Hindawi
Security and Communication Networks
Volume 2021, Article ID 6697898, 21 pages
https://doi.org/10.1155/2021/6697898

Research Article

Privacy-Preserving Two-Factor Key Agreement Protocol Based on Chebyshev Polynomials

Zuowen Tan

Department of Computing Science and Technology, Jiangxi University of Finance & Economics, Nanchang 330032, China

Correspondence should be addressed to Zuowen Tan; tanzyw@163.com

Received 24 December 2020; Revised 28 April 2021; Accepted 24 May 2021; Published 3 June 2021

Academic Editor: Ahmad Samer Wazan

Two-factor authentication is one of the widely used approaches to allow a user to keep a weak password and establish a key shared with a server. Recently, a large number of chaotic maps-based authentication mechanisms have been proposed. However, since the Diffie–Hellman problem of the Chebyshev polynomials defined on the interval [−1,+1] can be solved by Bergamo et al.'s method, most of the secure chaotic maps-based key agreement protocols utilize the enhanced Chebyshev polynomials defined on the interval (−∞,+∞). Thus far, few authenticated key agreement protocols based on chaotic maps have been able to achieve user unlinkability. In this paper, we take the first step in addressing this problem. More specifically, we propose the notions of privacy in authenticated key agreement protocols: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Then, we construct two two-factor authentication schemes with medium unlinkability based on Chebyshev polynomials defined on the interval [−1,1] and (−∞,+∞), respectively. We do the formal security analysis of the proposed schemes under the random oracle model. In addition, the proposed protocols satisfy all known security requirements in practical applications. By using Burrows-Abadi-Needham logic (BAN-logic) nonce verification, we demonstrate that the proposed schemes achieve secure authentication. In addition, the detailed comparative security and performance analysis shows that the proposed schemes enable the same functionality but improve the security level.

1. Introduction

User authentication is indispensable for many information systems. Authenticated key agreement enables users to establish a session key which two or more parties share over a public channel. The session keys are adopted in subsequent secure communications. Password, hard-ware, and biometrics are always utilized in authentication mechanisms [1–3]. Single-factor authentication only provides limited security; then, the combination of these methods together can achieve higher security. Due to the convenient portability of smart cards, two-factor authentication [4–6] has been intensively investigated.

In general, user privacy protection during authenticated key exchange is a big challenge. For two-factor authentication schemes, the important issues should be addressed carefully. Firstly, the authentication mechanism should hide the user's identity from any eavesdroppers and foreign servers. In other words, mutual authentication cannot reveal the real identity of the user, which is the basic goal of privacy protection, called anonymity. Secondly, the other aspect of user privacy protection is unlinkability. In many applications, the authentication mechanism should hide the user's movements from any eavesdroppers and other foreign servers. Any unauthorized entity cannot track the user's movements. Even if any outside adversary has accessed the message transmitted between the user and the server, it still cannot link the user to different authentication sessions.

Since chaotic maps provide the semigroup property and have higher efficiency than modular exponential operations and scalar multiplications on an elliptic curve, many chaotic, map-based, two-factor authenticated key agreement protocols [7–12] have been developed in recent years. Thus far, the user privacy preserving chaotic maps-based authenticated key agreement protocols have been intensively investigated [13–17].

1.1. Motivation. In two-factor authenticated key agreement protocols, the user privacy must be considered carefully. The basic security requirement is to preserve the user anonymity, while the stricter requirement is unlinkability or untraceability. These concepts in two-factor authenticated key agreement protocols are seldom discussed in detail. Till date, few two-factor authenticated key agreement protocols can provide users with a strong unlinkability.

To the best knowledge, most of the two-factor authenticated key agreements based on the Chebyshev polynomial protocols (hereafter, called TAKACP protocols) utilize the enhanced Chebyshev polynomials defined on the interval $(-\infty, +\infty)$. Now, few secure TAKACP protocols are based on the Chebyshev polynomials defined on the interval [−1, +1]. This is because those TAKACP protocols based on the Chebyshev polynomials over the interval [-1,+1] are vulnerable to Bergamo et al.'s attacks [18].

1.2. Our Contributions. The main contribution of this paper is the two-factor authenticated key agreement protocol based on Chebyshev polynomials defined on the interval [−1,1] and $(-\infty,+\infty)$, respectively, which solve all above-mentioned issues for the first time. They satisfy more security requirements than the existing TAKACP protocols.

In summary, we list the main contributions below:

The user privacy in two-factor authenticated key agreement protocol is expounded. According to the extent of user identity protection, the user privacy preserving in entity authentication protocols is classified into four concepts: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Of the four levels, the strong unlinkability is the highest while the anonymity-alone is fundamental for entity authentication. In this paper, we elaborate them by the formal probability model.

We analyze the Lin TAKACP protocol and reveal their weaknesses. Detailed analysis shows that it fails to provide session key security. And, it suffers from user impersonation attack. Next, it even cannot provide the weak unlinkability.

Analysis of formal security under Random Oracle model and BAN logic nonce verification demonstrate that the proposed schemes provide secure authentication against the CPCDH assumption and integer factorization hardness assumption.

The proposed schemes provide more security properties as compared to other TAKACP schemes. And the detailed comparative security analysis also shows that the proposed schemes avoid the weaknesses of the TAKACP protocols [19–21].

1.3. System Model

1.3.1. Network Model. The proposed two-factor authenticated key agreement protocols involve two entities, a user U and the server S. At the user registration phase, the server issues a smart card with secret information to U through a secure channel. When the user U logs in to the server, the server and the user authenticate each other over the public channels. In this paper, the two-factor authenticated key agreement protocol is required to provide users with privacy preserving. Any adversary cannot trace the user from the message transmitted over open channels.

1.3.2. Adversary Model. Consider an adversary A who gets the full control over the communication channel between the user U and the service provider S (except the registration phase). Thus, A could obtain the messages transmitted between the user and the server (except the registration message). Of the phases in a two-factor authenticated key agreement protocol, only the registration phase requires a secure channel between U and S. In other phases, there could be various kinds of passive and active adversaries in the communication channel between U and S. The adversary A can eavesdrop and even block the message transmitted, modify messages, remove messages, or insert messages into the communication channel. Its objective is to compromise the mutual authentication between U and S. A even impersonates *a user* and attempts to access the server, or the adversary impersonates *the server* and provides *the user* with false services.

In the single-server environment, since users register on the same server with the same master key, the inside attacker $\boldsymbol{A_0}/\boldsymbol{A_1}$ is very powerful. Hereinafter, we refer to a malicious server which may try to recover the password of its client or track the client as an adversary $\boldsymbol{A_0}$; a registered malicious user, or an adversary who has corrupted the user as an adversary $\boldsymbol{A_1}$; and while other adversaries are called as outside adversary $\boldsymbol{A_2}$. To simulate the inside attack, $\boldsymbol{A_0}$ and $\boldsymbol{A_1}$ can get the passwords and information stored in the smart cards of the users except those of a client under attack. If the server is the attack target, $\boldsymbol{A_1}$ is assumed to obtain the passwords and the information stored in the smart cards of all the users.

