An Improved Two-party Password-Authenticated Key Agreement Protocol with Privacy Protection Based on Chaotic Maps

Hongfeng Zhu, Yifeng Zhang
(Corresponding author: Hongfeng Zhu)

Software College, Shenyang Normal University
No. 253, HuangHe Bei Street, HuangGu District, Shenyang 110034, P.R. China
(zhuhongfeng1978@163.com; 1548452125@qq.com)
(Received Jan. 31, 2016; revised and accepted Apr. 23 & May 7, 2016)

Abstract

Since the 1990s, chaotic systems have widely used to cryptography which can be used to design kinds of secure protocols, digital signatures, hash functions and so on. Recently, Guo and Zhang proposed an chaotic public-key cryptosystem based key agreement protocol. In 2015, Lee has proved that Guo et al.'s scheme cannot resist off-line password guess attack. Then, Liu and Xue further point out that Guo et al.'s scheme has redundancy in protocol design and still has some security flaws. In this paper, we further prove that Liu's scheme has four flaws at least and a potential loophole. Moreover, these papers provided no privacy protection which is a very important property in the modern social network. So an improved Two-party Password-Authenticated Key Agreement Protocol with Privacy Protection is proposed for amending these flaws and loophole. Compared with the related literatures recently, our proposed scheme can not only own high efficiency and unique functionality, but is also robust to various attacks and achieves perfect forward secrecy. Finally, we give the security proof and the efficiency analysis of our proposed scheme.

Keywords: Chaotic maps, key agreement, off-line password-guessing attack, privacy protection

1 Introduction

Authenticated key exchange (AKE) allows two or more parties to compute shared keys and also ensures their identities are authentic in insecure networks. The mutual authentication and the key agreement are impartible and the reasons are:

1) A protocol only has the attribute of key agreement will lead the man-in-the-middle attacks at least, just like the first key agreement scheme Diffie-Hellman (D-H) key agreement [1].
2) A protocol only has the attribute of mutual authentication will bring about some function loss. For example, you can use mutual authentication scheme for acquiring E-mail service, but you cannot only use mutual authentication scheme for getting Instant Messaging service, because there is no session key to protect transmissive information. Unlike digital signature needing the third party for arbitration and many other properties, MAKA protocols are only related with the involving participants, so naturally the efficient chaotic cryptosystem is the first candidate.

Compared with other cryptosystem systems, chaotic system has numerous advantages, such as extremely sensitive to initial parameters, unpredictability, deterministic random-like process and so on. In the past few years, cryptography systems based on chaos theory have been studied widely [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16], such as two-party AKE protocols [3, 4, 5, 16], three-party AKE protocols [6], N-party AKE protocols [7], random number generating [8], symmetric encryption [9], asymmetric encryption [10], hash functions [11], digital signature [12], anonymity scheme [13], Multi-server Environment (Centralized Model) [14], Multiple Servers to Server Architecture (Distributed Model) [15].

In 2007, Xiao et al. [16] proposed a chaos-based key agreement protocol. However, Guo and Zhang [3] pointed out that Xiao et al.'s [16] scheme could not resist server spoofing attacks and denial-of-service (DoS) attacks. Furthermore, in Guo and Zhang [3] proposed an improved scheme, which claimed that their protocol could resist the security flaws of Xiao et al.'s protocol. Moreover, in [4], the author has proved that Guo et al.'s scheme cannot resist off-line password guess attack. However, the improved scheme in [4] introduces a traditional asymmetric encryption algorithm to address the issue. Very recently, Liu and Xue [5] pointed out Guo et al.'s protocol [3] has unnecessary redundancy in protocol design which will increase the implementation time of key agreement to bring about more unnecessary delay and also has the threat of replay attacks and DoS attacks.

In this paper, we demonstrate that Liu et al.'s protocol [5] has still security problems: password Guessing Attacks for privileged-insider, off-line Password Guessing Attacks for any adversary, stolen-verifier attacks and the complications from Off-line Password Guessing Attacks and Potential Loophole of XOR Operation. Based on [5], we provide an improved secure password and chaos-based two-party key agreement protocol. The main contributions are shown as below.

1) By analyzing of Liu et al.'s scheme, we found four flaws (password Guessing Attacks for privileged-insider, off-line Password Guessing Attacks for any adversary, stolen-verifier attacks and the complications from Off-line Password Guessing Attacks) and one loophole (Potential Loophole of XOR Operation).

2) The improved protocol provides privacy protection. Moreover, for eliminating Potential Loophole of XOR Operation and at the same time for improving efficiency, the proposed scheme uses multiplication in finite field method instead of XOR operation for two different length messages.

The rest of the paper is organized as follows: Review and cryptanalysis of Liu et al.'s protocol is given in Section 2. Next, an improved privacy-protection two-party password-authentication key agreement protocol is described in Section 3. Then, the security analysis and efficiency analysis are given in Section 4 and Section 5. This paper is finally concluded in Section 6.

2 Review of Liu et al.'s Protocol

In this section, we first describe the Chebyshev chaotic map, which has semigroup property and can be used to design chaos-based public-key cryptosystems. After that, we introduce Liu et al.'s two-party key agreement protocol and give its security analysis.

