Journal of Ambient Intelligence and Humanized Computing (2019) 10:3133–3142
https://doi.org/10.1007/s12652-018-1029-3

ORIGINAL RESEARCH

Attacks and solutions on a three-party password-based authenticated key exchange protocol for wireless communications

Chien-Ming Chen[1] · King-Hang Wang[2] · Kuo-Hui Yeh[3] · Bin Xiang[4] · Tsu-Yang Wu[5]

Received: 5 July 2018 / Accepted: 1 September 2018 / Published online: 10 September 2018

Abstract
A secure authenticated key exchange protocol is an essential key to bootstrap a secure wireless communication. Various research have been conducted to study the efficiency and security of these authenticated key exchange protocol. A recent work by Lu et al. addresses the needs of a three parties secure communication by presenting a new protocol that claimed to be resistance against various attacks. However we found that their protocol is still vulnerable against an off-line password guessing attack. In this attack, an adversary can obtain the password of an user without any direct interactions with the server. To surmount such problem, we propose a new three-party password-based authenticated key exchange protocol. The security of our protocol are proved by the automatic cryptographic protocol tool *proverif*. The protocol presented is also more secure and efficient comparing with other similar protocols in the literature.

Keywords Authenticated key exchange · Password-based · Wireless communication · Proverif

1 Introduction

Authenticated key exchange protocols are essential for bootstrap a secure wireless communications over the unprotected public channel. The protocol allows protocol participants to confirm the identities of their communication partners and agree a common session key to encrypt and sign messages in their communication session. Password-based authenticated key exchange protocols authenticate their partner using human-memoriable passwords. The security of these protocols rely on whether users are able to select high entropy passwords so that the adversary will not be able to guess the password on a random-password-guessing protocol interaction.

Three-party password-based authenticated key exchange (3PAKE) protocols are a family of protocols that set up secure communication for three parties using passwords (Chen et al. 2012; Xiong et al. 2017b; Chang et al. 2005; Chang and Chang 2004; Lee et al. 2005; Zhu 2017; Chang et al. 2011; Lin and Fu 2013; Xiong et al. 2017a; Zhu and Zhang 2017).

In the three-party setting, a trusted third party (TTP) is introduced to keep the passwords for the users and help the other two user parties in authentication and key exchange. It is required that a secure 3PAKE protocol (Li et al. 2018; Chen et al. 2018b; Chen et al. 2018a; Chen et al. 2016; Shen et al. 2015) to meet the following requirements for practical use:

1. Secure against on-line password guessing attacks: A 3PAKE protocol authenticates a user using its password. It is unavoidable that an attacker attempts to perform trial-and-error to guess the password by running the protocol many times. We say a protocol is secure against on-line password guessing attacks if the protocol

✉ Tsu-Yang Wu
wutsuyang@gmail.com

Chien-Ming Chen
chienming.taiwan@gmail.com

[1] College of Computer Science and Engineering, Shandong University of Science and Technology, Shandong, China

[2] Hong Kong University of Science and Technology, Hong Kong, China

[3] Department of Information Management, National Dong Hwa University, Taiwan, China

[4] Harbin Institute of Technology (Shenzhen), Shenzhen, China

[5] Fujian Provincial Key Laboratory of Big Data Mining and Applications, National Demonstration Center for Experimental Electronic Information and Electrical Technology Education, Fujian University of Technology, Fujian, China

can detect the failure login attempt of a particular user account. Under such circumstance other protocol participants (most often the TTP) can suspect the suspicious user account.
2. Secure against off-line password guessing attacks: A 3PAKE protocol is said to be secure against off-line password guessing attacks if no attacker can deduce a user password from a finite times of communication records.
3. Secure against impersonation attacks: A 3PAKE protocol is said to be secure against impersonation attacks if no attacker can impersonate another party without its password.
4. Secure against replay attacks: A 3PAKE protocol is said to be secure against replay attacks if no attacker can be authenticated by replaying some past communication records.
5. Secure against Man-In-The-Middle attacks: A 3PAKE protocol is said to be secure against Man-In-The-Middle (MITM) attacks if no attacker can allow some parties mutually authenticate each other while fail to agree the same session key.
6. Able to provide mutual authentication: A 3PAKE protocol is said to be able to provide mutual authentication if *all* protocol participants are convinced they are communicating with the trusted partners by the end of the protocol.
7. Secure against Known-session key attacks: A 3PAKE protocol is said to be secure against known-session key attacks if no any attacker who is given the session key of a past session can be able to be authenticated in the future, nor can be able to deduce the users password.
8. Able to provide session key perfect forward secrecy: A 3PAKE protocol is said to be able to provide session key perfect forward secrecy if no any attacker who is given the password would be able to recover the session key agreed in the past.