For a two-factor authentication scheme, the basic security property is that the user is required to both have the smart card and know the password. Since the smart cards cannot prevent the information stored in them from being extracted, for example, by monitoring their power consumption, the security of a two-factor authentication scheme is always discussed in the case that the smart card is stolen. In other words, when a user is under attack, A is allowed to either compromise the password or the smart card of the client under attack, but not both.

1.4. Organization of the Paper. The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces some preliminaries. Section 4 shows the limitations of the Lin protocol. Section 5 then presents two novel TAKACP protocols. Next, Section 6 analyzes the security of the proposed schemes. Comparison with the related smart-card-based protocols in terms of security properties and performance will be given in Section 7 , and Section 8 is the conclusion.

Table 1: Cryptographic methodologies and drawbacks of the existing schemes and the proposed schemes.

	Fan et al. [22]	Juang et al. [23]	Sun et al. [24]	Li et al. [25]	Guo et al. [19]	Lin [20]	Lee [21]	Proposed *TAKACP* protocols
Cryptographic methodologies	Symmetric encryption Rabin's public-key cryptosystem	Symmetric encryption ECC	Symmetric encryption ECC	Public-key cryptosystem ECC	Symmetric encryption chaotic map on the interval [−1,1]	Symmetric encryption chaotic map on the interval [−1,1]	Symmetric encryption chaotic map on the interval (−∞,+∞)	Symmetric encryption, Rabin's public-key cryptosystem chaotic map on the interval [−1,1]/(−∞,+∞)
Drawbacks	Insecure, no privacy preservation	Insecure, no privacy preservation	Traceability	Insecure, no privacy preservation	Traceability	Traceability	No free password updating	None

2. Related Work

In this section, we briefly review some prior related works. Recent years have witnessed efforts on two-factor authentication [19–42]. We summarize some existing two-factor authentication schemes with their methodologies used, limitations, and drawbacks in Table 1.

2.1. Two-Factor Authentication Based on Enhanced Chebyshev Polynomials Defined on the Interval (−∞, +∞) and Their Limitations. Researchers have developed chaotic maps-based key agreement protocols which utilize the enhanced Chebyshev polynomials defined on the interval (−∞, +∞). Xiao et al. [31] first presented a chaotic map-based authenticated key agreement protocol by utilizing the semigroup property of Chebyshev chaotic maps [32, 33]. Guo and Zhang [34] showed that the Xiao et al.'s protocol [31] cannot provide the contributory property of key agreement. A malicious server can predetermine the session key. Guo and Zhang presented an improved version [34]. However, Lee demonstrated that the Guo-Zhang protocol [34] is insecure against off-line password guessing attacks [35]. In addition, it fails to provide the session key security. Tseng et al. [7] proposed anonymous key agreement protocol based on Chebyshev chaotic maps. Unfortunately, Niu et al. [8] demonstrate that Tseng et al.'s protocol [7] fails to protect the user anonymity and suffers from inside attacks. Yoon [9] found that the Niu-Wang protocol [8] is vulnerable to Denial of Service attacks. Xue et al. [36] also improved Tseng et al.'s protocol. However, Tan [37] pointed out that the Xue-Hong protocol [36] cannot still provide user anonymity. Moreover, the Xue-Hong protocol suffers from man-in-the-middle attacks. In 2012, Gong et al. [38] proposed password-based key agreement protocol by using extended chaotic maps. Unfortunately, Wang and Luan [39] showed that the key agreement protocol [38] suffers from potential security problems.

In 2014, Lin [40] developed an authentication scheme using dynamic identity and chaotic maps. Later, Islam et al. [41] state that Lin's scheme suffers from user impersonation attack. Islam et al. also presented an improved provably secure scheme [41, 42] to solve the weaknesses of Lin's scheme. Unluckily, Jiang et al. [10] show that Islam's scheme is also vulnerable to some potential attacks. Based on the extended Chebyshev polynomials on the interval (−∞,+∞), Lee et al. [21] presented improvement on Lin's scheme [20]. However, in the login phase of the improved scheme [21], the smart card fails to validate the input of the user. Moreover, in the password change phase, the server must participate in the whole updating process of each user. Hence, it is inconvenient for users to update the password in Lee et al.'s scheme [21].

2.2. Two-Factor Authentication Based on Enhanced Chebyshev Polynomials Defined on the Interval [−1,+1] and Their Limitations. Few secure TAKACP protocols [19–21, 35] based on the Chebyshev polynomials defined on the interval [−1,+1] have been presented.

In Lee's TAKACP scheme [35], the user and the server must preshare a password. Hence, when users register with the server, the server must share one different password with every user. In 2013, Guo and Chang [19] have proposed a smart-card-based authenticated key agreement using chaotic maps over the interval [−1,+1]. Subsequently, Hao et al. [43] and Lin [20] pointed out that that there are some security pitfalls in the Guo-Chang scheme [19]. Any adversary can derive the session key only by using the messages transmitted between a user and the server. In addition, the Guo–Chang scheme fails to provide full protection for user identity due to a fixed parameter in every run of the protocol. To eliminate the above weaknesses, Lin presents an improved scheme [20] based on chaotic maps over the interval [−1,+1]. The Lin scheme is highly efficient since it is based on a simple symmetric cryptosystem. Unfortunately, Lee et al. [21] point out that the Lin scheme still fails to withstand denial-of-service and privileged insider attacks. In addition, the Lin scheme does not exhibit the contributory property of key agreements.

In this paper, we will show that the Lin scheme violates the session key security. The Lin scheme suffers from impersonation attacks. Specifically, it is still susceptible to Bergamo et al.'s attacks [44] from registered users of the

same server. Furthermore, we will demonstrate that the Lin scheme cannot provide the strong privacy protection. We also have found that it is inconvenient for users to update passwords in the Lin scheme [20] and the Guo–Chang scheme [19].

3. Mathematical Preliminaries

This section briefly introduces Chebyshev polynomials and two problems related to the chaotic maps. Then, we will discuss the user privacy in TAKACP protocols and define the different notions of user privacy.

3.1. Mathematical Preliminaries. Let n be an integer and x be a variable taking values over the interval $[-1,1]$. The Chebyshev polynomial T_n: $[-1,1] \longrightarrow [-1,1]$ of degree n is defined as

$$T_n(x) = \cos(n \cdot \arccos(x)), \quad x \in [-1,1]. \tag{1}$$

The recurrence relation of the Chebyshev polynomial is given by:

$$\begin{aligned} &T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad n \geq 2, \\ &T_0(x) = 1, T_1(x) = x. \end{aligned} \tag{2}$$

The cos(s) and arcos(s) are defined as cos: $R \longrightarrow [-1,1]$ and arcos: $[-1,1] \longrightarrow [0,\pi]$. There are some examples of Chebyshev polynomials that are shown as follows:

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

The Chebyshev polynomials hold two important properties.

Semigroup property: Assume that r and s are positive numbers. For $x \in [-1,1]$, $T_r(T_s(x)) = T_s(T_r(x))$. Due to its semigroup property, the Chebyshev polynomials satisfy the commutative property under the composition $T_r(T_s(x)) = T_s(T_r(x))$.