2.1 Chebyshev Chaotic Maps

Let n be an integer and let x be a variable with the interval $[-1, 1]$. The Chebyshev polynomial [17]. $T_n(x) : [-1, 1] \to [-1, 1]$ is defined as $T_n(x) = \cos(n \arccos(x))$ Chebyshev polynomial map $T_n : R \to R$ of degree n is defined using the following recurrent relation:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x). \tag{1}$$

where, $n \geq 2$, $T_0(x) = 1$, and $T_1(x) = x$. The first few Chebyshev polynomials are:

$$\begin{aligned} T_2(x) &= 2x^2 - 1, \\ T_3(x) &= 4x^3 - 3x, \\ T_4(x) &= 8x^4 - 8x^2 + 1, \\ &\vdots \end{aligned}$$

One of the most important properties is that Chebyshev polynomials are the so-called semi-group property which establishes that

$$T_r(T_s(x)) = T_{r \cdot s}(x). \tag{2}$$

An immediate consequence of this property is that Chebyshev polynomials commute under composition:

$$T_r(T_s(x)) = T_s(T_r(x)). \tag{3}$$

In order to enhance the security, Zhang [18] proved that semi-group property holds for Chebyshev polynomials defined on interval $(-\infty, +\infty)$. In our proposed protocol, we utilize the enhanced Chebyshev polynomials:

$$T_n(x) = (2xT_{n-1}(x) - T_{n-2}(x))(\bmod N), \tag{4}$$

where, $n \geq 2$, $x \in (-\infty, +\infty)$, and N is a large prime number. Obviously,

$$T_{r \cdot s}(x) = T_r(T_s(x)) = T_s(T_r(x)). \tag{5}$$

Definition 1. *Semi-group property of Chebyshev polynomials:*

$$\begin{aligned} T_r(T_s(x)) &= \cos(r\cos^{-1}(s\cos^{-1}(x))) \\ &= \cos(rs\cos^{-1}(x)) \\ &= T_{sr}(x) \\ &= T_s(T_r(x)). \end{aligned}$$

Definition 2. *Given x and y, it is intractable to find the integer s, such that $T_s(x) = y$. It is called the Chaotic Maps-Based Discrete Logarithm problem (CMBDLP).*

Definition 3. *Given x, $T_r(x)$, and $T_s(x)$, it is intractable to find $T_{rs}(x)$. It is called the Chaotic Maps-Based Diffie-Hellman problem (CMBDHP).*

2.2 Review of Liu et al.'s Protocol [5]

Assume that the user A and the server S share the hash value $h_{pw} = H(ID_A||PW_A)$ of A's password PW_A and A's identification ID_A. The hash value of the user's password is required to be stored in the server. Figure 1 shows the main process of Liu et al.'s protocol.

1) User A $\to$ Server S: $\{ID_A, N_1, r_a, T_1, T_2\}$.
User A generates a random number $r_a \in [-1, 1]$, a random integer r and a timestamp value N_1, then computes $T_r(r_a)$. Next, A computes the functions T_1 and T_2 as follows: $T_1 = H(h_{pw}||r_a||N_1) \oplus H(T_r(r_a))$, $T_2 = H(H(T_r(r_a)))$.

2) Server S $\rightarrow$ User A: $\{r_b, T_3, H(T_s(r_a))\}$.
After receiving the message, the server first verifies the timeliness of it: timestamp whether the N_1 in the received message is in a permitted time window. If not, the server S stops here. Otherwise, S goes on to take out his own copy of h_{pw} by using the index "ID_A," and computes the function K_{B_1} as follows: $K_{B_1} = H(h_{pw}||r_a||N_1)$. Then S computes the function $K_{B_1} \oplus T_1$ to get $X_1 (= H(T_r(r_a)))$ and further verifies whether $H(X_1) = T_2$. If not, B stops here; otherwise, B generates a random number $r_b \in [-1, 1]$ and a random integer s. Next S computes the function $T_s(r_a)$.Then S computes the functions T_3 and T_4 as follows: $T_3 = H(h_{pw}||r_a||r_b) \oplus T_s(r_a)$, $T_4 = H(T_s(r_a))$.

3) User A $\rightarrow$ Server S: $\{T_5\}$.
After receiving the message, User A computes the function $K_A = H(h_{pw}||r_a||r_b)$.Then A computes the function $K_A \oplus T_3$ to get the value of $X_2 (= (r_a))$ and verifies whether $H(X_2)$ is equal to the received T_4. If not, A stops here; otherwise, the server S is authenticated. After that, A computes the function $T_5 = H(h_{pw} \oplus r_b) \oplus T_r(r_a)$. Finally, A sends T_5 to S.

4) After receiving the message, the server S computes $K_{B_2} = H(h_{pw} \oplus r_b)$.Then S computes the function $K_{B_2} \oplus T_5$ to get the value of T_2 which is received in (1). If not, B stops here; otherwise, the user A is authenticated.

5) Respectively, A and S can calculate the share session key $K_{session} = T_r(T_s(r_a)) = T_s(T_r(r_a)) = T_{rs}(r_a)$.

2.3 Security of Liu et al.'s Protocol [5]

1) Fails to Prevent Password Guessing Attacks for privileged-insider of the server S.
In real environments, the user Alice may register with a number of servers by using a common password PW_A and the identity ID_A for his/her convenience. Thus, the privileged-insider of server may try to use the knowledge of user's identity and PW_A to access other servers. The details of password guessing attack in Liu's scheme are described as follows:

Step 1: In Liu's protocol, they assume that the user A and S share the hash value $h_{pw} = H(ID_A||PW_A)$.

Step 2: The privileged-insider of the server S guesses a password PW_A^* and computes $H(ID_A||PW_A^*)$.

Step 3: The privileged-insider of the server S compares $H(ID_A||PW_A^*)$ with h_{pw}.
A match in **Step 3** above indicates the correct guessing of Alice's password and the privileged-insider of server S succeeds to guess the low-entropy password $PW_A^* = PW_A$. Otherwise, the privileged-insider of server S repeats **Step 2**.
Note that above-mentioned steps can be done by off-line manner and Tang et al. [19] have modelled the password guessing attacks can be carried out between the challenger and a polynomial-time attacker.

2) Fails to Prevent Off-line Password Guessing Attacks for any adversary.
The details of off-line password guessing attack for any adversary in Liu's scheme are described as follows:

Step 1: In the Liu's protocol, an adversary can get all the transmitting messages, and he records four related messages $\{ID_A, r_b, T_2, T_5\}$.

Step 2: The adversary guesses a password PW_A^* and computes $H(H(H(ID_A||PW_A^*) \oplus r_b) \oplus T_5)$.

Step 3: The adversary compares $H(H(H(ID_A \; || \; PW_A^*) \oplus r_b) \oplus T_5)$ with T_2.
A match in **Step 3** above indicates the correct guessing of Alice's password and the adversary succeeds to guess the low-entropy password $PW_A^* = PW_A$. Otherwise, the adversary repeats **Step 2**. The main reason is that the Liu's protocol has the design defect: Using the transmitting messages, anyone can construct a function which only including one input variable password and a related output T_2.