In 2012, Tallapally proposed a three-party password-based protocol (Farash and Attari 2014) for wireless communication and claimed that their protocol could meet various secure requirements. However, Farash and Attari pointed out that Tallapally's 3PAKE protocol is vulnerable to undetectable on-line password guessing attacks and off-line password guessing attacks, and then proposed their improved scheme (Tallapally 2012). Unfortunately, Lu et al. (2015) observed that Farash and Attari protocol cannot resist off-line password guessing attacks as claimed. In order to defend such attacks, Lu et al. also proposed a modified 3PAKE protocol. In this paper, we find that Lu et al.'s protocol still cannot resist off-line password guessing attacks. In this paper we proposed a new 3PAKE for wireless communication. The proposed scheme is then validated by *Proverif*. Through the performance and security analysis, we show that the proposed scheme is more secure with similar efficiency.

The remainder of the paper is organized as follows. Section 2 briefly review the protocol of Lu et al.'s. We analyze their protocol and show the protocol cannot resist offline password guessing attack in Sect. 3. In Sect. 4, we propose our new three-party password-based protcol. Then we analyze the security and validity of our protocol with the automatic cryptographic protocol tool proverif in Sect. 5. Finally, our conclusion is present in Sect. 6.

2 Review of Lu et al.'s protocol

In this section, we review Lu et al.'s protocol briefly. The protocol involves three parties, the user A, B, and the server S. Each user shares the identity and the hashed password $h(pw)$ with S. The server initializes the system with a large prime p and a generator g. Following steps explain Lu et al.'s protocol. For simplicity, modulo p operations are hidden from computations. Note that notation used in this paper are summarized in Table 1.

1. A selects two nonces x, $r_a \in Z_p^*$ and computes $H_a = h(pw_a) \oplus r_a$, $V_a = h(pw_a||r_a||A)$, and $R_a = g_x \oplus V_a$. Then, A sends $m_1 = \{H_a, V_a, R_a, A\}$ to S. In the same time, B performs similarly as A. B first selects two nonces y, $r_b \in Z_p^*$ and computes similarly. Then, B sends $m_1' = \{H_b, V_b, R_b, B\}$ to S.
2. After S receives the messages, S retrieves the hashed passwords to recover r_a and r_b. Then, S checks the value V_a and the value V_b using corresponding values. If they are verified successfully, S computes $R_a' = R_a \oplus h(A||pw_a||r_a)$, and $R_b' = R_b \oplus h(A||pw_b||r_b)$. Then, S selects a nonce z and computes $N_s = g^z$, $K_{sa} = (R_a')^z$, $K_{sb} = (R_b')^z$, $T_a = h(pw_a||r_a||K_{sa}||A)$, $T_b = h(pw_b||r_b||K_{sb}||B)$. Next, S sends $m_2 = \{T_a, N_s, B\}$ to A and $m_2' = \{T_b, N_s, A\}$ to B.
3. After A receives the message, A computes $K_{as} = (N_s)^x$ and checks the value T_a using corresponding values. If it is verified successfully, A computes $W_a = h(A||B||K_{as}||pw_a||r_a)$ and sends $m_3 = \{A, W_a\}$ to S. B

Table 1 Notations used in this paper

Notations	Descriptions
pw_i	Password of user i
ID_i	Identification of user i
$h()$	Secure
$\oplus$	Exclusive-or
$\|\|$	Message concatenation
g	Generator

processes the received message in a similar way, and sends $m'_3 = \{B, W_b\}$ to S if the verification is correct.

4. After S receives the messages, S verifies W_a and W_b. If both return true, S computes $U_a = K_{sb} \oplus h(A||B||K_{sa}||pw_a||r_a)$ and $U_b = K_{sa} \oplus h(A||B||K_{sb}||pw_b||r_b)$, and sends $m_4 = \{U_a\}$ to A and $m'_4 = \{U_b\}$ to B.
5. After the messages arrive, A computes $K'_{sb} = U_a \oplus h(A||B||K_{sa}||pw_a||r_a)$, $sk = (K'_{sb})^x$, and $A_a = h(sk||A||B)$, while B computes $K'_{sa} = U_b \oplus h(A||B||K_{sb}||pw_b||r_b)$, $sk = (K'_{sa})^y$, and $A_b = h(sk||A||B)$. Then, they exchange the message $m_5 = \{A_a, A\}$ and $m'_5 = \{Ab, B\}$.
6. After the messages arrive, A and B checks the validity of A_a and A_b respectively.

Finally, if they both hold true, A and B establish a common session key $sk = g_{xyz}$.

3 Cryptanalysis of Lu et al.'s protocol

In this section, we show that Lu et al.'s protocol is vulnerable to an off-line password guessing attack. Assume that an adversary E overhears a protocol run among user A, user B, and server S, and captures the messages sent by them. Then, E performs the following steps to launch an off-line password guessing attack.

