Journal of Information Hiding and Multimedia Signal Processing ©2016 ISSN 2073-4212
Ubiquitous International Volume **7**, Number **2**, March **2016**

A Password-Authenticated Key Exchange Scheme Using Chaotic Maps towards Multiple Servers to Server

Hongfeng Zhu, Tianhua Liu, and Yu Xia

Software College, Shenyang Normal University
no.253, HuangHe Bei Street, HuangGu District, Shenyang, P.C 110034 - China
zhuhongfeng1978@163.com; liutianhua@sina.com; 876526606@qq.com

Received December, 2014; revised October, 2015

Abstract. *With rapid changes in the modern communication environment such as wireless mesh networks and cloud storing, it is necessary to put forward a kind more flexible and general architecture to adapt it. But the overwhelming majority of password-authenticated key agreement protocols using chaotic maps are based on three architectures (client/server, two clients/server and multi-server) and four security models (heuristic security, random oracle, ideal cipher and standard model). Moreover, most of the key exchange schemes adopting chaotic maps are usually by symmetric cryptography for exchanging some information. This will lead to a high calculated amount. Therefore, the paper will wipe out the symmetric cryptography, and only use chaotic maps, a secure pseudo-random function to construct a provable secure password authenticated key agreement protocol towards multiple servers to server architecture in the standard model. The new protocol resists dictionary attacks mounted by either passive or active network intruders, allowing, in principle, even weak password phrases to be used safely. It also offers perfect forward secrecy, privacy protection and some others security attributes. Finally, we give the security proof in the standard model and the efficiency analysis of our proposed scheme.*

Keywords: Key exchange, Mutual authentication, Chaotic maps, Multiple servers to server

1. **Introduction.** In 1998, Baptista [1] firstly connects cryptography with chaos theory. As a fundamental cryptographic primitive, key agreement protocol allows two or more parties to agree on shared keys which will be used to protect their later communication. Then, conbiming chaos theory and key agreement primitive, many authenticated key exchange (AKE) protocols [2-9] have been proposed. The literature [3] firstly proposed a new one-way authenticated key agreement scheme (OWAKE) based on chaotic maps with multi-server architecture. The OWAKE scheme is widely used to no need for mutual authentication environment on Internet, such as readers-to-journalists model and patient-to-expert model. Using the chaotic maps, the literature [4] firstly proposed a new multiple servers to server architecture (MSTSA) to solve the problems caused by centralized architecture, such as multi-server architecture with the registration center (RC). The core ideas of the proposed scheme are the symmetry (or called peer to peer) in the servers side and the transparency for the clients side. In brief, based on chaotic maps, there were many AKE protocols from functionality aspect, or from efficiency aspect, or from security aspect, or from architecture aspect to improve the AKE protocols.

Recently, Multi-server authenticated key agreement (MSAKA) architecture is more popular among the AKE protocols which aim to register at the registration center for log in other servers without register repeatedly. MSAKA protocols mainly want to solve the problems in a traditional single server with authentication schemes [11-14] which lead to the fact that user has to register to different servers separately. On a macro level MSAKA protocols can be divided into three phases in chronological order:

(1) the creative phase: The pioneer work in the field was proposed by Li et al. [15] in 2001. However, Lin et al. [16] pointed out that Li et al.s scheme takes long time to train neural networks and an improved scheme based on ElGamal digital signature and geometric properties on the Euclidean plane has also been given.

(2) the development phase: the main work in this phase is amended repeatedly. For example, Tsai [17] also proposed an efficient multi-server authentication scheme based on one-way hash function without a verification table. Because Tsais scheme only uses the nonce and one-way hash function, the problems associated with the cost of computation can be avoided in the distributed network environment. However, some researchers [18] pointed out that Tsais scheme is also vulnerable to server spoofing attacks by an insider server and privileged insider attacks, and does not provide forward secrecy.

(3) the diversification phase: the research emphasis shifts to functionality. Therefore, identity-based MSAKA protocols, based on bilinear pairings or elliptic curve cryptosystem (ECC) MSAKA protocols, dynamic identity-based MSAKA protocols and other MSAKA protocols came up recently[18-20].

Based on the chaotic maps, we believe MSAKA protocols is not a general solution because only one centralized registration center cannot handle so complex network environment. So based on our previous studies [4], we believe that we should design an AKE protocol in a more general architecture. So we propose the first towards mltiple servers to server architecture key exchange protocol using chaotic maps in standard model.

The rest of the paper is organized as follows: Some preliminaries are given in Section 2. Next, a novel chaotic maps problem is described in Section 3. Then, the non-interactive twin chaotic maps-key exchange protocol is given in Section 4. The Security of our proposed protocol is given in Section 5. The efficiency analysis of our proposed protocol and some feasible applications are given in Section 6. This paper is finally concluded in Section 7.

2. **Preliminaries.**

2.1. **Pseudo-random function ensembles.** If a function ensemble $F = \{F_n\}_{n \in N}$ is pseudo-random [21], then for every probabilistic polynomial oracle A and all large enough n, we have that: $Adv^F(A) = |\Pr[A^{F_n}(1^n) = 1] - \Pr[A^{G_n}(1^n) = 1]| < \varepsilon(n)$,
where $G = \{G_n\}_{n \in N}$ is a uniformly distributed function ensemble, $\varepsilon(n)$ is a negligible function, $Adv^F = \max_A\{Adv^F(A)\}$ denotes all oracle A, and $Adv^F(A)$ represents the accessible maximum.

2.2. **Definition and hard problems of Chebyshev chaotic maps.** Let n be an integer and let x be a variable with the interval $[-1, 1]$. The Chebyshev polynomial [9] $T_n(x) : [-1, 1] \to [-1, 1]$ is defined as $T_n(x) = \cos(ncos^{-1}(x))$. Chebyshev polynomial map $T_n : R \to R$ of degree n is defined using the following recurrent relation:
$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x)$, where $n \geq 2$, $T_0(x) = 1$, and $T_1(x) = x$.
The first few Chebyshev polynomials are:
$T_2(x) = 2x^2 - 1$, $T_3(x) = 4x^3 - 3x$, $T_4(x) = 8x^4 - 8x^2 + 1$, ...
One of the most important properties is that Chebyshev polynomials are the so-called semi-group property which establishes that $T_r(T_s(x)) = T_{rs}(x)$.

An immediate consequence of this property is that Chebyshev polynomials commute under composition $T_r(T_s(x)) = T_s(T_r(x))$.
In order to enhance the security, Zhang [10] proved that semi-group property holds for Chebyshev polynomials defined on interval $(-\infty, +\infty)$. The enhanced Chebyshev polynomials are used in the proposed protocol: $T_n(x) = (2xT_{n-1}(x) - T_{n-2}(x))(\mathrm{mod} N)$, where $n \geq 2$, $x \in (-\infty, +\infty)$, and N is a large prime number. Obviously,
$T_{rs}(x) = T_r(T_s(x)) = T_s(T_r(x))$.