Chaotic property: Since the Chebyshev polynomial $T_n(x)$ with the positive integer n has a unique continuous invariant measure with positive Lyapunov exponent lnn, it is a chaotic map with its invariant density $f^*(x) = 1/(\pi\sqrt{1-x^2})$. Specially, $T_2(x)$ is the well-known logistic map.

In 2008, Zhang [45] extended the definition of variables from the interval $[-1,1]$ to the interval $(-\infty, +\infty)$ as follows:

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

where p is a large prime number. And, these enhanced Chebyshev polynomials still commute under the composition, $T_r(T_s(x)) = T_s(T_r(x)) \bmod p$.

The Chebyshev polynomials over the interval $(-\infty, +\infty)$ have the discrete logarithm problem and Diffie-Hellman problem, which are assumed to be difficult to solve within a probabilistic polynomial time:

Definition 1. Chebyshev polynomial-based Discrete Logarithm (*CPDL*) problem. Given two elements x and y, find an integer r, such that $T_r(x) = y$, where $T_r(x)$ is the Chebyshev polynomial.

Definition 2. Chebyshev polynomial-based computational Diffie-Hellman (*CPCDH*) problem. Given three elements, x, Chebyshev polynomials $T_r(x)$, and $T_s(x)$, compute the value $T_{rs}(x)$.

In contrast with the Chebyshev polynomials over the interval $(-\infty, +\infty)$, the hardness assumption of *CPCDH* problems over the interval $[-1,1]$ does not hold. Given three elements, x, $T_r(x)$, and $T_s(x)$, although it is computationally infeasible to derive r from the known x and $T_r(x)$, one can apply the method mentioned in [18, 44] to derive

$$r^* = \frac{\arccos(T_r(x)) + 2k\pi}{\arccos(x)} | k \in Z, \tag{5}$$

such that $T_r(x) = T_{r^*}(x)$. Thus, one can compute the Diffie-Hellman value $T_{r^*}(T_s(x)) = T_s(T_{r^*}(x)) = T_s(T_r(x))$.

Definition 3. The success probability of a probabilistic polynomial time Turing machine Δ within time upper bound t in solving *CPCDH* problems is defined as:

$$\mathrm{Adv}^{\mathrm{CPCDH}}(t) = \Pr[\Delta(T_r(x), T_s(x)) = T_{rs}(x)]. \tag{6}$$

Definition 4. The Chebyshev polynomial-based computational Diffie-Hellman assumption (*CPCDH assumption*) is the assumption that CPCDH problems are hard. In other words, for every probabilistic Turing machine Δ, $\mathrm{Adv}^{\mathrm{CPCDH}}(t)$ is negligible.

Definition 5. Integer factorization assumption (*IF assumption*) is the assumption that integer factorization is hard. In other words, the probability $\mathrm{Adv}^{\mathrm{IF}}(t')$ of integer factorization for any probabilistic polynomial time Turing machine within the time upper bound t' is negligible.

3.2. Notions of Privacy in TAKACP Protocols. According to the extent of user identity protection, we divide user privacy preserving into four levels: anonymity-alone, weak unlinkability, medium unlinkability, and strong unlinkability. Among these concepts, the latter is stronger than the former, i.e., anonymity-alone ≤ weak unlinkability ≤ medium unlinkability ≤ strong unlinkability.

Let N be the number of members of any user group. Let $A_{1,2}^{\mathrm{Guess}}$ be the event that A (that is, $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$ but $\boldsymbol{A_0}$) guesses the identity of the user correctly from the user group, A_2^{Decide} (A^{Decide}) be the event that $\boldsymbol{A_2}$ ($\boldsymbol{A}$) decides whether any two executions of the protocol are from the same user, respectively.

Definition 6. (Anonymity-Alone). We define the advantage $\mathrm{Adv}^{\mathrm{Anon-Alone}}(\mathrm{A})$ of any adversary A as

$$\mathrm{Adv}^{\mathrm{Anon-Alone}}(A) = \left(\Pr\left[A_{1,2}^{\mathrm{Guess}}\right] - \frac{1}{N}\right) + \left(\Pr\left[A_2^{\mathrm{Decide}}\right] - 1\right). \tag{7}$$

The advantage $\mathrm{Adv}^{\mathrm{Anon-Alone}}(\mathrm{A})$ measures the sum of the probability of any adversary $\mathbf{A_1}$ or any outside adversary $\mathbf{A_2}$ obtaining the identity of the user and the probability of any outside adversary $\mathbf{A_2}$ linking different sessions with a certain user.

An authenticated key agreement protocol is called to provide Anonymity-Alone if $\mathrm{Adv}^{\mathrm{Anon\text{-}Alone}}(A)$ *is negligible*. In other words, for any group of N users, $\boldsymbol{A}$ cannot identify the actual user with the probability higher than the probability $1/N$ of guessing. Hence, the first addition item would approach 0. However, any outside adversary $\boldsymbol{A_2}$ can link different sessions to a certain user.

Definition 7. (weak unlinkability). We define the advantage $\mathrm{Adv}^{\mathrm{Weak-Unlin}}(\mathrm{A})$ of any adversary A as

$$\mathrm{Adv}^{\mathrm{Weak-Unlin}}(A) = \left(\Pr\left[A_{1,2}^{\mathrm{Guess}}\right] - \frac{1}{N}\right) + \left(\Pr\left[A_2^{\mathrm{Decide}}\right] - \frac{1}{2}\right). \tag{8}$$

An authenticated key agreement scheme achieves weak unlinkability if $\mathrm{Adv}^{\mathrm{Weak\text{-}Unlin}}(\mathrm{A})$ is negligible. Specifically, any adversary $\mathbf{A_1}$ or any outside adversary $\boldsymbol{A_2}$ cannot obtain the identity of any other user. Besides, any outside adversary $\boldsymbol{A_2}$ cannot link different sessions to a certain user with a probability larger than 1/2.

Definition 8. (medium unlinkability). We define the advantage $\mathrm{Adv}^{\mathrm{medium-Unlin}}(\mathrm{A})$ of any adversary A (here, A_1 and A_2 but A_0) as

$$\mathrm{Adv}^{\mathrm{Medium-Unlin}}(A) = \left(\Pr\left[A_{1,2}^{\mathrm{Decide}}\right] - \frac{1}{2}\right). \tag{9}$$

An authenticated key agreement scheme is called to satisfy medium unlinkability if $Adv^{Medium\text{-}Unlin}(A)$ *is negligible*. In other words, any participant except the server cannot link different logins to the same user.

Definition 9. (strong unlinkability). We define the advantage $\mathrm{Adv}^{\mathrm{Strong-Unlin}}(\mathrm{A})$ of any adversary A including A_0, A_1, and A_2 as

$$\mathrm{Adv}^{\mathrm{Strong-Unlin}}(A) = \left(\Pr\left[A^{\mathrm{Decide}}\right] - \frac{1}{2}\right). \tag{10}$$

An authenticated key agreement scheme is called to satisfy strong unlinkability if $\mathrm{Adv}^{\mathrm{Strong-Unlin}}(\mathrm{A})$ is negligible.