1. Extract H_a, V_a, R_a, A from m_1.
2. For each possible password pw', repeat:
   - Compute $r'_a = H_a \oplus h(pw')$, $V'_a = h(pw'||r'_a||A)$.
   - If $V_a = V'_a$, return pw'.

The second step indicates a loop to check whether the guessed password is correct or not. Since in each loop, it takes two hash operations and one exclusive-or operation, E can obtain the correct password in a time proportional to $|D| \times t_H$, where $|D|$ represents the size of the password set, and t_H is the time cost of a hash operation. The attack aims at the user A's password, but it can also be launched to obtain the user B's password. The adversary E simply eavesdrops the message m'_1, computes r'_b, V'_b, and checks the equation $V_b = V'_b$. The time cost is the same as the attack on A's password.

4 The proposed 3PAKE protocol

This section aims to propose an enhanced a new 3PAKE protocol to overcome the above mentioned problems with the Lu et al.'s protocol. The same as Lu et al.'s protocol, each user shares the identity and the hashed password $h(pw)$ with S. The server initializes the system with a large prime p and a generator g. For convenience, modulo p operations are also hidden from computations.

The protocol is showed in Fig. 1 and more details are provided as follows:

1. A randomly chooses one number a, and computes $R_A = g_a$, $K_{AS} = (g^x)^a$, $H_{AS} = h_1(ID_A||ID_B||R_A||K_{AS}||h_1(PW_A))$. Then, it sends $\{ID_A, ID_B, R_A, H_{AS}\}$ to server S. B also selects one random number b, computes $R_B = g_b$, $K_{BS} = (g^x)^b$, $H_{BS} = h_1(ID_A||ID_B||R_B||K_{BS}||h_1(PW_B))$ and sends $\{ID_A, ID_B, R_B, H_{BS}\}$ to server S.
2. Upon receiving the messages form A and B. S computes $K_{AS} = (R_A)^x$, $H'_{AS} = h_1(ID_A||ID_B||R_A||K_{AS}||h_1(PW_A))$, $K'_{BS} = (R_B)^x$, $H_{BS} = h_1(ID_A||ID_B||R_A||K_{BS}||h_1(PW_B))$, and then check $H'_{AS} \stackrel{?}{=} H_{AS}$, $H'_{BS} \stackrel{?}{=} H_{BS}$, which holds if A and B are legal. After that, S selects one random number c, computes $K'_{BS} = (R^B)^c$, $K'_{AS} = (R^A)^c$, $H_{SA} = h_1(K'_{BS} \oplus K_{AS}||K_{AS})$, $H_{SB} = h_1(K'_{AS} \oplus K_{BS}||K_{BS})$. Lastly, it sends $\{K'_{BS} \oplus K_{AS}, H_{SA}\}$ to A and $\{K'_{AS} oplus K_{BS}, H_{SB}\}$ to B.
3. After A receives the message, A computes $K'_{BS} = (K'_{BS} \oplus K_{AS}) \oplus K_{AS}$, and verify $H_{SA} \stackrel{?}{=} h_1(K'_{BS} \oplus K_{AS}||K_{AS})$. If verification does not hold, it terminates the session. Otherwise, computes $K = (K'_{BS})^a$, $SK = h_1(K)$, $M_{AB} = h_1(1||ID_A||ID_B||K)$, $M_{AS} = h_1(1||K_{AS}||M_{AB})$. Then, it sends $\{M_{AB}, M_{AS}\}$ to S.
4. After B receives the message, B computes $K'_{AS} = (K'_{AS} \oplus K_{BS}) \oplus K_{BS}$, and verifies $H_{SB} \stackrel{?}{=} h_1(K'_{AS} oplus K_{BS}||K_{BS})$. If verification does not hold, it terminates the session. Otherwise, B computes $K = (K'_{AS})^b$, $SK = h_1(K)$, $M_{BA} = h_1(1||ID_A||ID_B||K)$, $M_{BS} = h_1(1||K_{BS}||M_{BA})$. Then, it sends $\{M_{BA}, M_{BS}\}$ to S.
5. After receiving the messages, S verifies M_{AS} and M_{BS}. If both return true, S sends M_{BA} to A and M_{AB} to B, respectively.
6. After the messages arrive, A and B checks $M_{BA} \stackrel{?}{=} M_{AB}$ and $M_{AB} \stackrel{?}{=} M_{BA}$ respectively.

Finally, if they both hold true, A and B establish a common session key $sk = h_1(K) = h_1(g^{abc})$.