Definition 2.1. *(Semi-group property) Semi-group property of Chebyshev polynomials:* $T_{rs}(x) = T_r(T_s(x)) = \cos(r\cos^{-1}(s\cos^{-1}(x))) = \cos(rs\cos^{-1}(x)) = T_s(T_r(x)) = T_{sr}(x)$, *where r and s are positive ingeger and $x \in [-1, 1]$.*

Definition 2.2. *(Chaotic Maps-Based Discrete Logarithm (CDL) problem)*
Given x and y, it is intractable to find the integer s, such that $T_s(x) = y$. The probability that a polynomial time-bounded algorithm A can solve the CDL problem is defined as $Adv_A^{CDL}(p) = \Pr[A(x, y) = r : r \in Z_p^*, y = T_r(x) \bmod p]$.

Definition 2.3. *(CDL assumption) For any probabilistic polynomial time-bounded algorithm A, $Adv_A^{CDL}(p)$ is negligible, that is, $Adv_A^{CDL}(p) \leq \varepsilon$, for some negligible function ε.*

Definition 2.4. *(Chaotic Maps-Based Diffie-Hellman (CDH) problem)*
Given x,$T_r(x)$,and $T_s(x)$, it is intractable to find $T_{rs}(x)$.The probability that a polynomial time-bounded algorithm A can solve the CDH problem is defined as
$Adv_A^{CDH}(p) = \Pr[A(x, T_r(x) \bmod p, T_s(x) \bmod p) = T_{rs}(x) \bmod p : r, s \in Z_p^*]$.

Definition 2.5. *(CDH assumption) For any probabilistic polynomial time-bounded algorithm A, $Adv_A^{CDH}(p)$ is negligible, that is, $Adv_A^{CDH}(p) \leq \varepsilon$, for some negligible function ε.*

2.3. **Practical Environment.** The literature [4] firstly proposed a new multiple servers to server architecture (MSTSA), and now we set a prototype example in practical environment. denote the five rounds in Fig.1 respectively. We assume Alice wants to establish a session key with ServerB for getting the service of ServerB. So the initiator Alice broadcasts (A, ServerA, ServerB) in . Because Alice have already registered on ServerA, ServerA can use registered verifiers and ephemeral random numbers to authenticate Alice for helping ServerB in . In ServerA and ServerB will deliver the sensitive information to each other with Chaotic maps cryptosystem after authenticating each other. At the same time, ServerB will compute the session key with Alice after authenticating Alice and ServerA. In , ServerA sends sensitive information to Alice and finally Alice use sensitive information and the her own secret ephemeral random number to compute the session key with ServerB. (The same way for other servers and users)

3. The Proposed Protocol.

In this section, under the multiple servers to server architecture, a chaotic maps-based password authentication key agreement scheme is proposed which consists of three phases: registration phase, authentication key agreement phase and password update phase.

3.1. Notations.

In this section, any server i has its identity ID_{S_i} and public key $(x, T_{K_i}(x))$ and a secret key K_i based on Chebyshev chaotic maps, a secure one-way hash function $H(\cdot)$ and a pseudo-random function F. The concrete notations used hereafter are shown in Table1.

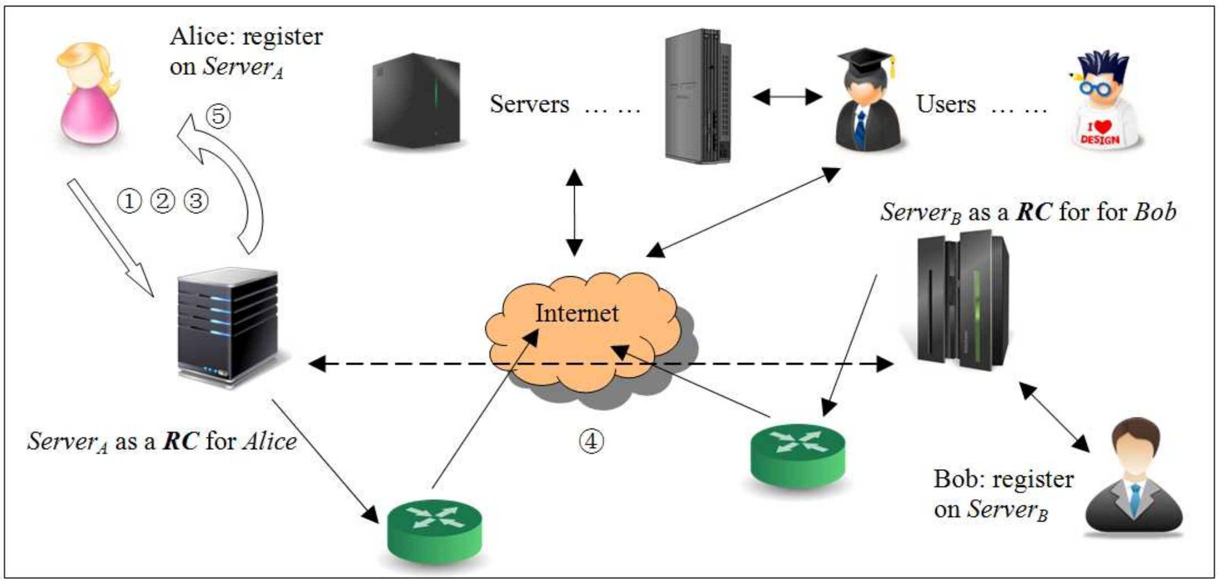

Alice: register on $Server_A$
⑤
① ② ③
Servers … …
Users … …
$Server_B$ as a **RC** for for *Bob*
Internet
$Server_A$ as a **RC** for *Alice*
④
Bob: register on $Server_B$

FIGURE 1. An example for practical environment of multiple servers to server architecture

TABLE 1. Notations

Symbol	Definition
$ID_A, ID_{Session}$	the identity of Alice and the session respectively
S_i, ID_{S_i}	The *i*th server，the identity of the *i*th server, respectively
TID_i	The temporary identity of party *i*
t, a, S_a, S_b, S_{aa}	nonces
$(x, T_K(x))$	public key based on Chebyshev chaotic maps
K	secret key based on Chebyshev chaotic maps
H	A secure one-way hash function
F	pseudo-random function
$\|\|$	concatenation operation

3.2. **Registration phase.** Concerning the fact that the proposed scheme mainly relies on the design of Chebyshev chaotic maps-based in multiple servers to server architecture, it is assumed that Alice can register at the serverA by secure channel and view the serverA as her own registration center to login on other servers for some services. The same assumption can be set up for users. Fig.2 illustrates the users registration phase.

