# Two-phase Commit with Security Services: Using Distinctive Proofs to Relieve Fragile Communications

Yang Sun[1], Xueshuai Feng[2], Hongfeng Zhu[3]
*(Corresponding author: Hongfeng Zhu)*

Software College, Shenyang Normal University
No.253, HuangHe Bei Street, HuangGu District, Shenyang 110034, P.R. China
(Email: zhuhongfeng1978@163.com)
*(Received July 2, 2015; revised and accepted Apr. 17 & Apr. 25, 2016)*

## Abstract

Inspired by stand-alone authentication, which can authenticate users when the connection to the central server is down, we present concepts called local proof and delayed proof that can adapt to two scenes when the authentication server is down: the former can solve self-authenticated to make local applications running without online authentication; the latter can solve two nodes to produce a session key for doing some transactions, but firstly they must exchange the delayed proof to prevent the fraud actions, specially, if the transaction is taking on the important process (such as contract signing or cash transaction), they must wait the authentication server is online. The key idea of our scheme is to improve the efficiency, and anyone can make effective use of the time to negotiate or do some unimportant things during the authentication server is down. Next, we propose a novel Chaotic Maps-based scheme against fragile communications, named CMFC, aiming to bypass the crashed authentication server temporarily for kinds of applications running. For important applications, we adopt the idea of two-phase commit protocol in our scheme: (1) the unavailable authentication server case, in which the CMFC can self-authenticated or compute a provisional delayed proof and a session key for two-party communicating. (2) the available authentication server case, in which, based on the phase (1) and the authentication server's verification, the two-party decides whether to commit (only if all have voted "Yes") or abort the transaction (otherwise). Finally, we give the formal security proof about our scheme with BAN logic and efficiency analysis.

*Keywords: Ban logic; chaotic maps, delayed proof; stand-alone authentication*

## 1 Introduction

In the arena of the network, a ubiquitous Internet demands pervasive connections, while keeping a stable communication environment is the foundation of our entire network. It is well known that user authentication mechanism is an essential stage for creating secure information systems, which can deter the spurious devices and services effectively. However, sometimes user authentication has to draw support from a remote central authorization server, namely, the user send a request to the central server firstly, then the server make a response to the user who requires authenticated, finally according to this challenge the user prove his identity. Each user needs to experience these steps for authentication. But from the whole process of authentication we can see that the central server is critical for completing authentication in the system. At the same time, it is also clear that due to the central server may suffer from various attacks and deliberate destructions, hence holding a dependable information system is not an easy task. Once the server under attack, the users will unable conduct authentication via the authorization server correctly, indicating the users only to wait in place until the server is available. In the long run, this situation will affect communication efficiency seriously and leave numerous hidden troubles for safety.

Yang et al. [14] present a two-factor (smart-card and password) mutual authentication protocol in 2008, aiming at build a generic construction framework for user authentication. Subsequently, in 2011, several researchers [5] put forward a new approach for authentication clients by three-factor (smart-card, password and biometrics). All of these stack factors are simply improving users vali-

date identities, whereas the above schemes are considering nothing about the dangerous that hidden in central server. With the purpose of achieving reliable communication between users and the central server, in 2014 Huang et al. [6] proposed a multi-factor authentication scheme for fragile communications. In this scheme, they first present a stand-alone authentication protocol that can authenticate users when the central server invalidation. Influenced by the stand-alone authentication mechanism, in this paper we present an efficient protocol to build a stable communication environment, which depends on local proof and delayed proof to complete users authentication when the central authorization server breakdown. Local proof is generated by the central server which can meet the requirement of users self-authentication [4] instead of online-authentication. Delayed proof and session key can help the users to set up a temporary platform to confer some inconclusive issues in the course of the authorization server is collapsed. But it is worth noting that the delayed proof is generated by proof, and only in the case of the central server unavailable can the two users employ the delayed proof in exchange. Meanwhile, if involves some crucial problems in the transmission, the users are not allowed to use the delayed proof, which means they must conduct this important process under the central server online. For example, in the process of online shopping, the buyer and the seller denote the two-node respectively, if the central server who refers to the AliPay collapse, two nodes can consult with each other privately with their own delayed proof and session key, but payment step must wait until the server (Alipay) recovery.

What we can see on previous works [2, 5, 10, 14] is that they merely provide the users authentication, but did not consider the situation of server crash. Aiming at meet the aforementioned requirements we first present a chaotic map-based scheme resist fragile communications [3], which we named Chaotic Maps-based scheme against fragile communications (CMFC) in the following scenario. Supposing there are two-node want to transmission under the authentication server called Alice and Bob, our scheme can be divided into two-stage roughly. The first stage we called self-authentication stage: When the authentication server is unavailable, Alice and Bob will build the delayed proof and session key by proof separately and to confer some unimportant affairs with their session key. The second stage named server authentication: Until the central server comes back online, Alice and Bob submit their own delayed proof to conduct authentication and if two-node successfully pass the certification finally, the server will permit the following communications between Alice and Bob.

In this paper, we design delayed proof framework and use the framework and chaotic maps [1, 8, 12, 15] to design the schemes for relieving fragile communications. We proposed several novel protocols which mainly intends to offer a temporary calculating platform to alleviate the waiting time of users. Therefore it is worth mentioning that after authenticated by the central server, the two-node have the right to determine whether they will continue to perform this service or opt-out, and only two-node all choose to continue is the process will keep on. From the above, you can see that our CMFC scheme is quite practical and necessary in critical situation when the server is unavailable because of certain attacks and destructions.

The rest of the paper is organized as follows: Some preliminaries are given in Section 2. Next, a new Local Proof framework and two instances are described in Section 3. In Section 4, we describe delayed proof framework and an instance. In Section 5, we give the security of our proposed protocol. The efficiency analysis of our proposed protocol is given in Section 6. This paper is finally concluded in Section 7.