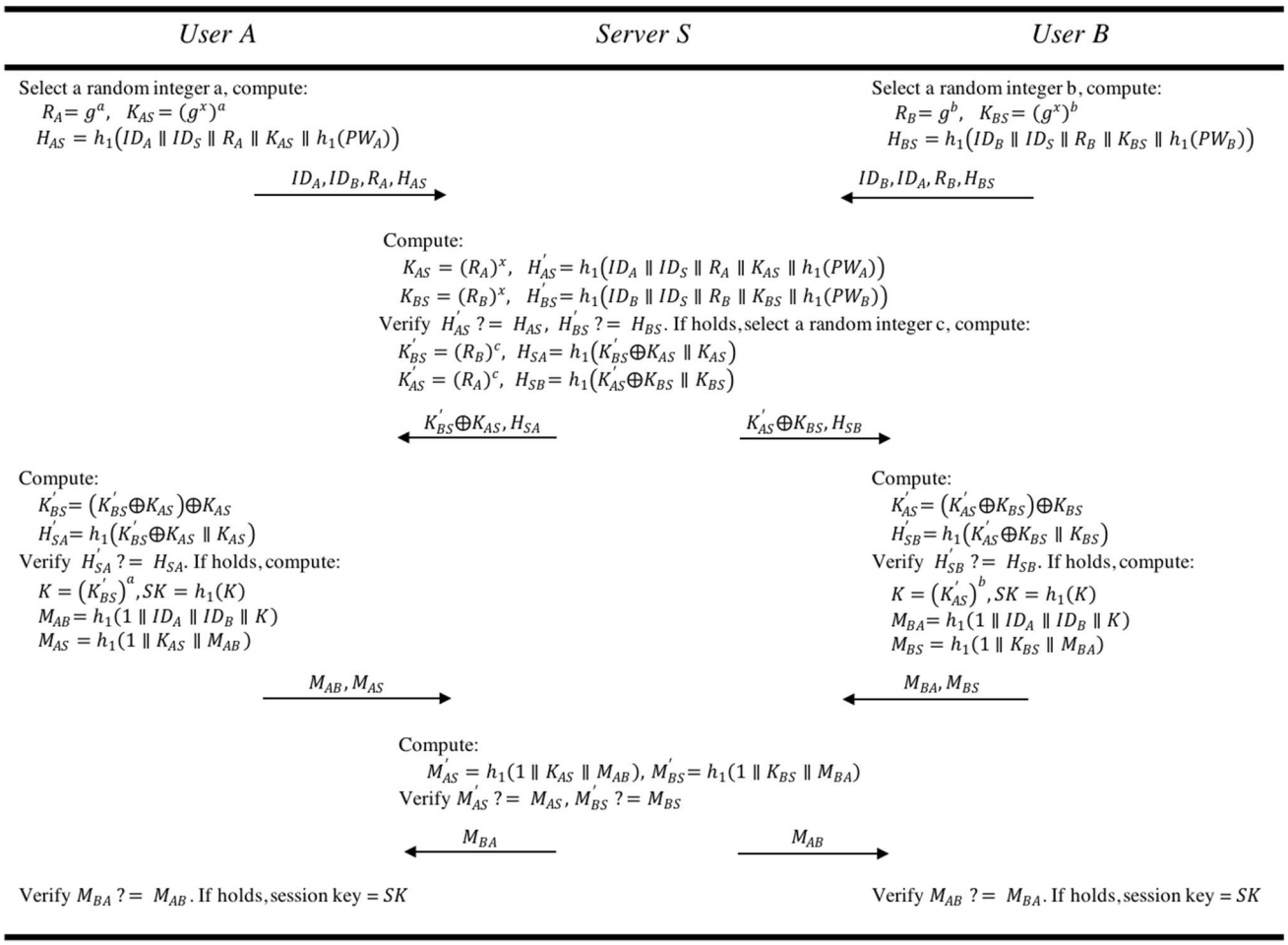

Fig. 1 The proposed 3PAKE protocol

5 Security analysis

This section presents security features of the proposed protocol, which reveals our protocol is safe from several kinds of attacks.

5.1 On-line password guessing attack

If an adversary E attempts to guess A's password, E guesses a password PW'_A and selects a random integer α', computes $R'_A = g^{\alpha'}$, $K'_{AS} = (g^x)^{\alpha'}$ and $H'_{AS} = h_1(ID_A||ID_s||R'_A||K'_{AS}||h_1(PW'_A))$. Then E sends the message $\{ID_A, ID_B, R'_A, H'_{AS}\}$ to the trust server S. However, S can detect this attack since it checks the validity of the value H'_{AS} when S receives the message. Also, if E attempts to guess B's password, S can detect it in the say way. Therefore, the proposed protocol can resist an on-line password guessing attack.

5.2 Off-line password guessing attack

Without loss of generality, we assume that an adversary E intercepts the message $\{ID_A, ID_B, R_A, H_{AS}, K'_{BS} \oplus K_{AS}, H_{SA}, M_{AB}, M_{AS}, M_{BA}\}$ and attempts to launch an off-line password guessing attack to obtain A's password (B is the same case). As we know, only the parameter H_{AS} is related to A's password. However, E cannot check whether his guessed password is right or not since without the parameter R_A, he cannot reconstruct H'_{AS} and compare it with H_{AS}. Therefore, the proposed protocol can resist an off-line password guessing attack.