Step 1. When a user Alice wants to be a new legal user, she chooses her identity ID_A and password PW_A. Then Alice computes $HPW_A = H\left(ID_A||PW_A||T_{K_A}\left(x\right)\right)$ and sends $\{ID_A, HPW_A\}$ to the server via a secure channel.

Step 2. Upon receiving $\{ID_A, HPW_A\}$ from the Alice, the serverA stores $\{ID_A, HPW_A\}$ in a secure way.

3.3. **Preprocessing of TID.** For simplicity, we construct a function to produce For simplicity, we construct a function to produce TID_i, the temporary identity of party i for clients or servers. Without loss of generality, we assume party i sends a TID_i to party j using $(x, T_{K_j}(x))$ for covering ID_i but only party j can recover the ID_i.

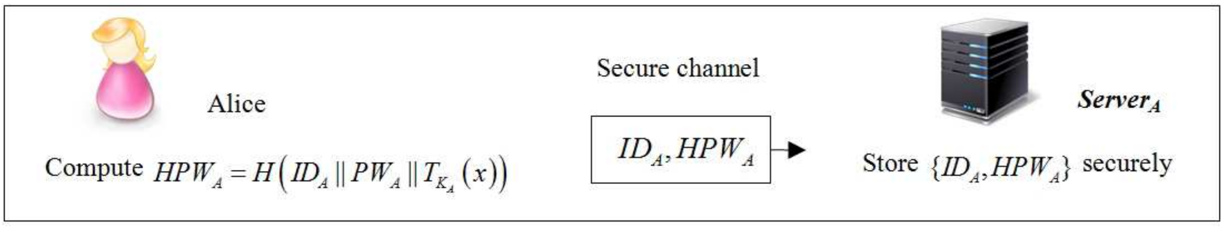

Alice
Secure channel
$Server_A$
Compute $HPW_A = H(ID_A \| PW_A \| T_{K_A}(x))$
ID_A, HPW_A
Store $\{ID_A, HPW_A\}$ securely

FIGURE 2. Server or a authenticated expert registration phase

The party i selects a large and random integer t, and computes $T_t(x)$, $C_t = T_tT_{K_j}(x)ID_i$, $FTID_i = F_{ID_i}(C_t\|T_t(x))$. Then the party i sends $\{T_t(x), C_t, FTID_i\}$ to the party j. After receiving the message $\{T_t(x), C_t, FTID_i\}$ from the party i, the party j will use $T_t(x)$ and his own secret key K_j to recover $ID_i = C_t/T_{K_j}T_t(x) = C_t/T_tT_{K_j}(x)$. Then the party j check if $F_{ID_i}(C_t\|T_t(x)) \overset{?}{=} FTID_i$. If above equation holds, the party j deems the ID_i is legal identity. Otherwise, the party j terminates the session.

3.4. **Authenticated key agreement phase.**

Remark 3.1. *We omit the production of temporary identity TID_i. The concrete process can be found in the Section 3.3.*

This concrete process is presented in the following Fig. 3.

Step 1. If Alice wishes to consult some personal issues establish with ServerB in a secure way, but Alice has not register at ServerB. So in our multiple servers to server architecture, Alice need not register at ServerB and she just uses her account at the ServerA to login in ServerB. Alice will choose a large and random a. Then the device of Alice will compute $T_a(x)$, $C_{A_1} = T_aT_{HPW_A}T_{K_A}(x)$, and $Mac_{AS} = F_{T_aT_{K_A}(x)}(ID_{Session}\|C_{A_1})$. After that, Alice sends $\{TID_A, TID_{S_B}, C_{A_1}, Mac_{AS}\}$ to ServerA where she registers on.

Step 2. After receiving the message $\{TID_A, TID_{S_B}, C_{A_1}, Mac_{AS}\}$ from Alice, ServerA will do the following tasks: (1) ServerA uses HPW_A to compute $T_a(x) = C_{A_1}/T_{HPW_A}T_{K_A}(x)$. (2) ServerA examines whether $Mac_{AS} = F_{T_aT_{K_A}(x)}(ID_{Session}\|C_{A_1})$ is valid in terms of the $(ID_{Session}\|C_{A_1})$. (3) ServerA selects a large and random integer S_a to compute $T_{S_a}(x)$, $C_{A_2} = T_aT_{S_a}T_{K_B}(x)$, $Mac_{SAB} = F_{T_aT_{K_B}(x)}(ID_{Session}\|C_{A_2})$
and sends $\{TID_A, TID_{S_A}, C_{A_2}, T_{S_a}(x), Mac_{SAB}\}$ to ServerB.

Step 3. After receiving the message $\{TID_A, TID_B, C_{A_2}, T_{S_a}(x), Mac_{SAB}\}$ from ServerA, ServerB will uses K_B to compute $T_a(x) = C_{A_2}/T_{S_a}T_{K_B}(x) = C_{A_2}/T_{K_B}T_{S_a}(x)$. Then ServerB examines whether $Mac_{SAB} = F_{T_aT_{K_B}(x)}(ID_{Session}\|C_{A_2})$ is valid in terms of the $(ID_{Session}\|C_{A_2})$. ServerB selects a large and random integer S_b and computes $T_{S_{bb}}(x)$, $C_{A_3} = T_{S_{bb}}T_{HPW_B}T_a(x)$, $Mac_{SB} = F_{T_aT_b(x)}(ID_{Session}\|C_{A_3})$ and sends $\{TID_A, TID_{S_B}, C_{A_3}, T_{S_b}(x), Mac_{SBA}\}$ to ServerA. And then ServerB computes the session key is $SK = F_{T_{S_b}T_a(x)}(1)$.

Step 4. After receiving the message $\{TID_A, TID_{S_B}, C_{A_3}, T_{S_b}(x), Mac_{SBA}\}$, ServerA uses K_A to compute $C'_{A_3} = T_{K_A}T_{S_b}(x) = T_{S_b}T_{K_A}(x) = C_{A_3}$. Then ServerA examines whether $Mac_{SBA} = F_{T_{K_A}T_{S_b}}(ID_{Session}\|C_{A_3})$ is valid in terms of the $(ID_{Session}\|C_{A_3})$. If holds, ServerA selects a large and random integer S_{aa} and computes $T_{S_{aa}}(x)$, $C_{A_4} = T_{S_{aa}}T_{HPW_A}T_{S_b}(x)$, $Mac_{SA} = F_{T_{S_{aa}}T_{HPW_A}}(ID_{Session}\|C_{A_4})$
and sends $\{ID_A, ID_{S_B}, C_{A_4}, T_{S_{aa}}(x), Mac_{SA}\}$ to Alice.