3) Fails to Prevent Stolen-verifier attacks.
An adversary gets the verifier table from servers by a hacking way, and then the adversary can launch any other attack which called stolen-verifier attacks. There is a verification table in the server side because the server and the user have shared the hash value $h_{pw} = H(ID_A||PW_A)$. The verification table can lead three problems: security of stolen-verifier attack, hard to maintain the verification table and wasting storage space.

4) The complications from Off-line Password Guessing Attacks.
Firstly, if an adversary gets many passwords of users by launching off-line password guessing attacks, he can also carry out DoS (Denial of Service) attacks. Secondly, the adversary can initiate impersonation attack to cheat a legal user by playing the server S, or cheat the server S by playing the legal user. Thirdly, the adversary may be eavesdropping all the time while hiding the case of the password leaking just for getting some important information.

5) PLXO (Potential Loophole of XOR Operation) [20].
First of all, there exists a kind of Potential Loophole about using with $\oplus$ in the whole Lu's scheme. The XOR operation must assure the same binary digits on both sides of.

Assume that $t = a \oplus b$, a is is short and b is long.So there are three scenarios as follows:

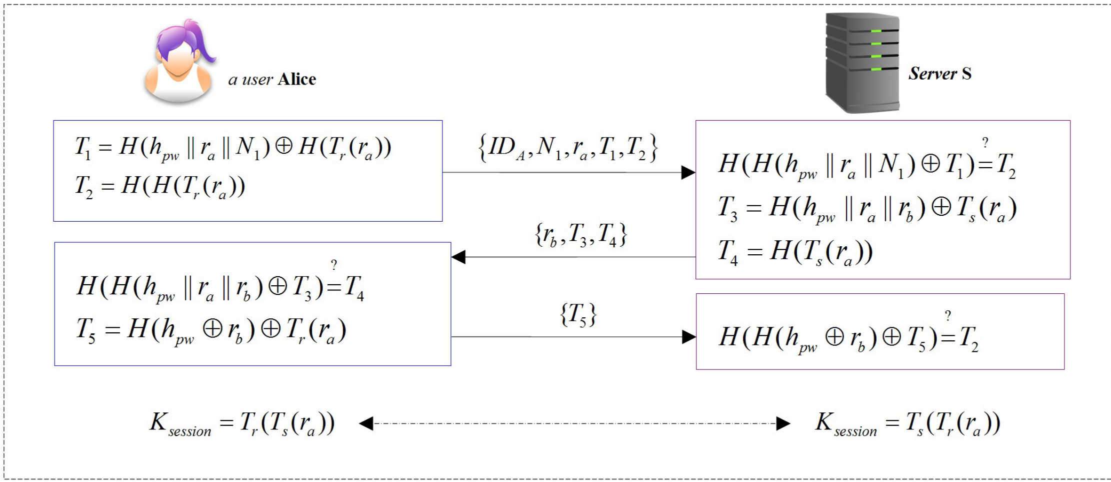

Figure 1: The process of Liu et al.'s protocol

Case 1: Extended a.
However, a may be the ID of user (such as in literature [5]), so the ID of user is not practical and friendly enough.

Case 2: Shorten b.
However, b may be a random number (such as in literature [5]), if b is shortened, it can be easily guessed. And if the protocol transmits a (may be the ID) in plaintext, anyone will get the b.

Case 3: Pad a.

Definition 4. *(Leak attack.) Leak attack is a kind of intercept attack that the attackers use various technologies to obtain the useful information from the messages eavesdropped from public channels.*

Definition 5. *(XOR with pad operation leaking attack.) This kind of attack is due to use XOR operation in a wrong way, which will lead to leak some sensitive information, and finally an adversary can get part of useful information, even the session key is not being detected. In literature [5], Trudy can launch a XOR with pad operation leaking attack.*

For pad a method, on one side, according to Kerckhoffs's principle: A cryptosystem should be secure even if everything about the system, except the key, is public knowledge. On the other side, the opposite peer must know the pad algorithm in order to decrypt the XORed cipher text. Based on above-mentioned, the pad method/algorithm must be opened, then $t = (a||pad) \oplus b$, and the values of a and b must be strictly private.

For example, we consider $T_5 = H(h_{pw} \oplus r_b) \oplus T_r(r_a)$, and we assume that the $H(h_{pw} \oplus r_b)$ has l bits, $T_r(r_a)$ has m bits. The leaking bits are $(m - l)$ bits (assume $(m - l)$). The shorter of the $H(h_{pw} \oplus r_b)$, the more of leaking information about $T_r(r_a)$. The Figure 2 shows that partial of $T_r(r_a)$ will be leak.

3 The Improved Two-party PAKA Protocol with Privacy Protection

In this section, we give an improved chaotic maps-based password-authentication key agreement scheme which consists of three phases: user registration phase, the improved two-party PAKA with privacy protection phase, password changing phase.

Table 2 is the notations used in this paper.

Table 1: Notations

Symbol	Definition
ID_A, ID_S	The identities of the user and the server, respectively
PW_A	The password of the user (Alice)
R, a, b	Random numbers
$(x, T_k(x))$	Public key based on Chebyshev chaotic maps for the server
k	Secret key based on Chebyshev chaotic maps for the server
H	A secure one-way hash function
$\|\|$	concatenation operation
T	Timestamp

3.1 User Registration Phase

Figure 3 illustrates the user registration phase.

1) User A $\rightarrow$ Server S: $\{ID_A, H(R||PW_A)\}$.
When a user wants to be a new legal user, she chooses her identity ID_A, a random number R,and computes $H(R||PW_A)$. Then Alice submits $ID_A, H(R||PW_A)$ to the S via a secure channel.

2) Server S $\rightarrow$ User A: B.