## 2 Chebyshev Chaotic Maps

Let $n$ be an integer and let $x$ be a variable with the interval [-1, 1]. The Chebyshev polynomial [13] $T_n(x) : [-1, 1] \rightarrow [-1, 1]$ is defined as $T_n(x) = cos(ncos^{-1}(x))$. Chebyshev polynomial map $T_n : R \rightarrow R$ 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:

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

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

$$T_r(T_s(x)) = T_{rs}(x).$$

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 [9] proved that semi-group property holds for Chebyshev polynomials defined on interval $(-\infty, +\infty)$. The enhanced Chebyshev polynomials are used in the proposed protocol:

$$T_n(x) = (2xT_{n-1}(x) - T_{n-2}(x))(modN),$$

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 1.** *(Enhanced Chebyshev polynomials) The enhanced Chebyshev maps of degree $n(n \in N)$ are defined as: $T_n(x) = (2xT_{n-1}(x) - T_{n-2}(x))(mod\ p)$, where $n \geq 2$, $x \in (-\infty, +\infty)$, and $p$ is a large prime number. Obviously, $T_{rs}(x) = T_r(T_s(x)) = T_s(T_r(x))$.*

**Definition 2.** *(DLP, Discrete Logarithm Problem) Given an integer $a$, find the integer $r$, such that $T_r(x) = a$.*

**Definition 3.** *(CDH, Computational Diffie-Hellman Problem) Given an integer $x$, and the values of $T_r(x), T_s(x)$, what is the value of $T_{rs}(x) \stackrel{?}{=} T_r(x)T_s(x)$*

It is widely believed that there is no polynomial time algorithm to solve DLP, CDH with a non-negligible probability.

# 3 Local Proof Framework and Instances

In this section, we first present a novel stand-alone authentication framework and two instances including password-based and biometric-password-based.

## 3.1 Notations

The concrete notations used hereafter are shown in Table 1. These notations can be used in Section 3 and Section 4.

## 3.2 Local Proof Framework

As we consider the efficiency and eliminate the verifier table on the server's side, we adopt only the server has the public key and secret key $\{(x, T_k(x)), K\}$. The server will compute the covered proof and local proof, and send them to the user by secure channel. Figure 1 illustrates the framework of the local authentication and some definitions are described as follows.

**Definition 4.** *Proof.*

A proof means that the server can compute it by his secret key to authenticate the specified user but the proof must be covered by password or biometric for preventing stored in the smart device directly.

**Definition 5.** *Covered Proof.*

A covered proof is a protection mechanism which must receive at least an external input (password or/and biometric) for recovering the proof.

**Definition 6.** *Local Proof.*

A local proof aims at adapting to stand-alone authentication when the authentication server is unavailable. In other words, the result of temporary computation by inputting the password or/and biometric which must equal to the local proof.

**Remark 1.** *The server is available means that both the communications and the authentication server are usable. The server is unavailable means that both the communications and the authentication server are unusable or either of them is unusable.*

## 3.3 Instance of Using Password

Figure 2 illustrates the instance of using password.

**Step 1.** When a user Alice wants to be a new legal user, she chooses her identity $ID_A$, password $PW_A$ and a random number $R$. Then Alice computes $H(PW_A||R)$ and sends $\{ID_A, H(PW_A||R)\}$ to the server via a secure channel.

**Step 2.** Upon receiving $\{ID_A, H(PW_A||R)\}$ from the Alice, the server computes $P = H(ID_A||K)$ as the proof of the user, $V = H(PW_A||R) \oplus H(ID_A||K)$ as the covered proof, $L = H(H(ID_A||K))$ as the local proof and sends $\{V, L\}$ to the Alice.

**Step 3.** Alice stores $\{V, L, R\}$ securely. Storage carrier may be smart card, applications' database or others.

**Local Authentication.** Alice inputs password to recover $P = H(PW_A||R) \oplus V$. Then Alice compares $H(P)$ with $L$. Authentication succeeds only if the hash value matches with the local proof $L$.

## 3.4 Instance of Using Biometric and Password

Figure 3 illustrates the instance of using biometric and password.

**Step 1.** When a user Bob wants to be a new legal user, he chooses his identity $ID_B$, password $PW_B$, a random number $R$ with inputting biometric image sample $B$. Then Bob computes $H(PW_A||R)$ and sends $\{ID_B, H(PW_B||R, B)$ to the server via a secure channel.

**Step 2.** Upon receiving $\{ID_B, H(PW_B||R, B)$ from the Bob, the server computes $P = H(ID_B||K)$ as the proof of the user, $V = H(PW_B||R) \oplus H(ID_B||K)$ as the covered proof, $L = H(PW_B||R) \oplus B$ as the local proof and sends $\{V, L\}$ to the Bob.

**Step 3.** Bob stores $\{V, L, R, \tau, d()\}$ securely.

**Local Authentication.** Bob inputs password to recover $B = H(PW_B||R) \oplus L$. Then Bob inputs $B^*$ and verify $d(B^*, B) < \tau$? Authentication succeeds only if $d(B^*, B) < \tau$.

Table 1: Notations

| Symbol | Definition |
|---|---|
| $ID_i$ $PW_i$ | The identity of user, the password of user, respectively |
| $R, a, b$ | Nonces |
| $(x, T_K(x))$ | The public key of the authentication server based on Chebyshev chaotic maps |
| $K$ | The public key of the authentication server based on Chebyshev chaotic maps |
| $B$ | The biometric sample of user |
| $\tau$ | Predetermined threshold for biometric verification |
| $d()$ | The symmetric parametric function |
| $H$ | A secure one-way hash function |
| $\|\|$ | Concatenation operation |