Step 5. After receiving the message $\{TID_A, TID_{S_B}, C_{A_4}, T_{S_{aa}}(x), Mac_{SA}\}$ from ServerA, Alice will uses HPW_A to compute $T_S(x) = C_{A_4}/T_{S_{aa}}T_{HPW_A}(x) = C_{A_4}/T_{HPW_A}T_{S_{aa}}(x)$.

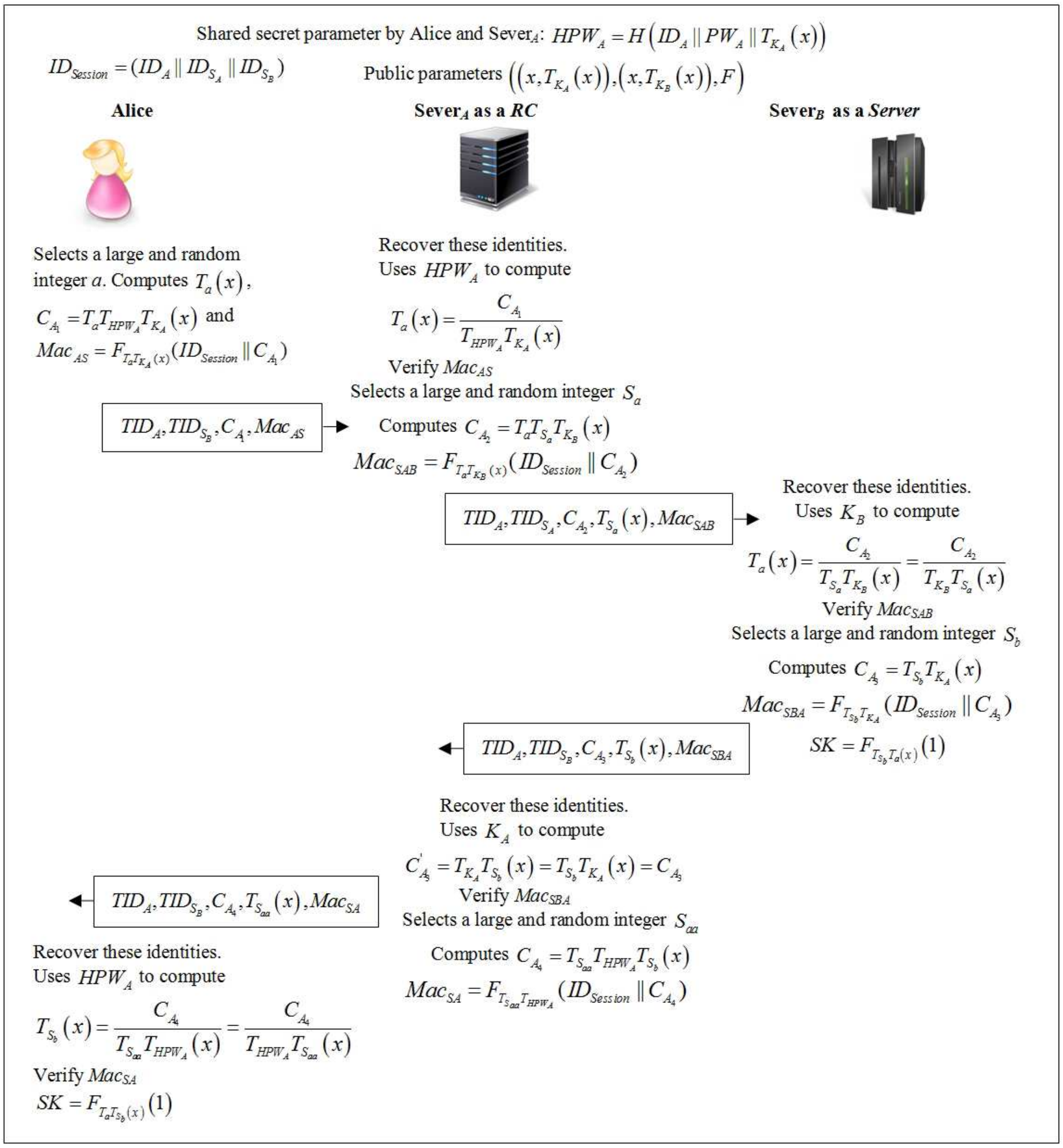

Shared secret parameter by Alice and Sever$_A$: $HPW_A = H(ID_A \| PW_A \| T_{K_A}(x))$
$ID_{Session} = (ID_A \| ID_{S_A} \| ID_{S_B})$
Public parameters $((x, T_{K_A}(x)), (x, T_{K_B}(x)), F)$
Alice
Sever$_A$ as a *RC*
Sever$_B$ as a *Server*
Selects a large and random integer a. Computes $T_a(x)$, $C_{A_1} = T_a T_{HPW_A} T_{K_A}(x)$ and $Mac_{AS} = F_{T_a T_{K_A}(x)}(ID_{Session} \| C_{A_1})$
$TID_A, TID_{S_B}, C_A, Mac_{AS}$
Recover these identities. Uses HPW_A to compute $T_a(x) = \frac{C_{A_1}}{T_{HPW_A} T_{K_A}(x)}$
Verify Mac_{AS}
Selects a large and random integer S_a
Computes $C_{A_2} = T_a T_{S_a} T_{K_B}(x)$
$Mac_{SAB} = F_{T_a T_{K_B}(x)}(ID_{Session} \| C_{A_2})$
$TID_A, TID_{S_A}, C_{A_2}, T_{S_a}(x), Mac_{SAB}$
Recover these identities. Uses K_B to compute $T_a(x) = \frac{C_{A_2}}{T_{S_a} T_{K_B}(x)} = \frac{C_{A_2}}{T_{K_B} T_{S_a}(x)}$
Verify Mac_{SAB}
Selects a large and random integer S_b
Computes $C_{A_3} = T_{S_b} T_{K_A}(x)$
$Mac_{SBA} = F_{T_{S_b} T_{K_A}}(ID_{Session} \| C_{A_3})$
$SK = F_{T_{S_b} T_a(x)}(1)$
$TID_A, TID_{S_B}, C_{A_3}, T_{S_b}(x), Mac_{SBA}$
Recover these identities. Uses K_A to compute $C'_{A_3} = T_{K_A} T_{S_b}(x) = T_{S_b} T_{K_A}(x) = C_{A_3}$
Verify Mac_{SBA}
Selects a large and random integer S_{aa}
Computes $C_{A_4} = T_{S_{aa}} T_{HPW_A} T_{S_b}(x)$
$Mac_{SA} = F_{T_{S_{aa}} T_{HPW_A}}(ID_{Session} \| C_{A_4})$
$TID_A, TID_{S_B}, C_{A_4}, T_{S_{aa}}(x), Mac_{SA}$
Recover these identities. Uses HPW_A to compute $T_{S_b}(x) = \frac{C_{A_4}}{T_{S_{aa}} T_{HPW_A}(x)} = \frac{C_{A_4}}{T_{HPW_A} T_{S_{aa}}(x)}$
Verify Mac_{SA}
$SK = F_{T_a T_{S_b}(x)}(1)$

FIGURE 3. Authenticated key agreement phase

Then Alice examines whether $Mac_{SA} = F_{T_{HPW_A} T_{S_{aa}}(x)}(ID_{Session} || C_{A_2})$ is valid in terms of the $(ID_{Session} || C_{A_4})$. If holds, Alice computes the session key is $SK = F_{T_a T_{S_b}(x)}(1)$.