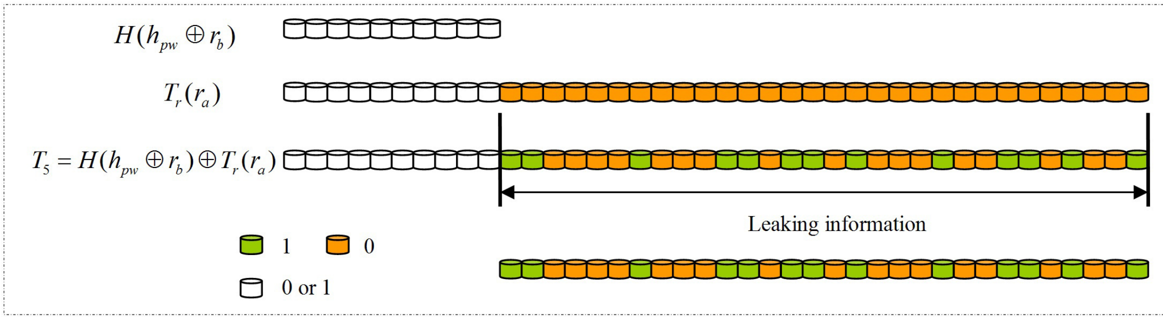

Figure 2: The process of how to leak some information

Upon receiving $ID_A, H(R||PW_A)$ from Alice, the S computes $B = H(ID_A||k) \oplus H(R||PW_A)$, where k is the secret key of the server S. Then Alice stores $\{R, B\}$ in a secure way.

3.2 The Improved Two-party PAKA with Privacy Protection Phase

This concrete process is presented in the following Figure 4.

1) User A $\rightarrow$ Server S: $\{T_a(x), C_1, C_2\}$.
If Alice wishes to consult some personal issues establish with S in a anonymous way, she will input password and compute $B^* = B \oplus H(R||PW_A)$, and then choose a random integer number a and compute $T_a(x)$, $C_1 = T_aT_k(x)(ID_A||T)$, $C_2 = H(B^*||C_1||ID_S)$. After that, Alice sends $\{T_a(x), C_1, C_2\}$ to S where she wants to get the server's service.

2) Server S $\rightarrow$ User A: $\{T_k(b), C_3, C_4\}$.
After receiving the message $\{T_a(x), C_1, C_2\}$, S firstly must confirm the identity of this message and check the timestamp. So based on the private key k, S computes $C_1/T_kT_a(x) = ID_A||T$ to get the source of this message and timestamp. If T is passed validation, S will compute $B^* = H(ID_A||k)$ and verifies $H(B^*||C_1||ID_S) \stackrel{?}{=} C_2$. If above equation holds, that means Alice is a legal user, or S will abort this process. After authenticating Alice, S chooses a random b and computes $C_3 = T_kT_a(x)b$, $C_4 = H(B^*||T_k(b)||T)$. Finally S sends $\{T_k(b), C_3, C_4\}$ to Alice.

3) User A $\rightarrow$ Server S: $\{C_5, C_6\}$.
Because $T_aT_k(x)$ has already computed before, Alice can get $b = C_3/T_aT_k(x)$ directly. Next, Alice computes $H(B^*||T_k(b)||T)$ and verifies $H(B^*||T_k(b)||T) \stackrel{?}{=} C_4$. If above equation holds, that means S is a legal server, or Alice will abort this process. After authenticating S, Alice computes $C_5 = T_aT_k(x)T_a(b)$, $C_6 = H(T_a(b))$ and sends $\{C_5, C_6\}$ to S. Finally, Alice computes the session key $K_{session} = T_a(T_k(b))$ locally.

4) After receiving the message $\{C_5, C_6\}$, S computes $T_a(b) = C_5/T_kT_a(x)$ and verifies $H(T_a(b)) \stackrel{?}{=} C_6$. If above equation holds, S will computes the session key $K_{session} = H(T_k(T_a(b)))$ locally.

3.3 Password Changing Phase

Figure 5 illustrates the password changing phase.

1) User A $\rightarrow$ Server S: $\{T_a(x), C_1, C_2, C_3\}$.
When Alice wants to change her password, she chooses $PW_A^{'}$,two random numbers $R^{'}$, a and computes $B^* = B \oplus H(R||PW_A)$, $T_a(x)$, $C_1 = T_aT_k(x)(ID_A||T)$, $C_2 = B^* \oplus H(R^{'}||PW_A^{'})$, $C_3 = H(B^*||C_1||C_2)$. Then Alice sends $\{T_a(x), C_1, C_2, C_3\}$ to the S.

2) Server S $\rightarrow$ User A: $\{C_4, C_5\}$.
Upon receiving $\{T_a(x), C_1, C_2, C_3\}$ from Alice, firstly must confirm the identity of this message and verify timestamp. So based on the private key k, S computes $C_1/T_kT_a(x) = ID_A||T$ to get the source of this message and timestamp. If T is passed validation, S computes $B^* = H(ID_A||k)$ and verifies $H(B^*||C_1||C_2) \stackrel{?}{=} C_3$. If above equation holds, that means Alice is a legal user, or S will abort this process. After authenticating Alice, S computes

$$
\begin{aligned}
H(R^{'}||PW_A^{'}) &= C_2 \oplus B^*, B^{'} \\
&= H(ID_A||k) \oplus H(R^{'}||PW_A^{'}), \\
C_4 &= T_kT_a(x)B^{'}, \\
C_5 &= H(B^{'}||T),
\end{aligned}
$$

and sends $\{C_4, C_5\}$ to Alice.

3) After receiving the message $\{C_4, C_5\}$, Alice computes stores $B^{'} = C_4/T_aT_k(x)$ and verifies $H(B^{'}||T) \stackrel{?}{=} C_5$. If above equation holds, Alice will store $\{R, B\}$ in a secure way.