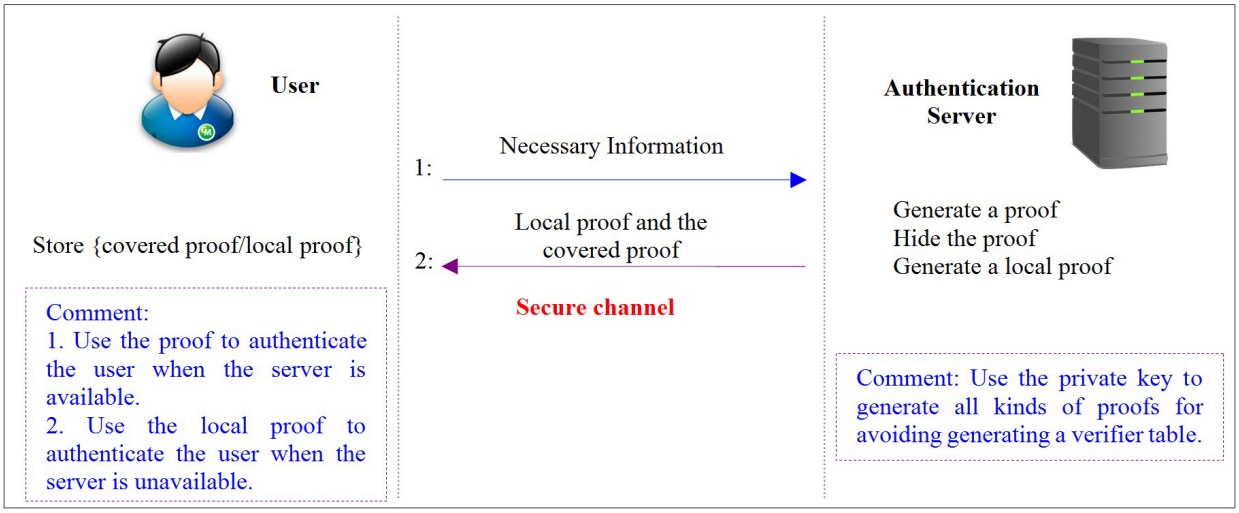


Figure 1: The framework of the local authentication

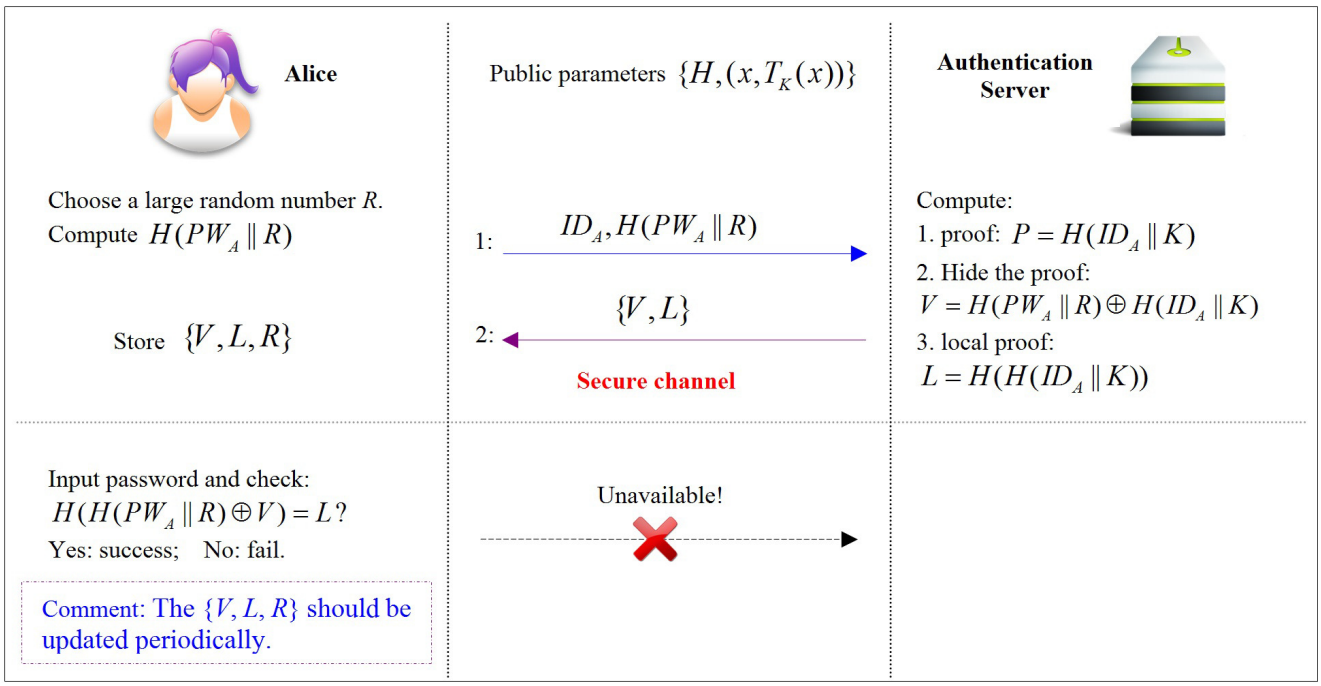


Figure 2: Chaotic maps-based for local authentication scheme with password

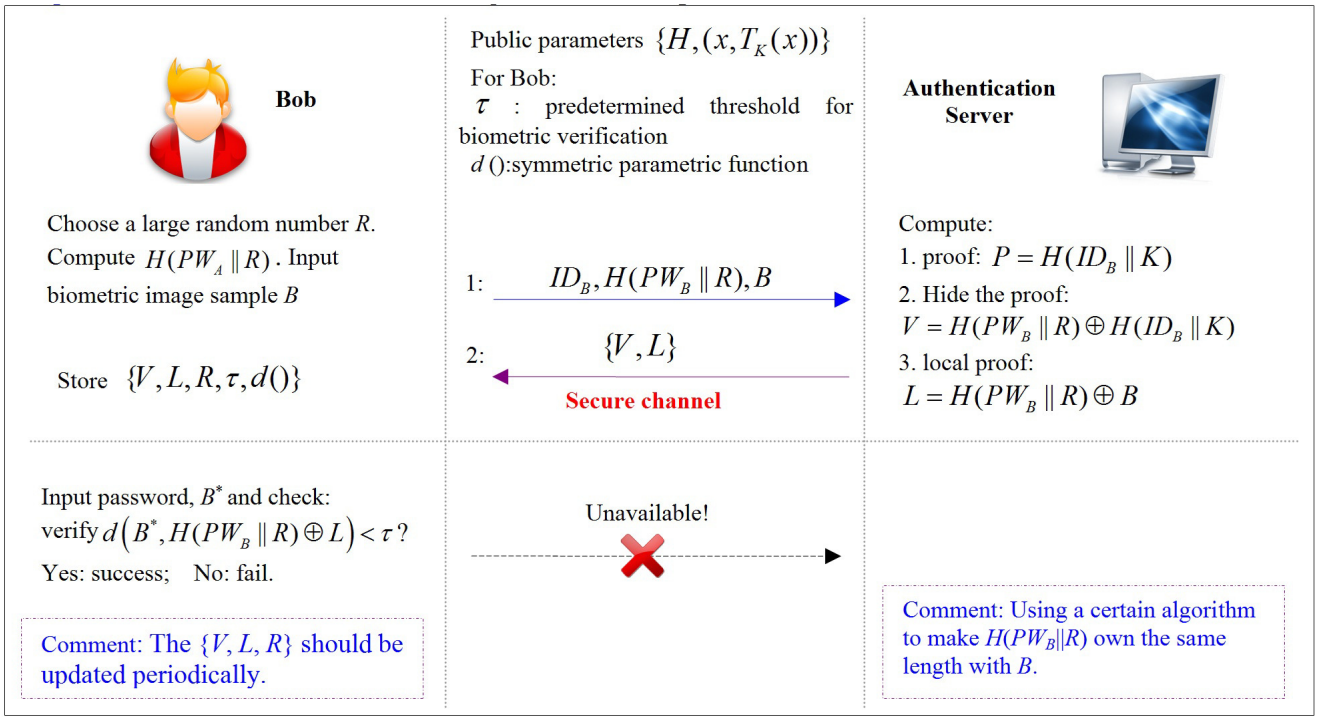


Figure 3: Chaotic maps-based for local authentication scheme with two-factor

# 4 Delayed Proof Framework and an Instance

In this section, we first present a novel Chaotic Maps-based local and delayed proof scheme based on password which is made up of three phases: registration phase, local authentication and delayed authentication.

In brief, the registration phase and local authentication phase are described as shown in Section 3.3 and Figure 2. So, in Section 4, we mainly describe the Delayed Proof Framework and an instance with password-based.

## 4.1 Delayed Proof Framework

The delayed proof is generated from proof which is covered by password or something else. So, the delayed proof is a temporary and random large number that can be used only once. Figure 4 illustrates the framework of the delayed authentication.

The first phase commit with security services (Generate delayed proof with a session key).

When two users (Alice and Bob) want to communicate with each other but the authentication server is unavailable. In order to accelerate efficiency and processing speed, they must temporary consult some unimportant things. So, they construct the delayed proof and session key at the same time based on the proof which can be authenticated by authentication server later.

