off-line password guessing attack. Additionally, we pointed out that Tso's scheme also suffers from impersonation attack. Therefore, we proposed an improved scheme to overcome the security weaknesses of the related schemes and proved its security in the random oracle model.

References

1. Farash MS, Bayat M, Attari MA (2011) Vulnerability of two multiple-key agreement protocols. Comput Electr Eng 37(2):199–204
2. Farash MS, Attari MA, Bayat M (2012) A certificateless multiple-key agreement protocol without one-way hash functions based on bilinear pairings. IACSIT Int J Eng Technol 4(3):321–325
3. Farash MS, Attari MA, Atani RE, Jami M (2013) A new efficient authenticated multiple-key exchange protocol from bilinear pairings. Comput Electr Eng 39(2):530–541

4. Farash MS, Attari MA (2013) Provably secure and efficient identity-based key agreement protocol for independent PKGs using ECC. ISC Int J Inf Secur 5(1):1–15
5. Farash MS, Attari MA (2014) A pairing-free ID-based key agreement protocol with different PKGs. Int J Netw Secur 16(2):143–148
6. Sakalauskas E, Katvickis A, Dosinas G (2010) Key agreement protocol over the ring of multivariate polynomials. Inf Technol Control 39(1):51–54
7. Lee CC, Lin TC, Hwang MS (2010) A key agreement scheme for satellite communications. Inf Technol Control 39(1):43–47
8. Hong JW, Yoon SY, Park DI, Choi MJ, Yoon EJ, Yoo KY (2011) An new efficient key agreement scheme for VSAT satellite communications based on elliptic curve cryptosystem. Inf Technol Control 40(3):252–259
9. Haiyan S, Qiaoyan W, Hua Z, Zhengping J (2013) A strongly secure pairing-free certificateless authenticated key agreement protocol for low-power devices. Inf Technol Control 42(2):105–112
10. Tseng YM, Yu CH, Wu TY (2012) Towards scalable key management for secure multicast communication. Inf Technol Control 41(2):173–182
11. Lo JW, Lin SC, Hwang MS (2010) A parallel password-authenticated key exchange protocol for wireless environments. Inf Technol Control 39(2):146–151
12. Chen BL, Kuo WC, Wuu LC (2012) A secure password-based remote user authentication scheme without smart cards. Inf Technol Control 41(1):53–59
13. Li CT (2011) Secure smart card based password authentication scheme with user anonymity. Inf Technol Control 40(2):157–162
14. Li CT, Lee CC (2011) A robust remote user authentication scheme using smart card. Inf Technol Control 40(3):236–245
15. Jiang Q, Ma J, Li G, Ma Z (2013) An improved password-based remote user authentication protocol without smart cards. Inf Technol Control 42(2):150–158
16. Bayat M, Farash MS, Movahed A (2010) A novel secure bilinear pairing based remote user authentication scheme with smart card. In: IEEE/IFIP International Conference on Embedded and ubiquitous computing (EUC), pp 578–582
17. Farash MS, Attari MA (2013) An enhanced authenticated key agreement for session initiation protocol. Inf Technol Control 42(4):333–342
18. Farash MS, Attari MA (2013) Cryptanalysis and improvement of a chaotic maps-based key agreement protocol using Chebyshev sequence membership testing. Nonlinear Dynam. doi:10.1007/s11071-013-1204-1
19. Lee CC, Chang YF (2008) On security of a practical three-party key exchange protocol with round efficiency. Inf Technol Control 37(4):333–335
20. Xie Q, Dong N, Tan X, Wong DS, Wang G (2013) Improvement of a three-party password-based key exchange protocol with formal verification. Inf Technol Control 42(3):231–237
21. Liu T, Pu Q, Zhao Y, Wu S (2013) ECC-based password-authenticated key exchange in the three-party setting. Arab J Sci Eng 68(8):2069–2077
22. Tu H, Shen H, He D, Chen J (2014) Security analysis and improvements of a three-party password-based key exchange protocol. Inf Technol Control 43(1):57–63
23. Zhao J, Gu D (2012) Provably secure three-party password-based authenticated key. Inf Sci 184(1):310–323
24. Yang JH, Cao TJ (2012) Provably secure three-party password authenticated key exchange protocol in the standard model. J Systems Softw 85(2):340–350
25. Xiong H, Chen Y, Guan Z, Chen Z (2013) Finding and fixing vulnerabilities in several three-party password authenticated key exchange protocols without server public keys. Inf Sci 235(1):329–340
26. Nam J, Paik J, Won D (2011) A security weakness in Abdalla et al'.s generic construction of a group key exchange protocol. Inf Sci 181(1):234–238
27. Zhao J, Gu D (2012) Provably secure three-party password-based authenticated key exchange protocol. Inf Sci 184(1):310–323
28. Lee TF, Hwang T (2010) Simple password-based three-party authenticated key exchange without server public keys. Inf Sci 180(9):1702–1714
29. Lou DC, Huang HF (2010) Efficient three-party password-based key exchange scheme. Int J Commun Systems 24(4):504–512
30. Wu S, Pu Q, Wang S, He D (2012) Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol. Inf Sci 215(1):83–96

31. Chien H (2011) Secure verifier-based three-party key exchange in the random oracle model. J Inf Sci Eng 27(4):1487–1501
32. Pu Q, Wang J, Wu S, Fu J (2013) Secure verifier-based three-party password-authenticated key exchange. Peer-to-peer networking and applications 6(1):15–25
33. Tallapally S (2012) Security enhancement on simple three party PAKE protocol. Inf Technol Control 41(1):15–22
34. Farash MS, Attari MA (2014) An enhanced and secure three-party password-based authenticated key exchange protocol without using server's public-keys and symmetric cryptosystems. Inf Technol Control 43(2):143–150
35. Farash MS, Attari MA (2014) An efficient and provably secure three-party password-based authenticated key exchange protocol based on Chebyshev chaotic maps. Nonlinear Dynam. doi:10.1007/s11071-014-1304-6
36. Yang H, Zhang Y, Zhou Y, Fu X, Liu H, Vasilakos AV (2014) Provably secure three-party authenticated key agreement protocol using smart cards. Comput Netw 58:29–38
37. Youn TY, Kang ES, Lee C (2013) Efficient three-party key exchange protocols with round efficiency. Telecommun Systems 52(2):1367–1376
38. Huang HF (2009) A simple three-party password-based key exchange protocol. Int J Commun Systems 22(7):857–862
39. Yoon EJ, Yoo KY (2011) Cryptanalysis of a simple three-party password-based key exchange protocol. Int J Commun Systems 24(4):532–542
40. Wu S, Chen K, Zhu Y (2013) Enhancements of a three-party password-based authenticated key exchange protocol. Int Arab J Inf Technol (IAJIT) 10(3):215
41. Chang TY, Hwang MS, Yang WP (2011) A communication-efficient three-party password authenticated key exchange protocol. Inf Sci 181(1):217–226
42. Tso R (2013) Security analysis and improvements of a communication-efficient three-party password authenticated key exchange protocol. J Supercomput. doi:10.1007/s11227-013-0917-8
43. Abdalla M, Pointcheval D (2005) Interactive Diffie–Hellman assumptions with applications to password-based authentication. In: Proceedings of FC'05, LNCS 3570, pp 341–356