4. Cryptanalysis of the Lin Takacp Protocol

In this section, we first tabulate the important notations in Table 2. We then review Lin's key agreement protocol [20].

Table 2: The notations.

Symbol	Description
S	The server
U	A user
PW	U's password
ID	U's identity
s	The master key of the server
x	A variable with value in [−1,1]
$\oplus$	Exclusive-OR operation
$h()$	A one-way hash function
$\|\|$	String concatenation
T_1, T_2, T_3	The timestamps
t	A random integer
SK	Session key
$E_k()$	Symmetric encryption algorithm with k
$D_k()$	Symmetric decryption algorithm with k

4.1. Brief Review of Lin's Takacp Protocol. The Lin's TAKACP scheme [20] is composed of four algorithms: system initialization, user registration, authenticated key exchange, and password change. The notations used in [20] are listed in Table 2. Figures 1–3 separately illustrate the phases of user registration, authenticated key exchange, and password change.

4.1.1. System Initialization. The server S selects a master key s. Then, S computes a Chebyshev polynomial of degree r, i.e., $T_r(x)$, where $x \in [-1, 1]$, and chooses a one-way hash function $h()$ and a symmetric encryption function $E_k()$ with the secret key k. S keeps r secret.

4.1.2. User Registration Phase. A user registration procedure consists of the following steps.

Step 1. The user U selects an identity ID, a password PW, and a random integer t. U computes $H = h(PW||t)$ and then sends the message $\{ID, H\}$ to the server via a secure channel.

Step 2. Upon receiving the registration request, the server S computes $R = E_s(ID||H)$, $D = H \oplus (x||T_r(x))$. Then, S writes $\{R, h(), E_k(), D\}$ to a smart card and sends the smart card to U via a secure channel.

Step 3. Upon receiving the smart card, U stores t into it.

4.1.3. Authenticated Key Exchange Phase. U first enters the identity ID and password PW. Then, the smart card runs the following steps on S:

Step 1. It chooses a random integer j and computes

$$\begin{aligned} \left(x \| T_r(x)\right) &= D \oplus h(\mathrm{PW} \| t), \\ v &= T_j(T_r(x)), \\ Q &= h(\mathrm{ID} \| H). \end{aligned} \tag{11}$$

Then, it issues the message $\left\{T_j(x), E_v(Q, R, T_1)\right\}$ to S.

Step 2. S computes $v = T_r(T_j(x))$ and decrypts $E_v(Q, R, T_1)$. Then, S checks the validity of the time

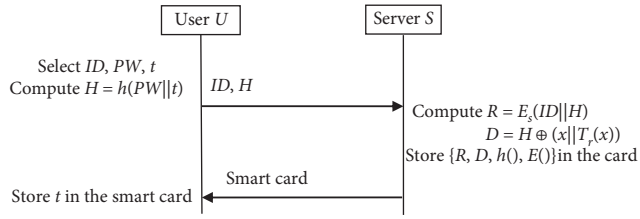

Figure 1: User registration phase of the Lin protocol.

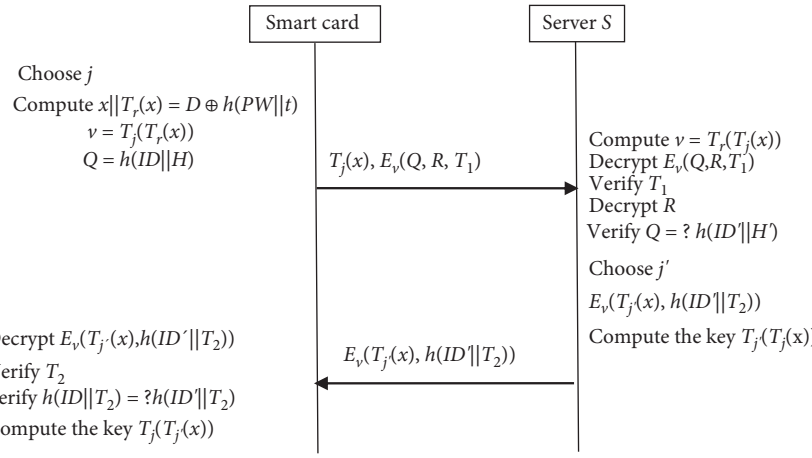

Figure 2: Authenticated key exchange phase of the Lin protocol.

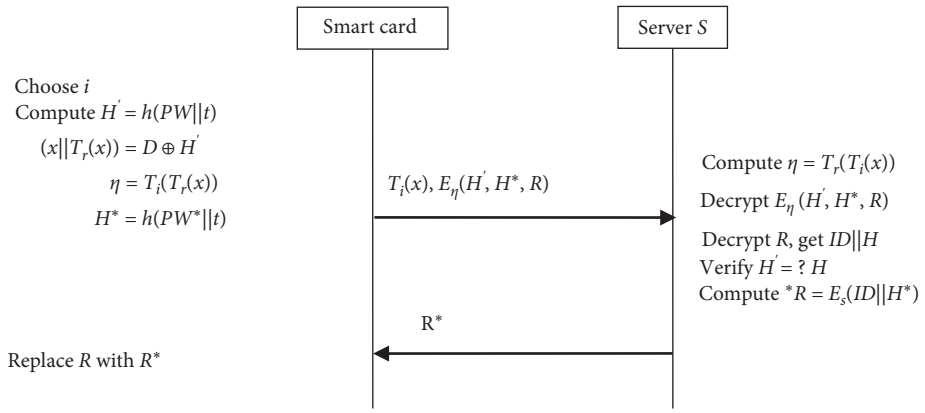

Figure 3: Password change phase of the Lin protocol.

stamp T_1. Next, S decrypts R and verifies whether $Q = h(\mathrm{ID}'\|H'))$ holds. If the equation holds, U is authenticated; otherwise, the session is terminated.

Step 3. S chooses a random integer j' and returns the login response $E_v(T_{j'}(x), h(\mathrm{ID}'\|T_2)))$ to the card. And, S computes session key $T_{j'}(T_j(x))$.

Step 4. The card first decrypts $E_v(T_{j'}(x), h(\mathrm{ID}'\|T_2)))$ and then checks whether the delay time for T_2 is acceptable. Next, the card checks whether $h(\mathrm{ID}\|T_2) = h(\mathrm{ID}'\|T_2))$ holds. If the equation holds, S is authenticated. And, the card computes the session key $T_j(T_{j'}(x))$.

4.1.4. Password Change Phase. U first inserts the smart card into a terminal and inputs his or her identity *ID*, the old password *PW*, and a new password PW*. Then, the smart card runs the following steps on S:

Step 1. The smart card chooses randomly a positive integer i and calculates

$$\begin{aligned} H' &= h(PW\|t), \\ \left(x\|T_r(x)\right) &= D\oplus H', \\ \eta &= T_i\left(T_r(x)\right), \\ H^* &= h\left(\mathrm{PW}^*\|t\right), \end{aligned} \tag{12}$$

and delivers $\{T_i(x), E_\eta(H', H^*, R)\}$ to S.

Step 2. S computes $\eta = (T_r(T_i(x)))$, then decrypts $E_\eta(H', H^*, R)$ and further $R = E_s(ID||H)$. Next, S checks whether the received H' is equal to H. If the equation holds, the server returns $R^* = E_s(\mathrm{ID}\|H^*)$ to the card.