5.3 Impersonation attack

Assume that an adversary E attempts to A with the intercepted message $\{ID_A, ID_B, R_A, H_{AS}, K'_{BS} \oplus K_{AS}, H_{SA}, M_{AB}, M_{AS}, M_{BA}\}$. However, without the correct A's password, E will be detected when S checks the value of H_{AS} which contains the correct value

```
(******** channel ******)
free ch:channel. (* public channel *)

(****** shared keys ******)
free SKa:bitstring [private].
free SKb:bitstring [private].

(****** constants ******)
free x:bitstring [private]. (* the server's secret key *)
const g:bitstring. (* the generator *)

(****** functions & reductions & equations ******)
fun h(bitstring):bitstring. (* hash function*)

fun exp(bitstring,bitstring):bitstring. (* exp mudolu operation *)
equation forall a:bitstring, b:bitstring;
exp(exp(g,a),b)=exp(exp(g,b),a).

fun conOne(bitstring):bitstring. (* 1 || bitstring *)
fun senc(bitstring,bitstring):bitstring. (* symmetric encryption *)
reduc forall m:bitstring, key:bitstring; sdec(senc(m,key),key)=m.

fun con(bitstring,bitstring):bitstring. (* concatenation operation *)

fun xor(bitstring,bitstring):bitstring. (* XOR operation*)
equation forall m:bitstring, n:bitstring;xor(xor(m,n),n)=m.

(******  password table ******)
table pwdTable(bitstring,bitstring).

(****** event ******)
event BeginUserA.
event EndUserA.
event BeginUserB.
event EndUserB
```

Fig. 2 Functions and rules

```
(****** queries ******)
query attacker(SKa).
query attacker(SKb).
query inj-event(EndUserA) ==> inj-event(BeginUserA).
query inj-event(EndUserB) ==> inj-event(BeginUserB).
```

Fig. 3 Four queries: The first two queries check if an adversary is unable to obtain the shared session key. The last two queries check if the identity cannot be forged

Fig. 4 The process of user *A*

```
(* ------------------------------ user A's process ------------------------------ *)
let
ProcessUserA(idA:bitstring,idB:bitstring,idS:bitstring,pwA:bitstring,
Ppub:bitstring) =

   new a:bitstring;
   let Ra = exp(g,a) in
   let Kas = exp(Ppub,a) in
   let Has = h(con(idA,con(idS,con(Ra,con(Kas,h(pwA)))))) in
   out(ch,(idA,idB,Ra,Has));

   in(ch,(=idA,xres:bitstring,Hsa:bitstring));
   let Kbs' = xor(xres,Kas) in
   let Hsa' = h(con(xor(Kbs',Kas),Kas)) in
   if Hsa' = Hsa then let K = exp(Kbs',a) in
   let SK = h(K) in
   let Mab = h(conOne(con(idA,con(idB,K)))) in
   let Mas = h(conOne(con(Kas,Mab))) in
   out(ch,(idA,Mab,Mas));

   in(ch,(=idA,Mba:bitstring));
   let Mba' = h(conOne(con(idA,con(idB,K)))) in
   if Mba' = Mba then
   out(ch,senc(SKa,SK));
   event EndUserA.
```

of *A*. Therefore, *E* cannot impersonate user *A*. Similarly, *E* cannot impersonate user *B* as well.

When *E* attempts to impersonate *S*, he also cannot succeed since without *S*'s secret key *x*, he cannot construct K_{AS}. Therefore, *E* cannot impersonate *S*.

5.4 Replay attack

Assume that *E* has obtained all the messages transmitted among *A*, *B* and *S*. However, he cannot perform this attack by replay *A*'s or *B*'s message since these parameters are protected by random integer and will be detected immediately when received by *S*. Similarly, when *E* replays the *S*'s messages, *A* or *B* will be detect with the aid of the selected random integers. Therefore, the proposed protocol can resist a replay attack.

5.5 Man-in-the-middle attack

From the above analysis, we can know that *E* cannot succeed by impersonating and replaying. On the other hand, because the transmitted parameters H_{AS}, H_{BS} contain the user's identities, thus man-in-the-middle attack cannot succeed. The proposed protocol can resist a man-in-the-middle attack.

5.6 Mutual authentication

In the proposed protocol, *S* authenticates *A* by computing H_{AS} and M_{AS} and authenticates *B* by computing H_{BS} and M_{BS}. Meanwhile, *A* and *B* authenticate *S* by computing H_{SA} and H_{SB}, respectively. Therefore, the proposed protocol can achieve mutual authentication.