If any authenticated process does not pass, the protocol will be terminated immediately.

3.5. **Password update phase.** This concrete process is presented in the following Fig. 4.

Step 1. If Alice wishes to update her password with ServerA, Alice will choose a new memorable password PW_A^{ε}. Then the device of Alice will compute $HPW_A^{\varepsilon} = H(ID_A || PW_A^{\varepsilon} || T_{K_A}(x))$, $T_{HPW_A^{\varepsilon}}(x)$, $C_{A_1} = T_{HPW_A^{\varepsilon}}(x) T_{HPW_A} T_{K_A}(x)$ and $Mac_{AS} = F_{T_{HPW_A^{\varepsilon}} T_{K_A}(x)}(ID_A || ID_{S_A} || C_{A_1})$. After that, Alice sends $\{TID_A, C_{A_1}, Mac_{AS}\}$ to ServerA where she registers on.

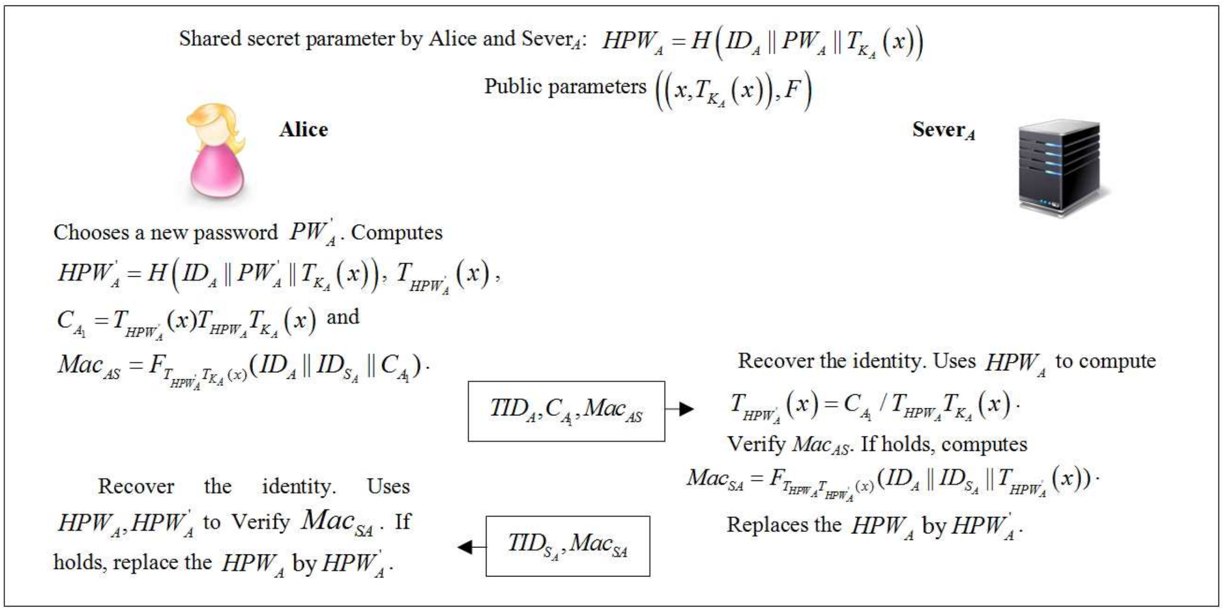

Shared secret parameter by Alice and Sever$_A$: $HPW_A = H(ID_A \| PW_A \| T_{K_A}(x))$
Public parameters $((x, T_{K_A}(x)), F)$
Alice
Sever$_A$
Chooses a new password $PW_A^{'}$. Computes $HPW_A^{'} = H(ID_A \| PW_A^{'} \| T_{K_A}(x))$, $T_{HPW_A^{'}}(x)$, $C_{A_1} = T_{HPW_A^{'}}(x) T_{HPW_A} T_{K_A}(x)$ and $Mac_{AS} = F_{T_{HPW_A^{'}} T_{K_A}(x)}(ID_A \| ID_{S_A} \| C_A)$.
TID_A, C_A, Mac_{AS}
Recover the identity. Uses HPW_A to compute $T_{HPW_A^{'}}(x) = C_{A_1} / T_{HPW_A} T_{K_A}(x)$. Verify Mac_{AS}. If holds, computes $Mac_{SA} = F_{T_{HPW_A} T_{HPW_A^{'}}(x)}(ID_A \| ID_{S_A} \| T_{HPW_A^{'}}(x))$. Replaces the HPW_A by $HPW_A^{'}$.
Recover the identity. Uses $HPW_A, HPW_A^{'}$ to Verify Mac_{SA}. If holds, replace the HPW_A by $HPW_A^{'}$.
TID_{S_A}, Mac_{SA}

FIGURE 4. Password update phase

Step 2. After receiving the message $\{TID_A, C_{A_1}, Mac_{AS}\}$ from Alice, ServerA will do the following tasks: (1) ServerA uses HPW_A to compute $T_{HPW_A^{\varepsilon}}(x) = C_{A_1}/T_{HPW_A}T_{K_A}(x)$. (2) ServerA examines whether $Mac_{AS} = F_{T_{HPW_A^{\varepsilon}}T_{K_A}(x)}(ID_A||ID_{S_A}||C_{A_1})$ is valid in terms of the $(ID_A||ID_{S_A}||C_{A_1})$. (3) If holds, ServerA computes $Mac_{SA} = F_{T_{HPW_A}T_{HPW_A^{\varepsilon}}(x)}(ID_A||ID_{S_A}||T_a(x))$ and sends $\{ID_A, ID_{S_A}, Mac_{SA}\}$ to Alice. Replaces the HPW_AbyHPW_A^{ε}.

Step 3. After receiving the message $\{TID_{S_A}, Mac_{SA}\}$ from ServerA, Alice will uses $HPW_A, HPW_A^{\varepsilon}$ to compute $Mac_{SA}^{\varepsilon} = F_{T_{HPW_A}T_{HPW_A^{\varepsilon}}(x)}(ID_A||ID_{S_A}||T_{HPW_A^{\varepsilon}}(x))$ to verify Mac_{SA}. If holds, Alice replaces the HPW_AbyHPW_A^{ε}.