Step 3. The smart card replaces R with R^*.

4.2. Security Weaknesses of Lin's TAKACP Protocol. Lin [20] demonstrates that the Guo-Chang TAKACP scheme [19] cannot provide full protection for the user's identity. Any passive inside adversary $\boldsymbol{A_1}$ (i.e., a malicious registered user) could derive the mutually shared session key between the user and the server only by intercepting the transmitted message. Lin claimed that their improvement eliminates the drawbacks. We show that the second security weakness of the Guo-Chang TAKACP scheme still exists in the Lin scheme [20]. In addition, the password change would not only bring inconvenience to the user but also lack the authentication of the server.

4.2.1. Violation of the Session Key Security. Assume that an inside adversary $\boldsymbol{A_1}$ has intercepted the key exchange message transmitted between the user U and the server. In the Lin scheme, the adversary could derive the session key by performing the following steps.

Since $\boldsymbol{A_1}$ is an inside adversary, $\boldsymbol{A_1}$ can use his password to calculate $x||T_r(x)$. After intercepting U's login message $\{T_j(x), E_v(Q, R, T_1)\}$, $\boldsymbol{A_1}$ has obtained $T_j(x)$. Although it is computationally infeasible to calculate j from x and $T_j(x)$, the adversary can apply the method mentioned in [18, 44] to compute

$$j^* = \frac{\arccos\left(T_j(x)\right) + 2k\pi}{\arccos(x)} | k \in Z, \tag{13}$$

such that $T_j(x) = T_{j^*}(x)$. Then, the adversary could compute the key $v = T_{j^*}(T_r(x)) = T_r(T_{j^*}(x)) = T_r(T_j(x))$.

$\boldsymbol{A_1}$ decrypts $E_v(T_{j'}(x), h(\mathrm{ID}'\|T_2)))$ with the key v and obtains $T_{j'}(x)$. Finally, the adversary calculates the session key $\mathrm{SK} = T_{j^*}(T_{j'}(x)) = T_{j'}(T_{j^*}(x)) = T_{j'}(T_j(x))$.

4.2.2. Suffering from User Impersonation Attack. Suppose that an inside adversary $\boldsymbol{A_1}$ does not want to pay the server for the service provided by S. $\boldsymbol{A_1}$ would try to impersonate a legitimate user U. After intercepting U's login message, the adversary launches the user impersonation attack as described below:

(1) With the login message $\{T_j(x), E_v(Q, R, T_1)\}$, $\boldsymbol{A_1}$ computes the key v through the same technique as in the leakage attack of the session key in Section 3. Then, $\boldsymbol{A_1}$ decrypts $E_v(Q, R, T_1)$ and obtains $\{Q, R, T_1\}$.

(2) $\boldsymbol{A_1}$ chooses a random integer j and computes $v = T_j(T_r(x))$. It transmits $\{T_j(x), E_v(Q, R, T_1)\}$ to the server S, where T is the current timestamp.

(3) After receiving the login message, S computes $v = T_r(T_j(x))$ and decrypts $E_v(Q, R, T)$. S first checks the validity of the time stamp T. Next, S decrypts R to derive $ID||H$, and verifies the identity. Since $Q = h(ID||H)$, the server believes that a legitimate user with ID has issued the login request. After that, S selects a Chebyshev polynomial $T_{j'}(x)$, encrypts it with other messages, and transmits the cipher-text to $\boldsymbol{A_1}$.

(4) $\boldsymbol{A_1}$ recovers the map $T_{j'}(x)$ with the key v and computes the session key.

4.2.3. Linking Different Sessions to a Same User. The Lin's TAKACP scheme enhances the protection of user identity. Although any adversary $\boldsymbol{A_1}$ or $\boldsymbol{A_2}$ cannot obtain the identity of any user, any inside adversary $\boldsymbol{A_1}$ can link different sessions to a certain user. The Lin scheme only can provide weak unlinkablity (for details, see Definition 7 in Section 3), since $\boldsymbol{A_1}$ may execute the linking of sessions to the user as follows.

(1) Intercept the different login messages $\{T_j(x), E_v(Q, R, T_1)\}$.

(2) Compute the different keys v through the approach described in Section 3.

(3) Decrypt $E_v(Q, R, T_1)$ and obtain $\{Q, R, T_1\}$. For the user, although the parameters $\{R, T_1\}$ change with different logins, Q will be unchanged. Thus, $\boldsymbol{A_1}$ can decide whether the users are the same by comparing the parameter Q's.

4.2.4. Defects of the Password Change. Firstly, the password change suffers from inside attacks. Consider that a registered user U' acts as an adversary $\boldsymbol{A_1}$. Since U' is a registered user of the server S, U' can derive $x||T_r(x)$ from his own (D', H') by using his own password. Assume that U' has intercepted U's password change request $\{T_i(x), E_\eta(H', H^*, R)\}$. As shown above, U' can calculate $i^* = ((\arccos(T_i(x)) + 2k\pi)/\arccos(x))(k \in Z)$, which satisfies the equation $T_i(x) = T_{i^*}(x)$. Thus, U' computes the key η and further decrypts $E_\eta(H', H^*, R)$. Then, U' selects randomly a value H'' of the same length of H^*. And U' computes $E_\eta(H', H'', R)$ and sends $\{T_i(x), E_\eta(H', H'', R)\}$ to the server. Since the equation $H' = H$ holds, the server will return $R^* = E_s(\mathrm{ID}\|H'')$ to the smart card. The smart card stores R^* instead of R. Thus, the password change has been fulfilled. However, the user U cannot login to the server any more with the new password PW^*. We describe the failure process as follows. When the user U tries to login to S, U computes $Q = h(\mathrm{ID}\|H^*)$ where $H^* = h(\mathrm{PW}^*\|t)$ and then issues $(T_j(x), E_v(Q, R^*, T_1))$ to the server. The server decrypts $E_v(Q, R^*, T_1)$ and acquires R^*. S further decrypts R^* to obtain $\{\mathrm{ID}, H''\}$. S computes $Q' = h(\mathrm{ID}\|H'')$. Since $Q' \neq Q$, the server will refuse U's login request.

Secondly, the Lin's TAKACP scheme requires that the server participate during the whole password change phase. In many applications, registered users always need to update

their passwords at intervals. Passwords should be freely updated by the smart card holder at will without any interaction with the server, while the server can be totally unaware of the password change. A TAKACP scheme should provide the users with free password changes. If the users' password change requires the server online, it must be a bottleneck. The Lin scheme requires the server S to compute R^* during the password change phase. Therefore, it is inconvenient to both the server and the users. For the Lin scheme, the password change is impractical.

Thirdly, during the password change phase, the server is not authenticated by the smart card. This would be a serious security drawback. Any adversary could impersonate the server and send an arbitrary value as R^*. The smart card will replace R with R^*. The real card holder cannot login to the server any longer since $R^* \neq E_s(\mathrm{ID}\|H^*)$. If the password change proceeds through a secure channel as in the user registration phase, the above security drawbacks will be removed. However, this is also infeasible.