5.7 Known-key security

In the proposed protocol, the session key $SK = h_1(g^{abc})$ is depend on the random integers *a*, *b*, and *c*, which are diffident in all sessions. Thus, an adversary *E* cannot compute the previous and the future session keys even he has known one session key.

Fig. 5 The process of user *B*

```
(* ------------------------------ user B's process ------------------------------ *)
let
ProcessUserB(idA:bitstring,idB:bitstring,idS:bitstring,pwB:bitstring,
Ppub:bitstring) =

   new b:bitstring;
   let Rb = exp(g,b) in
   let Kbs = exp(Ppub,b) in
   let Hbs = h(con(idB,con(idS,con(Rb,con(Kbs,h(pwB)))))) in
   out(ch,(idB,idA,Rb,Hbs));

   in(ch,(=idB,xres:bitstring,Hsb:bitstring));
   let Kas' = xor(xres,Kbs) in
   let Hsb' = h(con(xor(Kas',Kbs),Kbs)) in
   if Hsb‘ = Hsb then let K = exp(Kas',b) in
   let SK = h(K) in
   let Mba = h(conOne(con(idA,con(idB,K)))) in
   let Mbs = h(conOne(con(Kbs,Mba))) in
   out(ch,(idB,Mba,Mbs));

   in(ch,(=idB,Mab:bitstring));
   let Mab' = h(conOne(con(idA,con(idB,K)))) in
   if Mab' = Mab then
   out(ch,senc(SKb,SK));
   event EndUserB.
```

6 Simulation verification with proverif

Proverif is an automatic cryptographic protocol verifier, which is widely used to specify and analyze the security of authenticated key agreement protocols (Jiang et al. 2017; Chaudhry et al. 2017; Wu et al. 2017b; Abbasinezhad-Mood and Nikooghadam 2018a, b; Jiang et al. 2018; Wu et al. 2017a). In this section, we utilize Proverif to further analyze the security and validity of the proposed protocol. The whole simulation contains the following procedures:

- First, a public channel *ch* is defined for the communications. *SKa* and *SKb* are the session keys generated by the users, and *x* is the secret key of *S*. Then comes the functions and rules (Fig. 2).
- The purpose of the verification is to verify the following four queries. The former two are about whether an adversary can have the shared session keys or not, and the rest ones are about the correctness of the authentication process (Fig. 3).
- The process of user *A*. (Fig. 4).
- The process of user *B* (Fig. 5).
- The process of user *S* (Fig. 6).
- The main execution. (Fig. 7).
- The result of the proposed protocol. From the results, we can conclude that an adversary has no ways to obtain users' session key (Fig. 8).

7 Performance evaluation

This section describes performance evaluation of the proposed protocol along with other related protocols (Lu et al. 2015; Farash and Attari 2014; Tallapally 2012), in security properties, communication cost and estimated time. We focus on the security against on-line password guessing attack, off-line password guessing attack, impersonation attack, replay attack, man-in-the-middle attack, mutual authentication, known-key security and the session key perfect forward secrecy. From Table 2, we can observe that only the proposed protocol fulfills all the security properties.

To compare the communication cost, we assumed that the output of a user identity is 32-bit, a random integer is 128-bit, a hash function(SHA-256) is 256-bit, a trapdoor is

Fig. 6 The process of S

```
(* ------------------------------ server S's process ------------------------------ *)
let ProcessS(idA:bitstring,idB:bitstring,idS:bitstring) =

  (* get the messages from A and B*)
  in(ch,(=idA,idx:bitstring,Ra:bitstring,Has:bitstring));
  in(ch,(=idB,idxx:bitstring,Rb:bitstring,Hbs:bitstring));
  (* check A and B *)
  get pwdTable(=idA,hpwA) in
  let Kas = exp(Ra,x) in
  let Has' = h(con(idA,con(idS,con(Ra,con(Kas,hpwA))))) in
  get pwdTable(=idB,hpwB) in
  let Kbs = exp(Rb,x) in
  let Hbs' = h(con(idB,con(idS,con(Rb,con(Kbs,hpwB))))) in
  if Has' = Has then event BeginUserA;
  if Hbs' = Hbs then event BeginUserB;
  new c:bitstring;
  let Kbs' = exp(Rb,c) in
  let Kas' = exp(Ra,c) in
  let Hsa = h(con(xor(Kbs',Kas),Kas)) in
  let Hsb = con(xor(Kas',Kbs),Kbs) in
  out(ch,(idA,xor(Kbs',Kas),Hsa));
  out(ch,(idB,xor(Kas',Kbs),Hsb));

  in(ch,(=idA,Mab:bitstring,Mas:bitstring));
  let Mas' = h(conOne(con(Kas,Mab))) in
  if Mas' = Mas then out(ch,(idB,Mas));

  in(ch,(=idB,Mba:bitstring,Mbs:bitstring));
  let Mbs' = h(conOne(con(Kbs,Mba))) in
  if Mbs' = Mbs then out(ch,(idA,Mba)).
```

1024-bit, and modular exponential operation (DH-1024) is 1024-bit. From Table 3, we see that the proposed protocol costs less in bits then Farash et al.'s protocol (Farash and Attari 2014) and Lu et al.'s protocol (Lu et al. 2015), but a little bit more than Tallapally's protocol(Tallapally 2012). Thats because Tallapally's protocol does not design parameters for mutual authentication.