The second phase commit with security services (Delayed Authentication).

When the two users want to do some important things and the authentication server is available, they submit the other's delayed proof to the server. After receiving the confirm response from the authentication server, the two users must authenticate the authentication server firstly, and then they will continue the unfinished work.

## 4.2 An Instance with Password-based

Figure 5 illustrates an instance with password-based of the delayed authentication.

The first phase commit with security services (Generate delayed proof with a session key).

**Step 1.** Alice inputs $PW_A$ to get the proof: $H(ID_A||K)$. Then Alice selects a large random integer $a$ and computes $T_a(x)$. Finally Alice computes delayed proof: $m_A = \{ID_A, T_a(x), H(ID_A||T_a(x)), DP_A = H(H(ID_A||K) \| T_a(x) \| ID_A)\}$ and sends it to Bob. The same way for Bob.

**Step 2.** Upon receiving $m_A = \{ID_A, T_a(x), H(ID_A||T_a(x)), DP_A = H(H(ID_A||K) \| T_a(x) \| ID_A)\}$ from Alice, Bob computes $H' = H(ID_A||T_a(x))$ and check if it equals to the received hash value $H(ID_A||T_a(x))$. If holds, Bob computes the session key $H(T_bT_a(x))$. The same way for Alice.

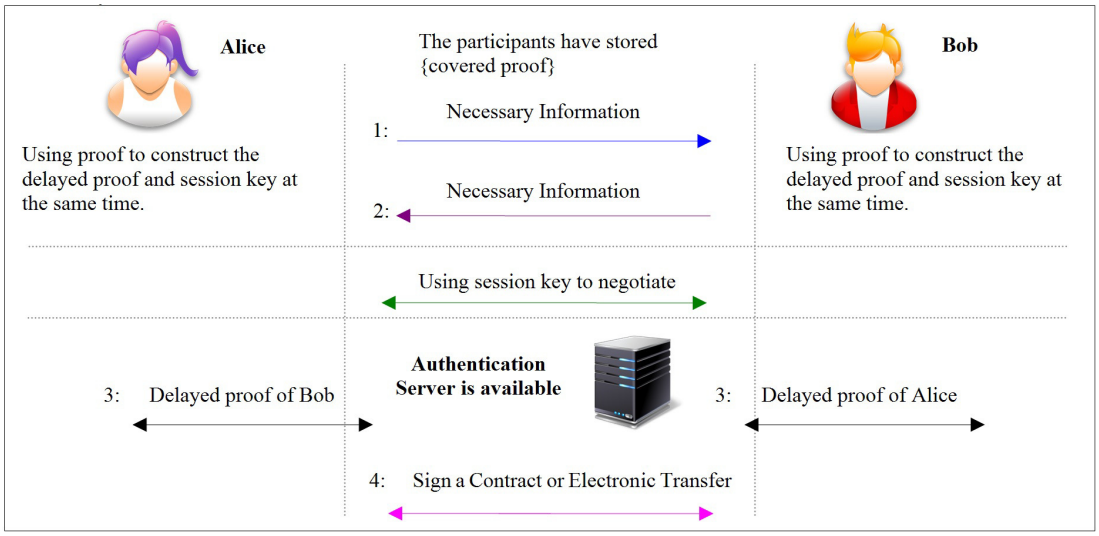


Figure 4: The framework of the delayed authentication

**Step 3.** Alice and Bob communicates with each other and both of them store the other's delayed proof securely.

The second phase commit with security services (Delayed Authentication).