5. The Proposed Takacp Protocol

Lin [20] showed that the Guo-Chang scheme suffers from inside attacks. The analysis above demonstrates that the Lin' TAKACP scheme [20] cannot still resist against inside attacks. The main cause is that an inside adversary $\boldsymbol{A_1}$ has the common chaotic map $T_r(x)$ with the registered users of the same server. After intercepting the chaotic map $T_j(x)$ transmitted over the public channel, $\boldsymbol{A_1}$ can derive an integer j^* satisfying $T_j(x) = T_{j^*}(x)$. The adversary computes the key v to decrypt $E_v(T_{j'}(x), h(\mathrm{ID}\|T_2), T_2)$ and derive $T_{j'}(x)$. Thus, the adversary can determine the Diffie-Hellman-like session key $T_{j^*}(T_{j'}(x))$.

To eliminate these weaknesses, we will seek cryptographic techniques to protect the functions $T_j(x)$ and $T_{j'}(x)$. In the following, we will use the quadratic residues to present two improved versions. We first describe an improved two-factor authentication scheme (hereafter called TAKACP-1) based on Chebyshev polynomials defined on the interval [−1,1] and then another two-factor authentication scheme (hereafter called TAKACP-2) based on Chebyshev polynomials defined on the interval (−∞,+∞).

5.1. Registration Phase. We adopt the same notations as those in the Lin scheme. The server S selects s as the symmetric encryption key, two distinct large primes p and q with $p \equiv q \equiv 3 \pmod 4$, and a one-way hash function $h()$: $\{0,1\}^* \longrightarrow \{0,1\}^l$ where l is a security parameter. The parameters p, q, and s are kept secret. Before a user U gains access to the server S, U must register by performing the following steps as shown in Figure 4.

Step R1. U selects a random integer n' with l bits, an identity ID, and password PW. Then, U computes $d = h(\mathrm{ID}\|\mathrm{PW})n'$ and delivers $\{ID, d\}$ to S through a secure channel.

Step R2. S computes $c = d \oplus E_s(ID||n)$, where $n = pq$. S stores $\{c, n, h()\}$ on a smart card and issues the smart card to U via a secure channel.

Step R3. The card computes.

5.2. Authenticated Key Exchange PHASE. The user U and the server S cooperatively perform the following steps to generate a session key SK, which is also illustrated in Figure 5.

Step A1. U inserts the smart card into the terminal and enters the identity ID and the password PW. The smart card checks whether d_2 is equal to $h((ID \oplus PW)||d_1||n)$. If ID and PW are valid, it computes $c = d_1 \oplus h(ID||PW)$. It generates a nonce n_1 of the c's binary length and randomly chooses a positive integer i and a real number x over [−1,1]. The card computes the Chebyshev polynomials $T_i(x)$, $e = (cn_1)^2 \bmod n$, $w_1 = h(n_1) \oplus (x\|T_i(x))$, $c_0 = h(ID||n_1)$, $\theta_1 = h(c\|x\|T_i(x)\|c_0)$, where $x||T_i(x)$ *is of l-bits*. The symbol means that the binary form of c interleaves the binary form of n_1 bit by bit. Then, the card transmits the message $M_1 = \{e, w_1, \theta_1\}$ to the server S.

Step A2. Once receiving the login request, S uses Chinese Remainder theorem with p and q to solve the square roots of e. S parses the four roots into two parts, c', n_1', respectively. Then, S decrypts c' to obtain (ID^*, n^*) and determines the right root by checking if n^* is equal to n. Finally, S obtains the right root $(c'n_1')$ and the identity ID^*. S computes $(x\|T_i(x)) = h(n_1') \oplus w_1$, $c_0^* = h(\mathrm{ID}^*\|n_1')$, and checks if the received θ_1 equals $h(c'\|x\|T_i(x)\|c_0^*)$. If the equation holds, the server believes that the login comes from a registered user with the identity ID^*. S generates a nonce n_2 such that $n_2||T_j(x)$ is l-bit in length and randomly chooses a positive integer j. Then, S computes

$$\begin{aligned} w_2 &= h\left(\mathrm{ID}^*\|n_1'\right) \oplus \left(n_2\|T_j(x)\right), \\ \mathrm{SK} &= h\left(\mathrm{ID}^*\|T_i(x)\|T_j(x)\|n_1'\|n_2\|T_j(T_i(x))\right), \\ \theta_2 &= h\left(c_0^*\|\left(T_j(x)\|x\right) \oplus h(n_2)\|\mathrm{SK}\right). \end{aligned} \tag{14}$$

And, S sends back the message $M_2 = \{w_2, \theta_2\}$ to U. Otherwise, S rejects the login request from U.

Step A3. Upon receipt of the response message from S, the smart card computes $(n_2\|T_j(x)) = w_2 \oplus h(\mathrm{ID}\|n_1)$, $SK^* = h(ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))$. Next, it checks whether θ_2 equals $h(c||(T_j(x)||x) \oplus h(n_2)||SK^*)$. If they are equal, the card authenticates the server. It computes $\theta_3 = h(\mathrm{SK}^*\|n_1\|n_2\|c_0)$ and forwards the message $M_3 = \{\theta_3\}$ to the server S. Otherwise, U terminates the session.

Step A4. After receiving the confirmation message from the card, S checks if $h(\mathrm{SK}\|n_1'\|n_2\|c_0^*)$ equals θ_3. If they are equal, the user U with identity ID is authenticated. Moreover, S confirms the session key SK.

5.3. Password Change Phase. If the user U wants to update his password, U performs the following steps.

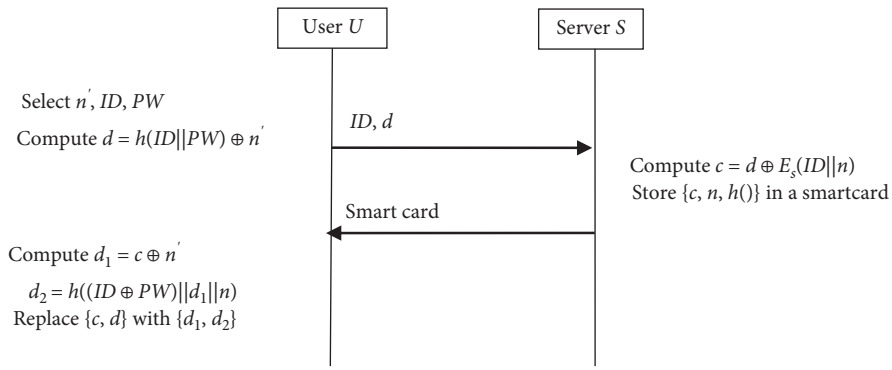

Figure 4: User registration phase of the proposed TAKACP-1 protocol.