We then use a personal smart device (iPhone 6s with ARM(armv8-a) CPU, 2GB RAM and iOS10.1.1 operation system) to execute the related operations such as SHA-256 and exponential modulo operation (based on GMP library). We execute each for 10000 times and compute the average running times. A hash function needs 0.014 ms and an exponential modulo operation needs 0.471 ms. With them, we compute the estimated time of the protocols in Table 4. With Table 4, we can see that the proposed protocol cost more time. Indeed, Lu et al.'s scheme (Lu et al. 2015) has a better performance (2.994 ms) than our protocol (4.962 ms). However, their scheme cannot withstand online/off-line password guessing attack. Also, their scheme does not provided mutual authentication. Our protocol provides these security requirements. In other aspect, even 1.968ms (4.962–2.994) is too short for human beings to sense. In return, the proposed protocol provides better security and communication cost.

8 Conclusion

In this paper, we demonstrate a weakness in Lu et al.s 3PAKE protocol. We find that their protocol is vulnerable to off-line password guessing attack. In order to erase the weakness of Lu et al.'s protocol and enhance

Fig. 7 The main execution

```
(* ---------------------------------- Main ---------------------------------- *)
process

   (* initialization *)
   new idA:bitstring; (* the id,password of user A *)
   new pwA:bitstring;
   insert pwdTable(idA,h(pwA));

   new idB:bitstring;  (* the id,password of user B *)
   new pwB:bitstring;
   insert pwdTable(idB,h(pwB));
   new idS:bitstring;  (* the id of the server S *)

   let Ppub = exp(g,x) in
   (!ProcessUserA(idA,idB,idS,pwA,Ppub)
| !ProcessUserB(idA,idB,idS,pwB,Ppub) | !ProcessS(idA,idB,idS))
```

Fig. 8 The result of the proposed protocol. All four queries return 'true' which means the attacker is unable to obtain the session key and also unable to forge the user's identity in the protocol

```
(* ---------------------------------- results ---------------------------------- *)
-- Query inj-event(EndUserB) ==> inj-event(BeginUserB)
RESULT inj-event(EndUserB) ==> inj-event(BeginUserB) is true.

-- Query inj-event(EndUserA) ==> inj-event(BeginUserA)
RESULT inj-event(EndUserA) ==> inj-event(BeginUserA) is true.

-- Query not attacker(SKb[])
RESULT not attacker(SKb[]) is true.

-- Query not attacker(SKa[])
RESULT not attacker(SKa[]) is true.
```

Table 2 Comparison of security properties

Protocols	C1	C2	C3	C4	C5	C6	C7	C8
Lu et al. (2015)	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
Farash et al. (2014)	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Tallapally. (2012)	No	No	Yes	Yes	Yes	No	Yes	Yes
Ours protocol	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

C1: withstand on-line password guessing attack; C2: withstand off-line password guessing attack; C3: withstand impersonateion attack; C4: withstand replay attack; C5: withstand man-in-the middle attack; C6: mutual authentication; C7: known-key security; C8: session key perfect forward secrecy

the security, we later proposed a new 3PAKE protocol. The security of the proposed protocol is proved by the cryptographic protocol tool proverif. We also compare our protocol against three other similar works and found that our communication cost is the lowest among the rest while maintaining a similar computation cost.

Table 3 Comparison of communication cost (in bits)

Protocols	Communication Cost	Bits
Lu et al. (2015)	$8L_{ID}+10L_h+6L_{DH}$	8960
Farash et al. (2014)	$4L_{ID}+6L_h+6L_{DH}$	7808
Tallapally (2012)	$2L_{ID}+2L_h+4L_{DH}+2L_{F(.)}$	6720
Ours protocol	$4L_{ID}+10L_h+4L_{DH}$	6484

L_{ID}: the length of a user identity; L_h : the length of the output of hash function; L_{DH} : the length of the output of modular exponential operation; $L_{F(.)}$: the length of the output of a trapdoor

Table 4 Comparison of estimated time

Protocol	Estimated Time	Time (in milliseconds)
Lu et al. (2015)	$22T_h+9T_e$	4.547
Farash et al. (2014)	$18T_h+9T_e$	4.491
Tallapally (2012)	$12T_h+6T_e$	2.994
Ours protocol	$18T_h+10T_e$	4.962

T_h: the time for executing a one-way hash function; T_e: the time for executing exponential modulo operation

Acknowledgements The work of Tsu-Yang Wu was supported in part by the Science and Technology Development Center, Ministry of Education, China under Grant no. 2017A13025 and the Natural Science Foundation of Fujian Province under Grant no. 2018J01636.