**Step 1.** Alice sends $\{ID_A, m_B\}$ to the authentication server in order to authenticate Bob. The same way for Bob.

**Step 2.** Upon receiving $\{ID_A, m_B\}$ from Alice, the server will compute $H' = H(H(ID_B||K)||T_b(x)||ID_B)$ based on its secret key and the user's identity. Then the server will check if $H' = DP_B$? If holds, the server finishes the task of Bob's authentication. Finally, the server computes $V_A = H(H(ID_A||K)||T_a(x)||ID_S)$ and sends $\{ID_S, V_A\}$ to Alice. The same way for Bob.

**Step 3.** Upon receiving $\{ID_S, V_A\}$ from the server, Alice firstly checks $V_A$ to authenticate the server. If the server is passed validation, Alice will confirm that Bob is the legal and real "Bob". The same way for Bob.

After the second phase is performed, Alice and Bob can continue to do the important things.

# 5 Security Consideration

## 5.1 Security Analysis for Local Authentication

The security of this kind local authentication scheme is based on one way secure hash function. The scheme aims at authenticate oneself efficiently with one-factor authentication. So, this scheme may be sacrifice some security for improving efficiency. For example, losing smart device and guessing attack(An adversary gets the user's smart device and then carries out the guessing attacks.) may deal with this scheme: An adversary gets the smart device and reads the information $\{V, L, R\}$. Then the adversary guesses a password $PW^*$ to compare $H(H(PW^*||R) \oplus V)$ with L repeatedly until guessing the right password.

So, in order to resist losing smart device and guessing attack, we choose two-factor local authentication scheme based on biometric and password. This kind of scheme must be verified by two-factor authentication which can lead to some computations and hardware spending.

## 5.2 Security Analysis for Delayed Authentication

For simplicity, we only discuss the delayed authentication with password-based, and we do not design with biometric. In this section, there is no local proof stored in smart device in our scheme for resisting lost smart device and guessing attack. So, we only use covered proof to construct the delayed proof. From the **Table 2**, we can see that the proposed scheme can provide known secure session key agreement, impersonation attack and so on.

## 5.3 Security Proof Based on the BAN Logic [1] for Delayed Authentication

For convenience, we first give the description of some notations (Table 3) used in the BAN logic analysis and define some main logical postulates (Table 4) of BAN logic. We combine the two phases (Generate delayed proof with a session key, Delayed Authentication) together to prove, because only in the second phase the two involved users can just do the important things.

According to analytic procedures of BAN logic and the requirement of delayed authentication scheme, our CMFC

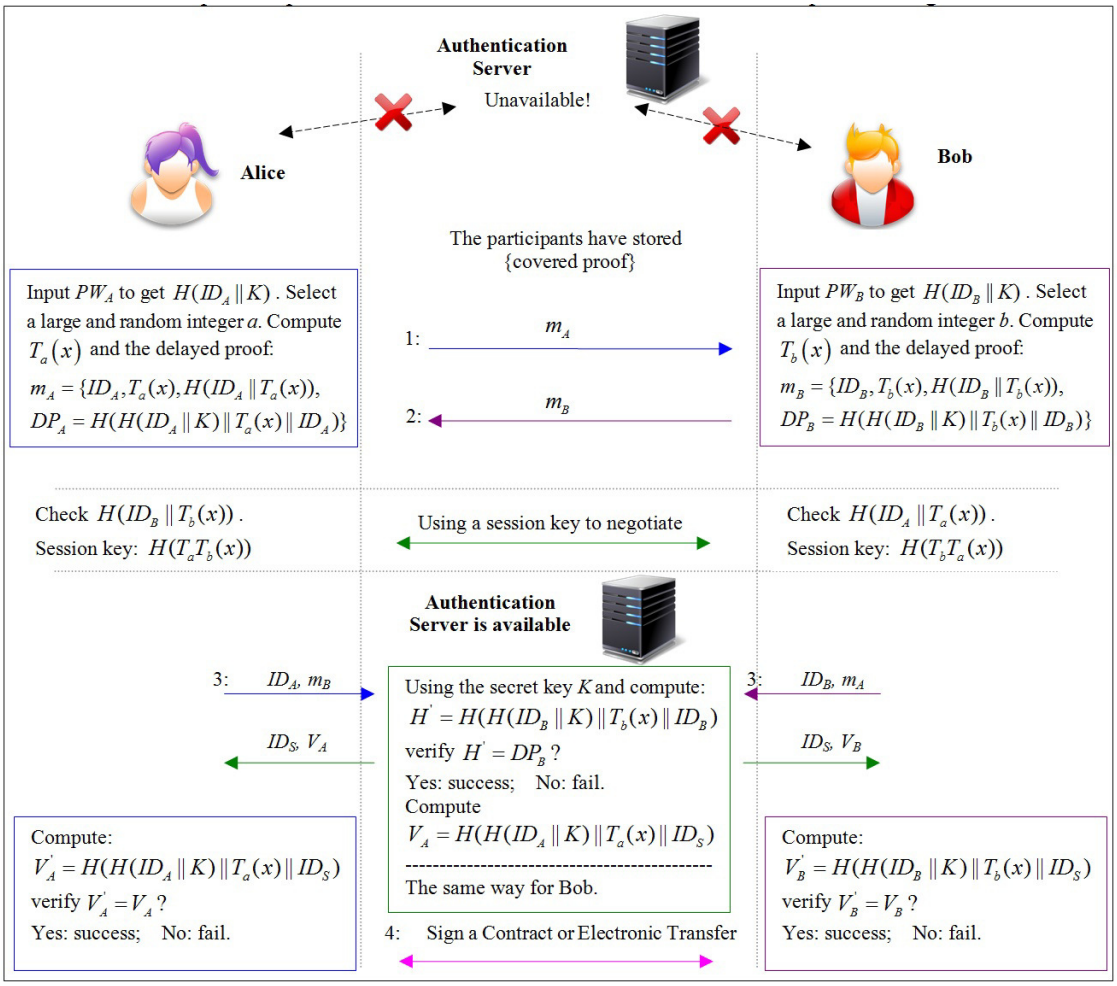


Figure 5: The delayed authentication with password-based and chaotic maps

Table 2: Definition and simplified proof with combining the two phases in section 4.2