4. **Security Consideration.** The section a theorem concerning the semantic security of our proposed protocol is given.

4.1. **Security Model.** We recall the protocol syntax and communication model [22-24]. The basic descriptions and some queries are shown in Table 2.

4.2. **Security Proof.**

Theorem 4.1. *Let Γ be a PAKE protocol towards multiple servers to server architecture described in Fig.3. Let $F : \{0,1\}^n \to \{0,1\}^{l(n)}$ be a pseudo-random function ensembles. Because the DDH assumption holds in enhanced Chebyshev chaotic maps, then*

$$Adv_{x,T_u,F}^{MSTSA-PAKE}(t,R) \le \frac{4q_e^2+4q_s^2+(q_e+q_s)^2}{2N_1} + 2(q_e+q_s)Adv^F + 2(\min\{q_e,q_r\}+ \min\{q_s,q_r\})Adv^F + 2(q_e+q_s)Adv_{x,T_u}^{DDH} + \frac{q_s}{2^{n-1}} + \frac{(q_e+q_s)^2}{N_1}\frac{q_s}{2N},$$

where n is a safe parameter, $l()$ is a function that can be computed in polynomial time. $N1$ is a large prime number, $u, T_u(x)$ are the private and public keys of the server, q_e, q_r, q_s represent the maximum number of Execute and Test that the adversary can inquire, and queries from Send-Client and Send-Server, N is the password dictionary Ds size, Adv_{x,T_u}^{DDH} represents the probability of breaking the DDH hypothesis, and Adv^F denotes the probability of breaking the pseudo-random function ensembles.

Table 2. Descriptions the model and the queries

Symbol	**Definition**
parties $P_1,...P_n$ or $(C_1,...C_n, S_1,...S_n)$	Modeled by probabilistic Turing machines. Two non-empty sets: User, the set of all clients, and Server, the set of trusted servers constitute the participants in our MSTSA-PAKE protocol.
Adversary Λ	A probabilistic Turing machine which controls all communication, with the exception that the adversary cannot inject or modify messages (except for messages from corrupted parties or sessions), and any message may be delivered at most once.
Sessiosns matching	If the outgoing messages of one are the incoming messages of the other
$\Pi^i_{U_1}, pid^i_{U_1}, sid^i_{U_1}$ $\Pi^j_{U_2}, pid^j_{U_2}, sid^j_{U_2}$	Denote participant U_1's instance i, who is involved with a partner participant U_2 in a session. $\Pi^i_{U_1}$ has the partner identification $pid^i_{U_1}$ and the session identification $sid^i_{U_1}$. The same means for $\Pi^j_{U_2}, pid^j_{U_2}, sid^j_{U_2}$.
Execute $(\Pi^i_{U_1}, S^i, S^j, \Pi^j_{U_2})$	This query returns the messages that were communicated in the course of an honest execution of the protocol among $\Pi^i_{U_1}, S^i, S^j, \Pi^j_{U_2}$.
Send-Client $(\Pi^i_{U_k}(k=1,2), m)$	This query returns the message that client instance $\Pi^i_{U_k}$, which would generate upon receipt of message m.
Send-Server $(S^k(k=1,2), m)$	This query returns the message that server instance S^k would generate upon receipt of message m. When receiving a fabricated message by an adversary, the server instance S^k responds in the manner prescribed by the protocol.
Corrupt $(U_k(k=1,2))$	This query returns the session key of the client instance $U_k(k=1,2)$.
Reveal $(\Pi^i_{U_k}(k=1,2))$	This query returns the password and the states of all instances of $U_k(k=1,2)$ only when it is defined.
Test $(\Pi^i_{U_k}(k=1,2))$	This query allows the adversary to be issued at any stage to a completed, fresh, unexpired session. A bit b is then picked randomly. If b = 0, the test oracle reveals the session key, and if b = 1, it generates a random value in the key space. The adversary Λ can then continue to issue queries as desired, with the exception that it cannot expose the test session.
Partnering	We say two instances $\Pi^i_{U_1}$ and $\Pi^j_{U_2}$ are partners iff: (a) They are successfully accepted. (b) $sid^i_{U_1} = sid^j_{U_2}$. (c) *pid* for $\Pi^i_{U_1}$ is $\Pi^j_{U_2}$ and vice versa. (d) No instance other than $\Pi^i_{U_1}$ and $\Pi^j_{U_2}$ accepts with a *pid* equal to $\Pi^i_{U_1}$ or $\Pi^j_{U_2}$.
Freshness	Let $\Pi^i_{U_1,U_2,S_1,S_2}$ be a completed session by a party U_1 with some other party U_2, and $\Pi^j_{U_2,U_1,S_2,S_1}$ be the matching session to $\Pi^i_{U_1,U_2,S_1,S_2}$. We say that the session $\Pi^i_{U_1,U_2,S_1,S_2}$ is fresh if U_1 and U_2 in session $\Pi^i_{U_1,U_2,S_1,S_2}$ and the matching session $\Pi^j_{U_2,U_1,S_2,S_1}$ are honest and the following conditions hold: (a) $\Pi^i_{U_1,U_2,S_1,S_2}$ has accepted the request to establish a session key. (b) $\Pi^i_{U_1,U_2,S_1,S_2}$ has not been revealed. (c) No matching conversation $\Pi^j_{U_2,U_1,S_2,S_1}$ of $\Pi^i_{U_1,U_2,S_1,S_2}$ has been revealed. (d) U_2, S have not been corrupted. (e) The adversary asks neither Send-Client $(\Pi^i_{U_1}, m)$ nor Send-Client $(\Pi^j_{U_2}, m)$ query.

In order to make the security proof simple, we firstly point out the differences between the literature [23] and our proposed protocol. Then we give the differences between the literature [24] and our proposed protocol. Finally, we will get the theorem 4.1.

(1) The differences between the literature [23] and our proposed protocol.

Using enhanced Chebyshev chaotic maps to replace ElGamal encryption. To be specific, $g^{x_2}, rg^{x_1}, Zg^{x_1}$ and $g^{x_1}h^{x_2}$ in the literature [23] should be replaced by $T_{x_2}(x), rT_{x_1}(x), ZT_{x_1}(x)$ and $T_{x_1}(x)T_{x_2}(h)$, respectively.

The birthday paradox should be used to replace the probability of random events when the event collision occurs. According to the birthday paradox, the probability of collisions in output $T_n(x)$ is at most $q_s^2/2N_1$, where q_s denotes the maximum number of Send-Client and Send-Server queries.

According to the birthday paradox, the probability of collisions in output $T_n(x)$ is at most $(q_s+q_e)^2/2N_1$, where q_s denotes the maximum number of Send-Client and Send-Server queries, q_e denotes the maximum number of Execute queries. Hence, the probability of distinguishing Mac_{**} with random integers is $(q_s+q_e)^2/2N_1$.