References

Abbasinezhad-Mood D, Nikooghadam M (2018a) Design and hardware implementation of a security-enhanced elliptic curve cryptography based lightweight authentication scheme for smart grid communications. Future Gener Comput Syst 84:47–57

Abbasinezhad-Mood D, Nikooghadam M (2018b) Efficient anonymous password-authenticated key exchange protocol to read isolated smart meters by utilization of extended chebyshev chaotic maps. IEEE Trans Ind Inform. https://doi.org/10.1109/TII.2018.2806974

Chang CC, Chang YF (2004) A novel three-party encrypted key exchange protocol. Comput Stand Interfaces 26(5):471–476

Chang TY, Yang WP, Hwang MS (2005) Simple authenticated key agreement and protected password change protocol. Comput Math Appl 49(5):703–714

Chang TY, Hwang MS, Yang WP (2011) A communication-efficient three-party password authenticated key exchange protocol. Inform Sci 181(1):217–226

Chaudhry SA, Naqvi H, Sher M, Farash MS, Hassan MU (2017) An improved and provably secure privacy preserving authentication protocol for SIP. Peer Peer Netw Appl 10(1):1–15

Chen BL, Kuo WC, Wuu LC (2012) A secure password-based remote user authentication scheme without smart cards. Inform Technol Control 41(1):53–59

Chen CM, Xu L, Wu TY, Li CR (2016) On the security of a chaotic maps-based three-party authenticated key agreement protocol. J Netw Intell 1(2):61–66

Chen CM, Fang W, Liu S, Wu TY, Pan JS, Wang KH (2018a) Improvement on a chaotic map-based mutual anonymous authentication protocol. J Inform Sci Eng 34(2):371–390

Chen CM, Xu L, Wang KH, Liu S, Wu TY (2018b) Cryptanalysis and improvements on three-party-authenticated key agreement protocols based on chaotic maps. J Int Technol 19(3):679–687

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. Inform Technol Control 43(2):143–150

Jiang Q, Zeadally S, Ma J, He D (2017) Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks. IEEE Access 5:3376–3392

Jiang Q, Chen Z, Li B, Shen J, Yang L, Ma J (2018) Security analysis and improvement of bio-hashing based three-factor authentication scheme for telecare medical information systems. J Ambient Intell Humaniz Comput 9(4):1061–1073

Lee SW, Kim HS, Yoo KY (2005) Efficient verifier-based key agreement protocol for three parties without server's public key. Appl Math Comput 167(2):996–1003

Li CT, Chen CL, Lee CC, Weng CY, Chen CM (2018) A novel three-party password-based authenticated key exchange protocol with user anonymity based on chaotic maps. Soft Comput 22(8):2495–2506

Lin JP, Fu JM (2013) Authenticated key agreement scheme with privacy-protection in the three-party setting. IJ Netw Secur 15(3):179–189

Lu Y, Peng H, Yang Y et al (2015) A three-party password-based authenticated key exchange protocol for wireless communications. Inform Technol Control 44(4):404–409

Shen H, Gao C, He D, Wu L (2015) New biometrics-based authentication scheme for multi-server environment in critical systems. J Ambient Intell Humaniz Comput 6(6):825–834

Tallapally S (2012) Security enhancement on simple three party PAKE protocol. Inform Technol Control 41(1):15–22

Wu F, Xu L, Kumari S, Li X (2017a) A privacy-preserving and provable user authentication scheme for wireless sensor networks based on internet of things security. J Ambient Intell Humaniz Comput 8(1):101–116

Wu F, Xu L, Kumari S, Li X, Shen J, Choo KKR, Wazid M, Das AK (2017b) An efficient authentication and key agreement scheme for multi-gateway wireless sensor networks in IoT deployment. J Netw Comput Appl 89:72–85

Xiong H, Choo KKR, Vasilakos AV (2017a) Revocable identity-based access control for big data with verifiable outsourced computing. IEEE Trans Big Data. https://doi.org/10.1109/TBDATA.2017.2697448

Xiong H, Tao J, Yuan C (2017b) Enabling telecare medical information systems with strong authentication and anonymity. IEEE Access 5:5648–5661

Zhu H (2017) A novel two-party scheme against off-line password guessing attacks using new theorem of chaotic maps. KSII Trans Int Inform Syst 11(12):6188–6204

Zhu H, Zhang Y (2017) An efficient chaotic maps-based deniable authentication group key agreement protocol. Wirel Pers Commun 96(1):217–229