| Attack Type | Security Requirements | Definition | Simplified Proof | Hard Problems |
|---|---|---|---|---|
| Automatic validation attacks | Guessing attacks (On-line or off-line) | In an off-line guessing attack, an attacker guesses a password or long-term secret key and verifies his/her guess, but he/she does not need to participate in any communication during the guessing phase. In an undetectable on-line guessing attack, an attacker searches to verify a guessed password or long-term secret key in an on-line transaction and a failed guess cannot be detected and logged by the server. | There is no any match value to compare | Secure one way hash |
| | Losting smart device and guessing attacks | An adversary gets the users smart device and then carries out the guessing attacks. | There is no any match value to compare | Secure one way hash |
| No freshness verify attacks | Perfect forward secrecy | An authenticated key establishment protocol provides perfect forward secrecy if the compromise of both of the nodes secret keys cannot results in the compromise of previously established session keys. | Different session has different nonces. | Chaotic maps problems |
| | Known session key security | Each execution of the protocol should result in a unique secret session key. The compromise of one session key should not compromise the keys established in other sessions. | Different session has different nonces. | Chaotic maps problems |
| Missing encrypted identity attacks | Man-in-the-middle attack(MIMA) | The MIMA attack is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other over a private connection, when in fact the entire conversation is controlled by the attacker. | All the information includes the $ID$ and some nonces: $a, b$ and the another form $T_a(x), T_b(x)$. | Chaotic maps problems |
| | Impersonation attack | An adversary successfully assumes the identity of one of the legitimate parties in a system or in a communications protocol. | All the information includes the $ID$ and some nonces: $a, b$ and the another form $T_a(x), T_b(x)$. | Chaotic maps problems |
| No freshness verify attacks | Replay attack | A replay attack is a form of network attack in which a valid data transmission is repeated or delayed maliciously or fraudulently. | Every important message includes the nonces: $a, b$ and the another form $T_a(x), T_b(x)$. | Chaotic maps problems |
| Design defect attacks | 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 are no any verification tables in any node. | Natural Resistance |

Table 3: 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 $Y$. |
| $X_K$ | The formula $X$ is encrypted under the key $K$. |
| $(X)_K$ | The formula $X$ is hash function with the key $K$. If there is no $K$, and that means is no key input. |
| $P \overset{K}{\longleftrightarrow} Q$ | 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 4: Logical postulates of the BAN logic

| Symbol | Definition |
|---|---|
| $\frac{P\mid\equiv P \overset{K}{\longleftrightarrow} Q, P\{X\}_K}{P\mid\equiv Q\mid\sim X}$ | R1: The message-meaning rule |
| $\frac{P\mid\equiv\#(X)}{P\mid\equiv\#(X,Y)}$ | R2: The freshness-conjuncatenation rule |
| $\frac{P\mid\equiv\#(X), P\mid\equiv Q\mid\sim X}{P\mid\equiv Q\mid\equiv X}$ | R3: The nonce-verification rule |
| $\frac{P\mid\equiv Q\mid\Rightarrow X, P\mid\equiv Q\mid\equiv X}{P\mid\equiv X}$ | R4: The jurisdiction rule |
| $\frac{P\mid\equiv Q\mid\equiv(X,Y)}{P\mid\equiv Q\mid\equiv X}$ | R5: The belief rules |
| Molecule can deduce denominator for above formulas. | |

scheme should satisfy the following goals in Table 5.

Table 5: Goals of the proposed scheme

| |
|---|
| Goal 1. $U_A\mid\equiv(U_A \overset{SK}{\longleftrightarrow} U_B)$; |
| Goal 2. $U_A\mid\equiv U_B\mid\equiv(U_A \overset{SK}{\longleftrightarrow} U_B)$; |
| Goal 3. $U_B\mid\equiv(U_A \overset{SK}{\longleftrightarrow} U_B)$; |
| Goal 4. $U_B\mid\equiv U_A\mid\equiv(U_A \overset{SK}{\longleftrightarrow} U_B)$; |

First of all, we transform the process of our protocol to the following idealized form. Because only the second phase can make the two users (Alice and Bob) do the important, we just begin with the authentication server is available to analyse.

For Alice and authentication server:

$(U_A \rightarrow Server)m_1$: $Server \lhd ID_A$, $ID_B$, $T_b(x)$, $(ID_B||T_b(x))$, $((H(ID_B||K))||T_b(x)||ID_B)$;

$(Server \rightarrow U_A)m_2$: $U_A \lhd ID_S$, $(H(ID_A \mid\mid K) \mid\mid T_a(x) \mid\mid ID_S)$.

For Bob and authentication server:

$(U_B \rightarrow Server)m_3$: $Server \lhd ID_A$, $ID_B$, $T_a(x)$, $(ID_A||T_a(x))$, $((H(ID_A||K))||T_a(x)||ID_A)$;

$(Server \rightarrow U_B)m_4$: $U_B \lhd ID_S$, $(H(ID_B \mid\mid K) \mid\mid T_b(x) \mid\mid ID_S)$.

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 6.