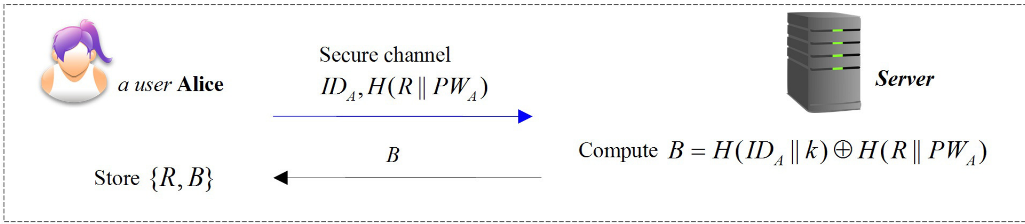

Figure 3: User registration phase

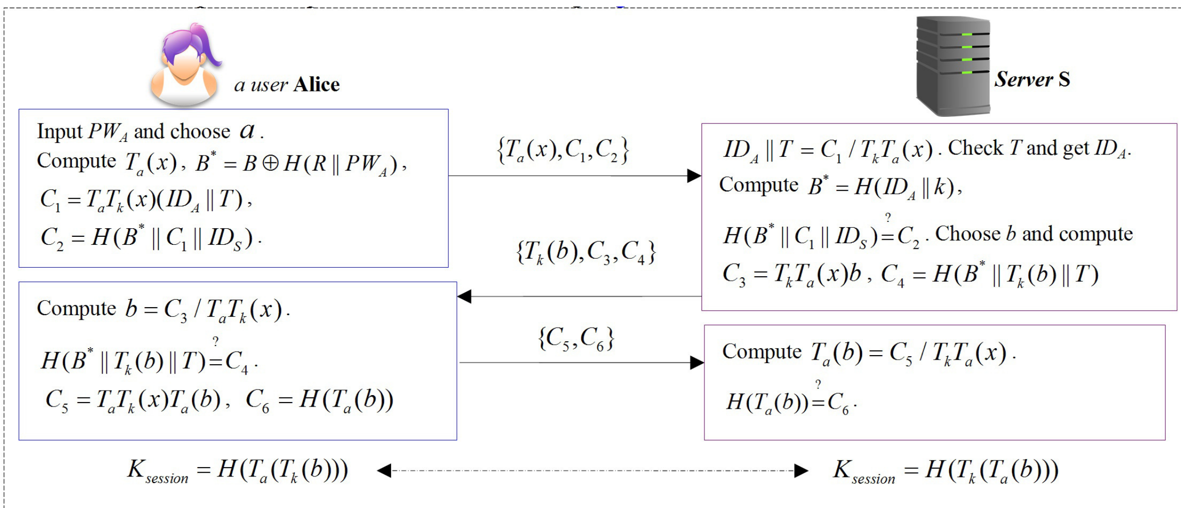

Figure 4: The improved two-party PAKA with privacy protection

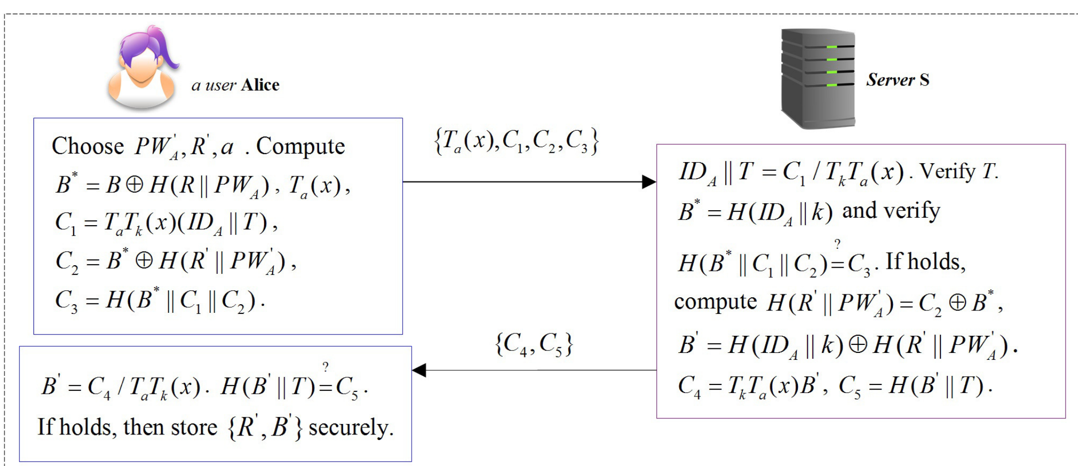

Figure 5: Password changing phase

4 Security Analysis

4.1 Security Proof Based on the BAN Logic [21]

For convenience, we first give the description of some notations (Table 2) used in the BAN logic analysis and define some main logical postulates (Table 3) of BAN logic.

According to analytic procedures of BAN logic and the requirement of deniable scheme, our NIDA scheme should satisfy the following goals in Table 4.

First of all, we transform the process of our protocol (The improved two-party PAKA with privacy protection phase) to the following idealized form.

(Alice $\rightarrow$ Server)C_1 : Server $\triangleleft$ $T_a(x)$, $T_aT_k(x)(ID_A||T)$, $(B^*||T_aT_k(x)(ID_A||T)||ID_S)$;

(Server $\rightarrow$ Alice)C_2 : Alice $\triangleleft$ $T_k(b)$, $T_kT_a(x)b$, $(B^* \,||\, T_k(b) \,||\, T)$;

(Alice $\rightarrow$ Server)C_3 : Server $\triangleleft$ $T_aT_k(x)T_a(b)$, $(T_a(b))$.

According to the description of our protocol, we could make the following assumptions about the initial state, which will be used in the analysis of our protocol in Table 5.

Based on the above assumptions, the idealized form of our protocol is analyzed as follows. The main steps of the proof are described as follows:

For C_1: According to the ciphertext C_1 and P_4, P_7 and attributes of chaotic maps, and relating with R_1, we could get:

S_1 : **Server** $|\equiv$ **Alice** $|\sim C_1$.
Based on the initial assumptions P_2, P_4, and relating with R_2, we could get:

S_2 : **Server** $|\equiv \#C_1$.
Combine $S_1, S_2, P_2, P_4, P_7, R_3$ and attributes of chaotic maps, we could get:

S_3 : **Server** $|\equiv \#ID_A$, $T_a(x)$, $(B^* \,||\, T_aT_k(x)(ID_A \,||\, T) \,||\, ID_S)$.
Based on R_5, we take apart S_3 and get:

S_4 : **Server** $|\equiv \#ID_A$, S_5 : **Server** $|\equiv \#T_a(x)$.
Combine S_3, S_4 and attributes of chaotic maps, we can get the fresh and privacy protection about Alice's identity. Combine S_5 and attributes of chaotic maps, we can authenticate the message $T_a(x)$ is fresh and comes from Alice exactly.