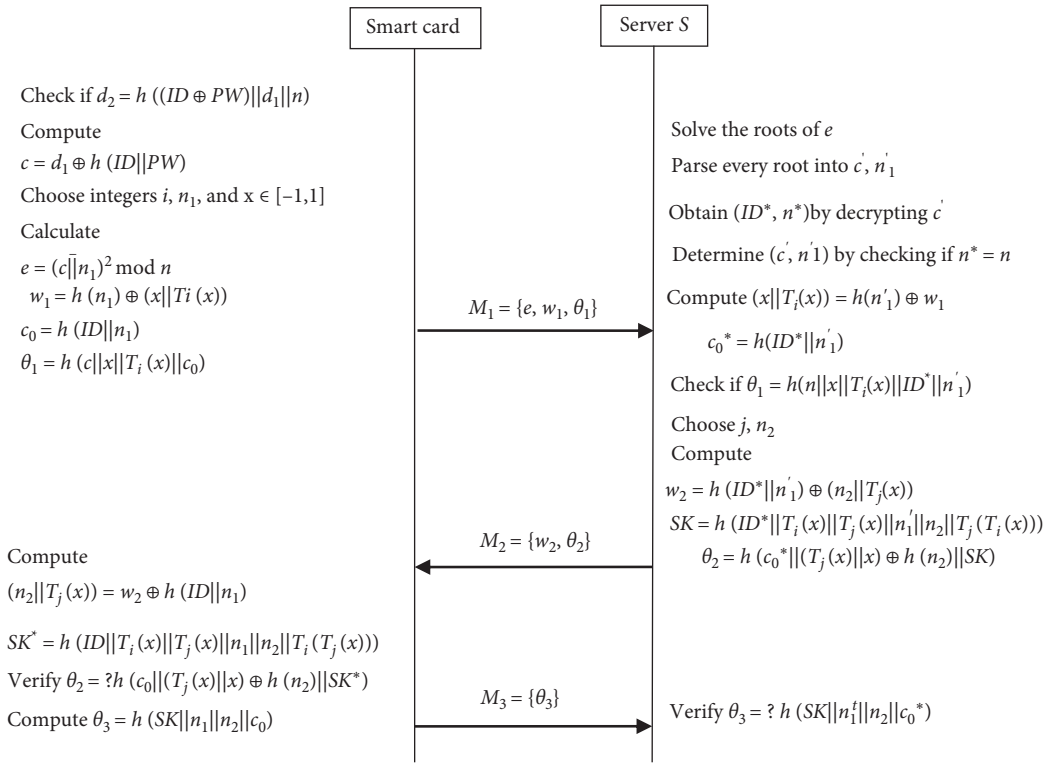

Figure 5: Authenticated key exchange phase of the proposed TAKACP-1 protocol.

Step C1. U inserts the smart card into the terminal and inputs the identity ID and the old password PW. Then, U issues the *updating* request.

Step C2. The smart card checks whether $d_2 = h((ID \oplus PW)||d_1||n)$ holds. If the equation holds, it answers *accepting updating*.

Step C3. U submits a new password PW_{new}.

Step C4. The smart card computes

$$\begin{aligned} d_1' &= d_1 \oplus h(\text{ID}\|\text{PW}) \oplus h\left(\text{ID}\middle\|\text{PW}_{\text{new}}\right), \\ d_2' &= h\left(\left(\text{ID}\middle\|\text{PW}_{\text{new}}\right)d_1'\|n\right). \end{aligned} \tag{15}$$

The card removes $\{d_1, d_2\}$ and stores $\{d_1', d_2'\}$.

Now, we describe briefly the TAKACP-2 scheme. There is a little difference between the registration phase of the TAKACP-2 scheme and that of the TAKACP-1 scheme. We will give the detailed description of the registration phase of TAKACP-2 scheme. Another fundamental difference of the authenticated key exchange phase of *TAKACP-2* scheme from *TAKACP-1* scheme is that the real number x is drawn from the interval $(-\infty, +\infty)$. The card/server computes the Chebyshev polynomial $T_i(x)/T_j(x)$ mod p_0. By making similar modifications to the registration phase, we can have the password change phase. Here, we omit the description of the authenticated key exchange phase and the password change phase in the TAKACP-2 scheme.

5.4. Registration Phase of TAKACP-2. We adopt the same notations as those in the *TAKACP-1* scheme. The server S selects a large prime p_0 besides the symmetric encryption key s, two large primes p, q, and a one-way hash function $h()$:

$\{0,1\}^* \longrightarrow \{0,1\}^l$. S keeps p, q, and s secret. Then, U performs the following steps to execute the registration.

Step R1'. U selects a random integer n', an identity ID, and password PW. Then, U computes $d = h(\mathrm{ID}\|\mathrm{PW})n'$ and delivers $\{ID, d\}$ to S through a secure channel.

Step R2'. S computes $c = d \oplus E_s(ID||n||p_0)$ where $n = pq$. S stores $\{c, n, p_0, h()\}$ on a smart card and issues the smart card to U via a secure channel.

Step R3'. The card computes

$$d_1 = c \oplus n',$$
$$d_2 = h\big((\mathrm{ID} \oplus \mathrm{PW})\|d_1\|(n \oplus p_0)\big). \tag{16}$$

Then, U removes $\{n', c, d\}$ and stores $\{d_1, d_2\}$ in the card.

6. Security Analyses

In this section, we will present the formal semantic security analysis of the proposed protocols under the random oracle model in Part A. Mutual authentication between a user and a server will be confirmed through the widely used BAN logic [46–48] in Part B. In Part C, we conduct the detailed informal security analysis of the proposed protocol. The formal security analysis and informal security analysis both show that our schemes provide stronger security attributes.

6.1. Formal Security Analysis in Random Oracle Model. In this subsection, we introduce a formal security model under the widely used Real Or-Random model [49], the authentication security model [50], and the sequence of game models [51].

Assume that the server is a trustworthy entity. The server can accept the registration of users and validate the real identity of the users to provide them with services. There exists a secure channel between the server and the user to protect the registration of the user. In the following, we will apply the Dolev-Yao threat model (DY model) [52] to analyze the security of the proposed schemes. According to DY model, any two communicating parties communicate over an insecure channel. Assume that a polynomial time adversary has the ability to control the communication channel, for example, modifying, injecting, monitoring, and deleting messages over the open channel. Any adversary $\boldsymbol{A}$ can make oracle queries, which model the adversary's capabilities in a real attack. The goal of the adversary is to penetrate the anonymous authentication of a key agreement protocol by compromising requirements for the protocols described below. A malicious registered user may act as $\boldsymbol{A}$ to attempt to obtain the identity information of other users who have registered on the same server.

We will simulate various security attacks on the proposed schemes through all possible oracle queries listed below.

Execute(U^i, S^j): This query models eavesdropping attacks on honest execution among the user instance U^i and the server instance S^j. The output of this query consists of the messages that were exchanged during the honest execution of the protocol.

Send(U^i/S^j,m): This query models an active attack. The oracle query enables $\boldsymbol{A}$ to receive an actual response from a participant U^i/S^j. Specifically, the adversary $\boldsymbol{A}$ sends a message m to instance U^i/S^j, and the participant instance U^i/S^j follows the protocol to give a reply.

Reveal(U^i/S^j): This query models known session key attacks. If no session key is defined for an instance U^i/S^j, or if either U^i/S^j, or its partner is asked a *Test* query, the output of this query is the invalid symbol $\perp$. Otherwise, it returns the current session key SK, which has been established between U^i/S^j and its partner to $\boldsymbol{A}$.