Based on the above assumptions, the idealized form of our scheme is analyzed as follows. We only analyze the process of Alice and authentication server, and the same way for Bob and authentication server.The main steps of the proof are described as follows:

**For $m_1$:**

According to $m_1$ and $P_1$, $P_5$ and relating with $R_1$, we could get: $S_1 : U_A \mid\equiv Server \mid\sim m_1$.

Based on $m_1$ and the initial assumptions $P_1$, $P_3$, $P_4$, $P_5$, $P_7$, we could get: $S_2 : Server \mid\equiv \#m_1$.

Combine $S_1, S_2, P_3, P_4, P_5, P_7, R_2$, we could get: $S_3$: $Server \mid\equiv \#\ ID_A$, $ID_B$, $T_b(x)$, $((H(ID_B \mid\mid K)) \mid\mid T_b(x) \mid\mid ID_B)$.

Based on $R_3$, we take apart $S_3$ and get: $S_4$ : $Server \mid\equiv \#T_b(x)$, $S_5 : Server \mid\equiv \#((H(ID_B||K)) \mid\mid T_b(x) \mid\mid ID_B)$.

Based on the secret key $K$ and $S_2$, the server can authenticate Bob by computing the new hash value to compare with $S_5$. If the server authenticate Bob, it will continue to the process of $m_2$.

**For $m_2$:**

According to $m_2$ and $P_1, P_7$ and relating with $R_1$, we could get: $S_6 : Server \mid\equiv U_A \mid\sim m_2$.

Based on $m_2$ and the initial assumptions $P_1$, $P_3$, $P_4$, $P_5$, $P_7$, we could get: $S_7 : U_A \mid\equiv \#m_2$.

Table 6: Assumptions about the initial state of our protocol

| Initial states | |
|---|---|
| $P_1 : U_A\| \equiv \overset{T_K(x)}{\longrightarrow} Server$ | $P_2 : U_B\| \equiv \overset{T_K(x)}{\longrightarrow} Server$ |
| $P_3 : U_A\| \equiv \#(a)$ | $P_4 : U_B\| \equiv \#(b)$ |
| $P_5 : U_A\| \equiv U_A \overset{H(ID_A\|\|K)}{\longleftrightarrow} Server$ | $P_6 : U_B\| \equiv U_B \overset{H(ID_B\|\|K)}{\longleftrightarrow} Server$ |
| $P_7 : Server\| \equiv U_A \overset{H(ID_A\|\|K)}{\longleftrightarrow} Server$ | $P_8 : Server\| \equiv U_B \overset{H(ID_B\|\|K)}{\longleftrightarrow} Server$ |

Combine $S_6, S_7, P_3, P_4, P_5, P_7, R_2$, we could get: $S_8$ : $U_A \mid\equiv \#ID_S, (H(ID_A||K)||T_a(x)||ID_S)$.

Based on $R_3$, we take apart $S_8$ and get: $S_9$ : $Server \mid\equiv \#(H(ID_A||K)||T_a(x)||ID_S)$.

Based on $P_5, P_7$ and $S_9$, Alice can authenticate the server by computing the new hash value to compare with $S_9$.

**Combine:**

Because the Alice, Bob and the Server communicate each other just now, they confirm the other is on-line. And based on $S_9, R_4$ with chaotic maps problems, we could get:

**Goal 1.** $U_A\| \equiv (U_A \overset{SK}{\longleftrightarrow} U_B)$;

**Goal 2.** $U_A\| \equiv U_B\| \equiv (U_A \overset{SK}{\longleftrightarrow} U_B)$.

The same way for Bob and the server, we could get:

**Goal 3.** $U_B\| \equiv (U_A \overset{SK}{\longleftrightarrow} U_B)$;

**Goal 4.** $U_B\| \equiv U_A\| \equiv (U_A \overset{SK}{\longleftrightarrow} U_B)$.

According to (Goal 1 $\sim$ Goal 4), we know that both Alice and Bob believe that the Server can authenticate them and the session key is fresh based on the fresh nonces $a, b$.

# 6 Efficiency Analysis

## 6.1 The Comparisons Among Different Algorithms

Compared to RSA, ECC and Bilinear map, Chebyshev polynomial computation problem offers smaller key sizes, faster computation, as well as memory, energy and bandwidth savings. Chaotic maps encryption algorithm: As a special form of motion, Chaos means that in a certain nonlinear system can appear similar to the behavior of random phenomena without needing any random factors. Chaotic system has the characteristics of certainty, boundness, sensibility to initial parameters and unpredictability, etc. Chaotic maps encryption algorithm utilizes the unique semi-group mature of Chebyshev chaotic maps, based on two difficult problems-the chaotic maps discrete logarithm problem and the chaotic maps Diffie-Hellman problem, puts forward a kind of encryption algorithm. Compared with ECC encryption algorithm, Chaotic maps encryption algorithm avoids scalar multiplication and modular exponentiation computation, effectively improves the efficiency. However, Wang [13] 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 [7]. Moreover, the computational cost of XOR operation could be ignored when compared with other operations. According to the results in [11], one pairing operation requires at least 10 times more multiplications in the underlying finite field than a point scalar multiplication in ECC does in the same finite field.

Through the above mentioned analysis, we can reached the conclusion approximately as follows:

$$T_p \approx 10T_m, T_m \approx 3T_c, T_c \approx 2.42T_s, T_s \approx 17.4T_h,$$

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, $T_s$: Time for symmetric encryption algorithm, $T_h$: Time for Hash operation.

About these algorithms, our proposed CMFC scheme only used the chaotic cipher and a secure one way hash as the main algorithm (see Table 7) which are more efficient bilinear pair operation and a point scalar multiplication operation ECC-based. Especially for hash operation, it can be ignored compared with the other three algorithms.

## 6.2 The Sum Up About Our Scheme's Efficiency

Because there are no any related literatures, we can not give any comparisons about efficiency. From **Table 7**, we can conclude that our CMFC scheme has high-efficient property.