For C_2: According to the ciphertext C_2 and P_1, P_5, P_6 and attributes of chaotic maps, and relating with R_1, we could get:

S_6 : **Alice** $|\equiv$ **Server** $|\sim C_2$.
Based on the initial assumptions P_3, P_5, and relating with R_2, we could get:

S_7 : **Alice** $|\equiv \#C_2$.
Combine $S_6, S_7, P_3, P_5, P_6, R_3$ and attributes of chaotic maps, we could get:

S_8 : **Alice** $|\equiv \#T_k(b), (B^*||T_k(b)||T)$.
Based on R_5, we take apart S_8 and get:

S_9 : **Alice** $|\equiv \#T_k(b)$, S_{10} : **Alice** $|\equiv \#\ (B^* \,||\, T_k(b) \,||\, T)$.
Combine S_8, S_9 and attributes of chaotic maps, we can get the fresh and privacy protection about $T_k(b)$. Combine S_{10} and attributes of secure chaotic maps-based hash function, we can authenticate the message $T_k(b)$ comes from Server exactly.

For C_3: According to the ciphertext C_3 and P_7 and attributes of chaotic maps, and relating with R_1, we could get:

S_{11} : $Server| \equiv Alice|\tilde{}\, C_3$.
Based on the initial assumptions P_2, P_4, and relating with R_2, we could get:

S_{12} : $Server| \equiv \#C_3$.
Combine $S_{11}, S_{12}, P_2, P_4, P_7, R_3$ and attributes of chaotic maps, we could get:

S_{13} : $Server| \equiv \#T_a(b), (T_a(b))$.
Based on R_5, we take apart S_3 and get:

S_{14} : $Server| \equiv \#T_a(b)$, S_{15} : $Server| \equiv \#(T_a(b))$.
Combine S_{13}, S_{14} and attributes of chaotic maps, we can get the fresh and privacy protection about $T_a(b)$. Combine S_{15} and attributes of secure chaotic maps-based hash function, we can authenticate the message $T_a(b)$ comes from Server exactly.

Combination:
Because Alice and Server communicate each other just now, they confirm the other is on-line. Moreover, since Server can get ID_A from the $T_aT_k(x)(ID_A||T)$ with his own secret key, and based on $S_4, S_5, S_{14}, S_{15}, R_4$ with chaotic maps problems, we think that the server could get the session key $K_{session} = H(T_k(T_a(b)))$ and Goal 3. $Server \equiv (Server \overset{K_{session}}{\longleftrightarrow} Alice)$, Goal 4. $Server| \equiv Alice| \equiv (Server \overset{K_{session}}{\longleftrightarrow} Alice)$. At the same way, based on S_9, S_{10}, R_4 with chaotic maps problems, we think that Alice could get the session key $K_{session} = T_a(T_k(b))$ and Goal 1. $Alice| \equiv (Alice \overset{K_{session}}{\longleftrightarrow} Server)$, Goal 2. $Alice| \equiv Server| \equiv (Alice \overset{K_{session}}{\longleftrightarrow} Server)$.

4.2 Resistance to Possible Attacks

In this section, we analyze the process of security proof privacy protection, Resistance to stolen-verifier attacks, Impersonation attack, Man-in-the-middle attack, Replay attack, Known-key security, Perfect forward secrecy and Guessing attacks (On-line or off-line) respectively.

Table 2: Notations of the BAN logic

Symbol	Definition
$P\mid\equiv X$	The principal P believes a statement X, or P is entitled to believe X.
$\#(X)$	The formula X is fresh.
$P\mid\Rightarrow X$	The principal P has jurisdiction over the statement X.
$P \lhd X$	The principal P sees the statement X.
$P\mid\sim X$	The principal P once said the statement X.
(X, Y)	The formula X or Y is one part of the formula (X, Y).
$\langle X\rangle_Y$	The formula X combined with the formula Y.
$\{X\}_Y$	The formula X is encrypted under the key K.
$(X)_Y$	The formula X is chaotic maps-based hash function with the key K.
$P \overset{K}{\longleftrightarrow} Q$	The principals P and Q use the shared key K to communicate. The key K will never be discovered by any principal except P and Q.
$\overset{K}{\longrightarrow} P$	The public key of P, and the secret key is described by K^{-1}.

Table 3: Logical postulates of the BAN logic

Symbol	Definition
$\frac{P\mid\equiv P \overset{K}{\longleftrightarrow} Q, P\{X\}_K}{P\mid\equiv Q\mid\sim X}$	The message-meaning rule (R_1)
$\frac{P\mid\equiv\#(X)}{P\mid\equiv\#(X,Y)}$	The freshness-conjunction rule (R_2)
$\frac{P\mid\equiv\#(X), P\mid\equiv Q\mid\sim X}{P\mid\equiv Q\mid\equiv X}$	The nonce-verification rule (R_3)
$\frac{P\mid\equiv Q\mid\Rightarrow X, P\mid\equiv Q\mid\equiv X}{P\mid\equiv X}$	The jurisdiction rule (R_4)
$\frac{P\mid\equiv Q\mid\equiv(X,Y)}{P\mid\equiv Q\mid\equiv X}$	The belief rules (R_5)
Remark 3: Molecule can deduce denominator for above formulas.	