(2) The differences between the literature [24] and our proposed protocol.

We convert the low entropy secret password PW to high entropy cryptography key by a one-way hash function $HPW_A = H\left(ID_A||PW_A||T_{K_A}(x)\right)$ which is more secure way than the literature [24] only stored password in the server database.

Our proposed protocol has one less Mac_{**} for each party, so there is must have one less $(q_s+q_e)^2/2N_1$. Our proposed protocol sets up in multiple servers to server architecture which has only one password with the RC server. That means one Send-Client query will test only one password in the same set. So in our protocol, when relating with N (N is the password dictionary Ds size), and it is must be multiplied by 1/2.

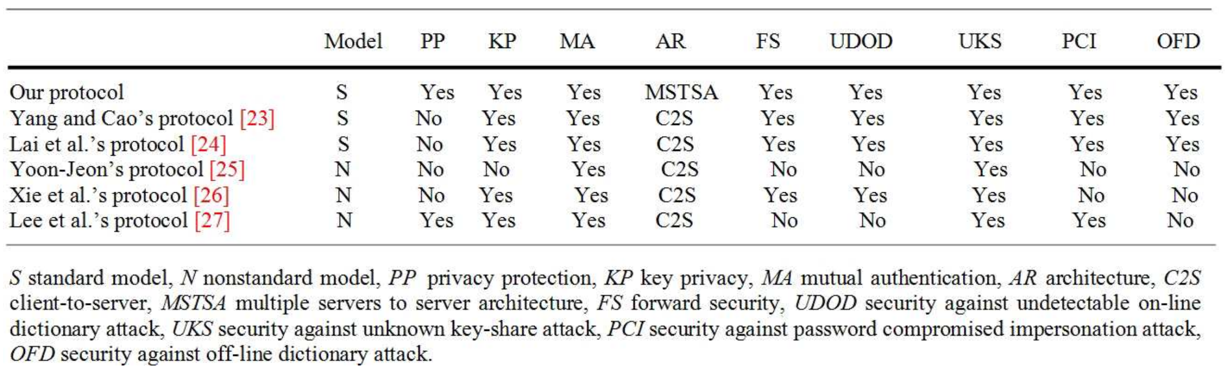
The detailed descriptions of these games and lemmas are analogous to those in literature [24], with the differences discussed above, and therefore, they are omitted.

Theorem 4.2. *Our proposed two-realm PAKE protocol ensures key privacy against the server based on the fact that DDH assumption holds in the enhanced Chebyshev chaotic maps and F is a secure pseudo-random function ensemble, and* $Adv_D^{k_p}(\Lambda_{k_p}) \le 4q_s Adv_{x,T_u}^{DDH} + 2q_e Adv^F$, *where qe and qs denote the maximum number of queries to the oracle Execute and Send-Client.*

The proof of Theorem 4.2 is similar to that of Theorem 5.2 in [23] and Theorem 3 in [24]. The difference between our proposed protocol and the literature [23] is that we just replace the enhanced Chebyshev chaotic map values with the ElGamal discrete logarithm values. The difference between our proposed protocol and the literature [24] is that our proposed protocol is designed in different realm with different password, so some changed details can be described in the section 4.2(2).

Next, from the Table 3, we can see that the proposed scheme can provide secure session key agreement, 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.

Table 3. Security comparison existing protocols for 3PAKE based on Chebyshev chaotic maps and our protocol



	Model	PP	KP	MA	AR	FS	UDOD	UKS	PCI	OFD
Our protocol	S	Yes	Yes	Yes	MSTSA	Yes	Yes	Yes	Yes	Yes
Yang and Cao's protocol [23]	S	No	Yes	Yes	C2S	Yes	Yes	Yes	Yes	Yes
Lai et al.'s protocol [24]	S	No	Yes	Yes	C2S	Yes	Yes	Yes	Yes	Yes
Yoon-Jeon's protocol [25]	N	No	No	Yes	C2S	No	No	Yes	No	No
Xie et al.'s protocol [26]	N	No	Yes	Yes	C2S	Yes	Yes	Yes	No	No
Lee et al.'s protocol [27]	N	Yes	Yes	Yes	C2S	No	No	Yes	Yes	No

S standard model, *N* nonstandard model, *PP* privacy protection, *KP* key privacy, *MA* mutual authentication, *AR* architecture, *C2S* client-to-server, *MSTSA* multiple servers to server architecture, *FS* forward security, *UDOD* security against undetectable on-line dictionary attack, *UKS* security against unknown key-share attack, *PCI* security against password compromised impersonation attack, *OFD* security against off-line dictionary attack.

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. In our proposed protocol, no time-consuming modular exponentiation and scalar multiplication on elliptic curves are needed. However, Wang [9] proposed several methods to solve the Chebyshev polynomial computation problem.

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 [28]. Moreover, the computational cost of XOR operation could be ignored when compared with other operations.

For simplicity, the literatures [16-19] in the different realms architecture, we omit the comparisons table detailedly. The reason is that our proposed protocol are mainly based on chaotic maps algorithms which is more efficient than the other algorighms, such as RSA and ECC, in the literatures [16-19].

Table 4 shows performance comparisons between our proposed scheme and the literature of [23-27] in three-party architecture with chaotic maps.

Table 4. Cost comparison existing protocols for 3PAKE based on Chebyshev chaotic maps and our protocol

	R	**RN**	**PKE**	**SKE**	***T***	***H***	***D***	***F***
The others paper		$(A/B/S)$	$(A/B/S)$	$(A/B/S)$	$(A/B/S)$	$(A/B/S)$	$(A/B/S)$	$(A/B/S)$
Our protocol		$(A/S_A/S_B)$	$(A/S_A/S_B)$	$(A/S_A/S_B)$	$(A/S_A/S_B)$	$(A/S_A/S_B)$	$(A/S_A/S_B)$	$(A/S_A/S_B)$
Our protocol	5	1/1/2	0/1/1	0/0/0	3/6/2	0/0/0	0/0/0	2/2/1
Yang and Cao's protocol [23]	4	2/2/3	0/0/1	0/0/1	0/0/0	0/0/0	0/0/0	4/4/2
Lai et al.'s protocol [24]	4	2/2/3	0/0/1	0/0/1	6/6/10	0/0/0	0/0/0	4/4/2
Yoon-Jeon's protocol [25]	5	2/1/0	2/2/0	1/1/1	2/2/0	2/0/2	1/1/2	0/0/0
Xie et al.'s protocol [26]	6	1/1/1	2/2/0	3/3/0	3/3/2	5/5/4	2/2/4	0/0/0
Lee et al.'s protocol [27]	5	1/1/1	2/2/0	4/4/0	3/3/2	4/4/7	0/0/0	0/0/0

R Round, *RN* Random number, *PKE* Public key encryption, *SKE* Secret key encryption. A: participant A, B: participant B, S: Single Server, S_A: $Server_A$ as ***RC***, S_B: $Server_B$, *T*, *D*, *H* and *F* represent the time for performing a Chebyshev polynomial computation, a symmetric encryption/decryption, a one-way hash function and pseudo-random function, respectively.