Table 7: Our proposed scheme's efficiency

| Type | Instance | Party | Ours CMFC scheme | |
|---|---|---|---|---|
| Local Proof | Only password | Server | Registration | $2T_h$ |
| | | | Local authentication | No need |
| | | Alice | Registration | $1T_h$ |
| | | | Local authentication | $2T_h$ |
| | Biometric and password | Server | Registration | $1T_h$ |
| | | | Local authentication | No need |
| | | Bob | Registration | $1T_h$ |
| | | | Local authentication | $1T_h$+$1T_s$ |
| Delayed Proof | Only password | Server | Unavailable Server Phase | No need |
| | | | Delayed authentication | $4T_h$ |
| | | Alice | Unavailable Server Phase | $5T_h$+$1T_c$ |
| | | | Delayed authentication | $2T_h$ |
| | | Bob | Unavailable Server Phase | $5T_h$+$1T_c$ |
| | | | Delayed authentication | $2T_h$ |

$T_h$:Time for Hash operation
$T_s$:Time for symmetric parametric function
$T_c$:The time for executing the $T_n(x)$ mod $p$ in Chebyshev polynomial using the algorithm in literature [9].

# 7 Conclusions

In this paper, we propose CMFC, a novel idea towards resisting fragile communications which is divided into the framework and the instance. The framework is the macrostructure which can be implemented by many algorithms, such as RSA, ECC and Bilinear map. For clearing expression of the framework, we illustrate three instances: For local proof framework, we use one factor (password) and two-factor (biometric with password) to construct two instances for adapting to different environment. For delayed proof framework, we only use one factor (password) to construct a instance due to limited space. In addition, we give some new definitions about the meanings of different proofs. Security consideration and efficiency analysis are also focused on discussion. CMFC is not a panacea, but it offers reasonable security, preferable efficiency, easy usability, and appears to fit well with some practical applications – when the authentication server is unavailable. We often face the fragile communications, so we must do something, that is the core motivation of this paper.

# Acknowledgement

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

# References

[1] P. Barreto, B. Lynn, M. Scott, "On the selection of pairing-friendly groups," in *Selected Areas in Cryptography*, LNCS 3006, pp.17–25, Springer-Verlag, 2004.

[2] A. Bhargav-Spantzel, A. C. Squicciarini, S. Modi, M. Young, E. Bertino and S. J. Elliott, "Privacy preserving multi-factor authentication with biometrics," *Journal of Computer Security*, vol. 15, no. 5, pp. 529–560, 2007.

[3] S. Bradner, "What a fragile communications web we've woven," *Network World*, vol. 25, no. 6, pp.33, 2008.

[4] D. Chhabra, S. Zhao, W. Lee and N. Okamoto, "Negotiated self-authenticated experience and homeland travel loyalty: implications for relationship marketing," *Anatolia*, vol. 23, no. 3, pp. 429–436, 2012.

[5] X. Huang, X. Yang, A. Chonka, J. Zhou, and R. H. Deng, "A generic framework for three-factor authentication: Preserving security and privacy in distributed systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1390–1397, 2011.

[6] X. Huang, X. Yang, E. Bertino, J. Zhou and X. Li, "Robust Multi-Factor Authentication for Fragile Communications," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 6, pp. 568–581, 2014.

[7] Q. Jiang, F. Wei, S. Fu, J. Ma, G. Li and A. Alelaiwi, "Robust extended chaotic maps-based three-factor authentication scheme preserving biometric template privacy," *Nonlinear Dynamics*, vol. 83, no. 4, pp. 2085-2101, Mar. 2016.

[8] L. Kocarev, and S. Lian, *Chaos-Based Cryptography: Theory, Algorithms and Applications*, Springer, 2011.

[9] L. Zhang, "Cryptanalysis of the public key encryption based on multiple chaotic systems," *Chaos Solitons Fractals*, vol. 37, no. 3, pp. 669–674, 2008.
[10] H. Wang, H. Zhang, J. Li and X. Chen, "A(3,3) visual cryptography scheme for authentication," *Journal of Shenyang Normal University (Natural Science Edition)*, vol. 31, no. 101(03), pp. 397-400, 2013.
[11] X. Wang, and J. Zhao, "An improved key agreement protocol based on chaos," *Communication Nonlinear Science Number Simulation*, vol. 15, pp. 4052–4057, 2010.
[12] G. Yang, D. S. Wong, H. Wang and X. Deng, "Two-factor mutual authentication based on smart cards and passwords," *Journal of Computer and System Sciences*, vol. 74, no. 7, pp. 1160–1172, 2008.
[13] H. Zhu, Y. Zhang, Y. Zhang and H. Li, "A novel and provable authenticated key agreement protocol with privacy protection based on chaotic maps towards mobile network," *International Journal of Network Security*, vol. 18, no. 1, pp. 116–123, Jan. 2016.
[14] H. Zhu, Y. Zhang, Y. Xia, and H. Li, "Password-authenticated key exchange scheme using chaotic maps towards a new architecture in standard model," *International Journal of Network Security*, vol. 18, no. 2, pp. 326–334, Mar. 2016.
[15] H. Zhu, Y. Zhang, H. Li, and L. Lin, "A novel biometrics-based one-time commitment authenticated key agreement scheme with privacy protection for mobile network," *International Journal of Network Security*, vol. 18, no. 2, pp. 209–216, Mar. 2016.

**Yang Sun** obtained his master degree in Information Science and Engineering from Northeastern University. Yang Sun is a full associate professor of the Kexin software college at Shenyang Normal University. He is also a department head of network engineering. He has research interests in wireless networks, mobile computing, cloud computing, social networks and network security. Yang Sun had published more than 15 international journal and conference papers on the above research fields.

**Xueshuai Feng** graduated with a Bachelor of Engineering from Shenyang Normal University in 2015. In his college, after completing the learning task, he interests in exploring his professional knowledge. During graduate, under the guidance of his master instructor, he researches IoT security theory and technology.

**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.