Table 4: Goals of the proposed scheme

Goals	
Goal 1. $Alice\mid\equiv(Alice \overset{K_{session}}{\longleftrightarrow} Server)$;	Goal 2. $Alice\mid\equiv Server\mid\equiv(Alice \overset{K_{session}}{\longleftrightarrow} Server)$;
Goal 3. $Server\equiv(Server \overset{K_{session}}{\longleftrightarrow} Alice)$;	Goal 4. $Server\mid\equiv Alice\mid\equiv(Server \overset{K_{session}}{\longleftrightarrow} Alice)$;

Table 5: Assumptions about the initial state of our protocol

Initial states	
$P_1 : Alice\mid\equiv \overset{T_k(x)}{\longrightarrow} Server$	
$P_2 : Server\mid\equiv Server \overset{B^*}{\longleftrightarrow} Alice$	$P_3 : Alice\mid\equiv Server \overset{B^*}{\longleftrightarrow} Alice$
$P_4 : Alice\mid\equiv\#(a)$	$P_5 : Server\mid\equiv\#(b)$
$P_6 : Alice\mid\equiv Alice \overset{T_aT_k(x)}{\longleftrightarrow} Server$	$P_7 : Server\mid\equiv Alice \overset{T_kT_a(x)}{\longleftrightarrow} Server$

Table 6: Security of our proposed protocol

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14
[5](2010)	No	Mutual	No	No	Yes	Yes	No	Yes	Yes	No	No	No	No	No
[7](2015)	No	Mutual	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
[8](2015)	No	Mutual	No	No	Yes	Yes	Yes	Yes	No	No	No	No	No	No
Ours	Yes	Mutual	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	BAN	Yes

S1: Single registration; S2: Authentication; S3: Privacy protection; S4:Resistance to stolen-verifier attack; S5: Resistance to impersonation attack; S6: man-in-the-middle attack; S7: Resistance to replay attack; S8: Known-key security; S9: Perfect forward secrecy; S10: Guessing attacks (On-line or off-line) (Including Prevent Password Guessing Attacks for privileged-insider or for any adversary) S11: Resistance to Potential Loophole of XOR Operation; S12: Update password phase S13: Formal security proof S14: Hiding timestamp Yes/No: Support/Not support

Privacy protection. The node which possesses the secret key k can compute $T_kT_a(x)$ and get the user's ID, so only the server knows the identity of the user. Furthermore, only the user and the server can compute the B^*, so the user need not get the plaintext of identity of the server and convinces the peer is the server.

Resistance to stolen-verifier attacks. In the proposed scheme, the server side need not maintain any verification table. Thus, the stolen-verifier attack is impossible to initiate in the proposed scheme.

Impersonation attack. An adversary cannot impersonate anyone of the user and the server. The B* and the secret key k can achieve authentication and confidentiality. The $\{a, b, T\}$ can achieve freshness and associativity of all the transmissive messages. So there is no way for an adversary to have a chance to carry out impersonation attack. Furthermore, because Alice is an identity hiding and legal user, an adversary can not impersonate Alice at all.

Man-in-the-middle attack. Because $C_i(1 \leqslant i \leqslant 6)$ contain the participants's identities, timestamp or nonces and The $\{a, b, T\}$ can achieve freshness and associativity of all the transmissive messages, a man-in-the-middle attack cannot succeed.

Replay attack. That any message of Alice was replayed by an adversary is meaningless. Because Alice is an ID hiding user, the adversary only can create a vision user to initiate the replay attack. Moreover the $\{a, b, T\}$ can achieve freshness and associativity of all the transmissive messages.

Known-key security. Since the session key $SK = T_aT_k(b) = T_kT_a(b)$ is depended on the random nonces a and b, and the generation of nonces is independent in all sessions, an adversary cannot compute the previous and the future session keys when the adversary knows one session key. And in the password update phase, any session key is only used once, so it has known-key security attribute.

Perfect forward secrecy. In the proposed scheme, the session key $SK = T_aT_k(b) = T_kT_a(b)$ is related with a and b, which were randomly chosen by Alice and the server S respectively. Because of the intractability of the chaotic maps problems, an adversary cannot compute the previously established session keys.

Guessing attacks (On-line or off-line). Any transferred messages on the public channel have not password involved, so guessing attacks can not happen.

From the Table 6, we can see that the proposed scheme can provide privacy protection, perfect forward secrecy and so on. As a result, the proposed scheme is more secure and has much functionality compared with the recent related scheme.

5 Efficiency Analysis

Compared to RSA and ECC, Chebyshev polynomial computation problem offers smaller key sizes, faster computation, as well as memory, energy and bandwidth savings. To be more precise, on an Intel Pentium4 2600 MHz processor with 1024 MB RAM, where n and p are 1024 bits long, the computational time of a one-way hashing operation, a symmetric encryption/decryption operation, an elliptic curve point multiplication operation and Chebyshev polynomial operation is 0.0005s, 0.0087s, 0.063075s and 0.02102s separately [22]. Moreover, the computational cost of XOR operation could be ignored when compared with other operations. Table 7 shows performance comparisons between our proposed scheme and the literatures of [3, 4, 5]. we sum up these formulas into one so that it can reflect the relationship among the time of algorithms intuitively. $T_p \approx 10T_m \approx 30T_c \approx 72.6T_s \approx 1263.24T_h$, where: T_p: Time for bilinear pair operation, T_m: Time for a point scalar multiplication operation, T_c: The time for executing the $T_n(x)$ mod p in Chebyshev polynomial,

Table 7: Comparisons between our proposed scheme and the related literatures

Protocols(Authentication phase)		[5] (2010)	[7] (2015)	[8] (2015)	Ours
Computation	User	$11T_h + 2T_c + 6T_{xor}$	$6T_h + 2T_c + 1T_{xor}$	$6T_h + 2T_c + 3T_{xor}$	$4T_h + 2T_c + 1T_{xor}$
	Server	$11T_h + 2T_c + 5T_{xor}$	$6T_h + 2T_c + 1T_{xor}$	$6T_h + 2T_c + 4T_{xor}$	$4T_h + 2T_c$
	Total	$22T_h + 4T_c + 11T_{xor} \approx 190.432\ T_h$	$12T_h + 4T_c + 2T_{xor} \approx 180.432\ T_h$	$12T_h + 4T_c + 7T_{xor} \approx 180.432\ T_h$	$8T_h + 4T_c + 1T_{xor} \approx 176.432\ T_h$
Communication	Messages	6	2	3	3
	rounds	6	2	3	3
Design	Concise design	No	No	Yes	Yes
	Number of nonces	4	3	4	2
	Model	*Random Oracle*	*Random Oracle*	*Random Oracle*	*Random Oracle*
T_h : *Time for Hash operation* T_{xor} : *Time forXORedoperation*					
T_c : *The time for executing the* $T_n(x)$ *modp in Chebyshev polynomial using the algorithm in literature* [9]					

T_s: Time for symmetric encryption algorithm, T_h: Time for Hash operation. As in Table 6 and Table 7, we can draw a conclusion that the proposed scheme has achieved the improvement of both efficiency and security.