6. Conclusion.

In this paper, we conduct a comprehensive and general study of PAKE protocol over standard model using chaotic maps towards MSTSA. As far as we know, there is no general and extensible architecture about distributed network environment based on chaotic maps has been proposed. Through our exploration, we firstly clarify that the PAKE scheme using chaotic maps towards multiple servers to server architecture is more suitable for the real environment. Then, we proposed a suitable protocol that covers those goals and offered an efficient protocol that formally meets the proposed security definition. 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.

REFERENCES

[1] M. S. Baptista, Cryptography with chaos,*Physics Letters A*, vol. 240, no. 1, pp. 50-54, 1998.

[2] C. Lee, C. Li, and C. Hsu, A three-party password-based authenticated key exchange protocol with user anonymity using extended chaotic maps, *Nonlinear Dyn*,vol. 73, pp. 125-132, 2013.

[3] H. F. Zhu, Y. F. Zhang and Y. Zhang, A One-Way Authentication Key Agreement Scheme with User Anonymity Based on Chaotic maps towards Multi-Server Architecture, *Journal of Information Hiding and Multimedia Signal Processing*,vol. 6, no. 2, pp. 274-287, March 2015.

[4] H. F. Zhu, M. Jiang, X. Hao and Y. Zhang, Robust Biometrics-based Key Agreement Scheme with Smart Cards towards a New Architecture, *Journal of Information Hiding and Multimedia Signal Processing*,vol. 6, no. 1, pp. 81-98, January 2015.

[5] H. F. Zhu, X. Hao, Y. F. Zhang and M. Jiang, A biometrics-based multi-server key agreement scheme on chaotic maps cryptosystem, *Journal of Information Hiding and Multimedia Signal Processing*,vol. 6, no. 2, pp. 211-224, March 2015.

[6] L. R. Devaney, An Introduction to Chaotic Dynamical System, *Cummings Publishing Company Inc.*,The Benjammin, Menlo Park, 1986.

[7] J. C. Jiang, Y. H. Peng, Chaos of the Chebyshev polynomials,*Nat. Sci. J. Xiangtan Univ*,vol. 19, no. 3, pp. 3739, 1996.

[8] L. Zhang, Cryptanalysis of the public key encryption based on multiple chaotic systems,*Chaos Solitons Fractals*,vol. 37, no. 3,pp. 669674, 2008.

[9] X Wang, and J. Zhao, An improved key agreement protocol based on chaos,*Commun. Nonlinear Sci. Numer. Simul*,vol. 15, pp. 4052-4057, 2010.

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

[11] Y. Y. Lee, and Y. C. Chiu, Improved remote authentication scheme with smart card,*Comput Stand Interfaces*, vol. 27, no. 2, pp. 177180, 2005.

[12] E. J. Yoon, E. K. Ryu, and K. Yoo, An improvement of HwangLeeTangs simple remote user authentication scheme,*Comput Secur*,vol. 24, no. 1, pp. 5056, 2005.

[13] Y. F. Chang, C.C. Chang, and Y. W. Su, A secure improvement on the user-friendly remote authentication scheme with no time concurrency mechanism,*In: Proc of 20th international conference on advanced information networking and applications (AINA06), IEEE Computer Society*,pp. 741745, 2006.

[14] M. K. Khan, and J Zhang, Improving the security of a flexible biometrics remote user authentication scheme, *Comput Stand Interfaces*,vol. 29, no. 1, pp. 8285, 2007.

[15] L. H, Li, I. C. Lin, and M. S. Hwang, A remote password authentication scheme for multi-server architecture using neural networks, *IEEE Transactions on Neural Networks*,vol.12, no. 6, pp. 14981504, 2001.

[16] I. C. Lin, M.S. Hwang, and L. H. Li, A new remote user authentication scheme for multi-server architecture,*Future Generation Computer Systems*,vol.19, no.1, pp. 1322, 2003.

[17] J. L. Tsai, Efficient multi-server authentication scheme based on one-way hash function without verification table, *Comput Secur*,vol. 27, no. 34, pp. 115121,2008.

[18] S. P. Ravi, C. D. Jaidhar, and T. Shashikala, Robust Smart Card Authentication Scheme for Multi-server Architecture, *Wireless Pers Commun*,vol. 72, pp. 729745, 2013.

[19] B. Wang, and M. Ma, A smart card based efficient and secured multi-server authentication scheme, *Wireless Personal Communications*,vol. 68, Issue 2, pp. 361-378, 2013.

[20] E. J. Yoon, and K.-Y. Yoo, Robust biometrics-based multi-server authentication with key agreement scheme for smart cards on elliptic curve cryptosystem, *J Supercomput*,vol. 63, pp. 235255, 2013.

[21] V. Shoup, Sequences of games: a tool for taming complexity in security proofs. report 2004/332,*International Association for Cryptographic Research (IACR)*,2004.

[22] C. T. Ran, and K. Hugo, Analysis of key-exchange protocols and their use for building secure channels,*EUROCRYPT*,vol. 2045 of Lecture Notes in Computer Science, pp. 453-474, 2001.

[23] J. H. Yang, , T. J. Cao, Provably secure three-party password authenticated key exchange protocol in the standard model, *J. Syst. Softw*,vol. 85, pp. 340350, 2012.

[24] L. Hong, A. Mehmet, Orgun, Jinghua Xiao, et al, Provably secure three-party key agreement protocol using Chebyshev chaotic maps in the standard model, *Nonlinear Dyn*,vol. 77, pp. 14271439, 2014.

[25] E. J. Yoon, I. S. Jeon, An efficient and secure Diffie-Hellman key agreement protocol based on Chebyshev chaotic map.*Commun. Nonlinear Sci. Numer. Simul.*,16, 23832389 (2011).

[26] Q. Xie, J. M. Zhao, X. Y. Yu, Chaotic maps-based threeparty password-authenticated key agreement scheme.*Nonlinear Dyn.*,74, 10211027 (2013).

[27] C. C. Lee, C. T. Li, C. W. Hsu, A three-party passwordbased authenticated key exchange protocol with user anonymity using extended chaotic maps.*Nonlinear Dyn.*, 73, 125132 (2013).

[28] L. Kocarev, and S. Lian, Chaos-Based Cryptography: Theory, Algorithms and Applications, pp. 5354, 2011.