6 Conclusion

In the paper, we give four flaws and one loophole in Liu et al.'s scheme, and then propose an improved protocol which amends all the flaws and provides the privacy protection at the same time. But what I want to emphasize is that all the plaintexts, even timestamp, are protect by our proposed scheme for achieving privacy protection, and that is to say, the attacker can get only some ciphertexts but nothing. Finally, after comparing with related literatures respectively, we found our proposed scheme has satisfactory security, efficiency and functionality. Therefore, our protocol is more suitable for practical applications.

Acknowledgments

This work is supported by the Liaoning Provincial Natural Science Foundation of China (Grant No. 201602680).

References

[1] P. Bergamo, P. D. Arco, A. D. Santis and L. Kocarev, "Security of public-key cryptosystems based on Chebyshev polynomials," *IEEE Transactions on Circuits and Systems*, vol. 52, no. 7, pp. 1382–1393, 2005.

[2] M. Burrows, M. Abadi and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.

[3] J. Chen, J. Zhou and K.W. Wong, "A modified chaos-based joint compression and encryption scheme," *IEEE Transactions on Circuits and Systems*, vol. 58, no. 2, pp. 110–114, 2011.

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

[5] X. Guo and J. Zhang, "Secure group key agreement protocol based on chaotic hash," *Information Sciences*, vol. 180, no. 20, pp. 4069–4074, 2010.

[6] C. Kai and W. C. Kuo, "A new digital signature scheme based on chaotic maps," *Nonlinear Dynamics*, vol. 74, pp. 1003–1012, 2013.

[7] T. F. Lee, "Enhancing the security of password authenticated key agreement protocols based on chaotic maps," *Information Sciences*, vol. 290, pp. 63–71, 2015.

[8] Y. Liu and K. Xue, "An improved secure and efficient password and chaos-based two-party key agreement protocol," *Nonlinear Dynamics*, vol. 84, no. 2, pp. 549–557, 2016.

[9] L. Kocarev and S. Lian, "Chaos-based cryptography: A brief overview," *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6–21, 2002.

[10] F. Özkaynak, "Cryptographically secure random number generator with chaotic additional input," *Nonlinear Dynamics*, vol. 78, no. 3, pp. 2015–2020, 2014.

[11] Q. Tang and K. K. R. Choo, "Secure password-based authenticated group key agreement for data-sharing peer-to-peer networks," *Lecture Notes in Computer Science*, vol. 3989, Springer, pp. 162–177, 2006.

[12] H. R. Tseng, R. H. Jan and W. Yang, "A chaotic maps-based key agreement protocol that preserves user anonymity," in *IEEE International Conference on Communications (ICC'09)*, pp. 1–6, 2009.

[13] H. Wang, H. Zhang, J. Li and X. U. Chen, "A(3,3) visual cryptography scheme for authentication," *Journal of Shenyang Normal University (Natural Science Edition)*, vol. 31, no. 3, pp. 397–400, 2013.

[14] X. Wang and J. Zhao, "An improved key agreement protocol based on chaos," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 12, pp. 4052–4057, 2010.

[15] D. Xiao, X. Liao and S. Deng, "A novel key agreement protocol based on chaotic maps, " *Information Sciences*, vol. 177 no. 4, pp. 1136–1142, 2007.

[16] S. J. Xu, X. B. Chen, R. Zhang, Y. X. Yang and Y. C. Guo, "An improved chaotic cryptosystem based on circular bit shift and XOR operations," *Physics Letters A*, vol. 376, no. 10, pp. 1003–1010, 2012.

[17] L. Zhang, "Cryptanalysis of the public key encryption based on multiple chaotic systems," *Chaos Solitons Fractals*, vol. 37, no. 3, pp. 669–674, 2008.

[18] H. F. Zhu, "Cryptanalysis and provable improvement of a chaotic maps-based mobile dynamic ID authenticated key agreement scheme," *Security and Communication Networks*, vol. 8, no. 17, pp. 2981–2991, 2015.

[19] H. F. Zhu, "Flexible and password-authenticated key agreement scheme based on chaotic maps for multiple servers to server architecture," *Wireless Personal Communications*, vol. 82, no. 3, pp. 1697–1718, 2015.

[20] H. F. Zhu, "A provable privacy-protection system for multi-server environment," *Nonlinear Dynamics*, vol. 82, no. 1, pp. 835–849, 2015.

[21] H. F. Zhu, "A provable one-way authentication key agreement scheme with user anonymity for multi-server environment," *KSII Transactions on Internet And Information Systems*, vol. 9, no. 2, pp. 811–829, 2015.

[22] H. F. Zhu, "Sustained and authenticated of a universal construction for multiple key agreement based on chaotic maps with privacy preserving," *Journal of Internet Technology*, vol. 17, no. 5, pp. 1–10, 2016.

Biography

Hongfeng Zhu obtained his Ph.D. degree in Information Science and Engineering from Northeastern University. Hongfeng Zhu is a full associate professor of the Kexin software college at Shenyang Normal University.He is also a master's supervisor. He has research interests in wireless networks, mobile computing, cloud computing,social networks, network security and quantum cryptography. Dr. Zhu had published more than 50 international journal papers (SCI or EI journals) on the above research fields.

Yifeng Zhang he is an undergraduate from Shenyang Normal University, major in information security management. In the four years of college, after completing her studies, he enjoys reading the book related to this major. Under the guidance of the teacher, he has published four articles in EI journals.