Corrupt(U,a): The query models the capability of $\boldsymbol{A}$ to obtain the secret information of a user participant U, thereby corrupting the protocol.

If $a = 1$, the query returns U's password to $\boldsymbol{A}$.

If $a = 2$, the query returns the message stored in the user U's smart card with $\boldsymbol{A}$. The oracle simulates that when $\boldsymbol{A}$ gains the smart card of user U, it can extract the secret stored information.

Test(U^i/S^j): If no session key is defined, for instance, U^i/S^j or if either U^i/S^j or its partner is asked a *Reveal* query, the output of this query is the invalid symbol $\perp$. Otherwise, the oracle flips a coin b. If $b = 1$, the output is the session key. Otherwise, the output is a random string drawn from the space of session keys. The *Test* query is invoked once by the adversary with a fresh oracle. The query is used to define the semantic security of the session key SK.

Definition 10. An instance U^i/S^j is called to *be accepted*, if upon receiving the last expected protocol message, it goes into an accept state. The ordered concatenation of all sent and received messages by instance U^i/S^j forms the session identification(sid) of U^i/S^j for the current session.

Definition 11. Two instances U^i and S^j are said to *be partnered* if the three conditions hold simultaneously: (1) both U^i and S^j are accepted; (2) both U^i and S^j share the same sid; and (3) U^i and S^j are mutual partners of each other.

Definition 12. An instance U^i/S^j is called to *be fresh* if the following conditions are fulfilled simultaneously: (1) U^i/S^j is in the accepting state; (2) Reveal(U^i/S^j) query has never been submitted to U^i/S^j or its partner; and (3) strictly fewer than two Corrupt(U^i,a) queries have been made to U^i or strictly fewer than two Corrupt(U^i,a) queries have been submitted to S^j's partner U^i.

Definition 13. For the *semantic security*, the security model is defined by a game, which consists of two phases. In the first phase, an adversary A is allowed to adaptively issue

Send, Execute, Reveal, and Test queries. In the second phase, the adversary A executes a single Test (U^i/S^j) query with the chosen bit b directed to a fresh instance and the query outputs a guess bit b' for b. If $b' = b$, then the adversary A wins the above game, i.e., A succeeds in breaking the semantic security of the game of a TAKACP protocol. Let Succ(A) be an event where the adversary A wins the above game. The advantage of the adversary A in breaking the semantic security of the TAKACP protocol is defined by

$$\mathrm{Adv}^{\mathrm{TFAKA}}(A) = \left|\Pr\left(\mathrm{Succ}(A) - \frac{1}{2}\right)\right| = \left|\Pr(b' - b) - \frac{1}{2}\right|. \tag{17}$$

A TAKACP protocol is said to be *semantically secure* if the advantage $\mathrm{Adv}^{\mathrm{TFAKA}}(A)$of any probabilistic polynomial time-bounded adversary A is negligible.

Theorem 1. *Let D(W) be a uniformly distributed password (identity) dictionary of size $|D|(|W|)$. Let A (including A_1 and A_2) be a polynomial time-bound adversary against the semantic security of the TAKACP-2 scheme. Suppose A makes at most q_s times Send queries, q_e times Execute queries, and q_h times hash oracle queries. Then, we have*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{TFAKA-2}}(A) \le{}& \frac{\left(q_s^2 + q_e^2\right)}{2^{n+1}} + \frac{\left(q_s^2 + q_e^2\right)}{2(p_0 - 1)} + \frac{q_h^2}{2^{l+1}} \\ &+ \frac{3q_h}{2^{l-1}} + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} + \frac{q_s}{2^{3l}} + \frac{q_s}{|D| + |W|} \\ &+ q_h \mathrm{Adv}^{\mathrm{CPCDH}}(t) + \mathrm{Adv}^{\mathrm{IF}}(t')\frac{q_s}{|W|},\end{aligned} \tag{18}$$

where l refers to the string length of hash results, $Adv^{CMCDH}(t)$ is the success probability of any probabilistic polynomial time Turing machine within time upper bound t in solving CPCDH problems, and $Adv^{IF}(t')$ is the probability of integer factorization for any probabilistic polynomial time Turing machine within time upper bound t'.

Proof. We shall use the approach of sequent games to prove this theorem. We first define a sequence of modified attack games G_i ($i = 0,1,2,3,4,5$) starting from G_0 and terminating at G_5. Let $Succ_i$ be an event defined as the successful guessing of the bit b in *Test* query corresponding to each game G_i by an adversary A.

Game G₀: This starting game and the real protocol in random oracles are assumed to be identical. Hence, G_0 is the actual attack game. By definition, we have

$$\mathrm{Adv}^{\mathrm{TFAKA-2}}(A) = \left|\Pr[\mathrm{succ}_0] - \frac{1}{2}\right|. \tag{19}$$

Game G₁: This game is the same as the game G_0 except that the game simulates all oracle queries including *Send*, *Reveal*, *Corrupt*, *Execute*, *Test*, and *hash* queries. The hash oracles and *Reveal*, *Test*, *Corrupt*, and *Execute* queries are simulated in Table 3. We simulate the *Send* queries in Table 4 as in the actual attack game. The simulations maintain three lists of queries: (1) list L_h records the answers to hash oracles, (2) list L_A records the answers to the queries which are initiated by A, and (3) list L_T records the transcripts between S and U.

This game is perfectly indistinguishable from the real execution of the protocol. Hence, we have

$$\Pr[\mathrm{succ}_1] = \Pr[\mathrm{succ}_0]. \tag{20}$$

Game G₂: In this game, we consider collisions among the results of hash queries, random numbers, and Chebyshev polynomials in the transcripts of messages M_1, M_2, and M_3. We take the random value h from $\{0,1\}^l$ as the response of the hash queries. If this query is directly asked by the adversary and $(*, h) \leftarrow L_h$, we abort the game. Otherwise, h is returned. Following the birthday paradox, the probability of collisions of the oracle hash query is at most $(qh^2/2^{l+1})$. Furthermore, the messages contain random numbers $\{n_1, n_2\}$ and two Chebyshev polynomials $\{T_i(x), T_j(x)\}$. And, the probability of random numbers and polynomials collision is at most $((q_s^2 + q_e^2)/2^{n+1}) + ((q_s^2 + q_e^2)/2(p_0 - 1))$. Games $\mathbf{G_2}$ and $\mathbf{G_1}$ are perfectly indistinguishable except that the abovementioned collision causes the game to abort. Hence, we have

$$\left|\Pr(\mathrm{Succ}_2) - \Pr(\mathrm{Succ}_1)\right| \le \frac{\left(q_s^2 + q_e^2\right)}{2^{n+1}} + \frac{\left(q_s^2 + q_e^2\right)}{2(p_0 - 1)} + \frac{q_h^2}{2^{l+1}}. \tag{21}$$

Game G_3: This game would abort the execution in the situation where A obtains a valid authenticator without active participation of hash oracles. In the *TAKACP-2* protocol, the authenticated key exchange phase involves three message communications, M_i, $i = 1,2,3$. We consider three cases, *Send*(U, M_1), *Send*(S, M_2), *Send*(U, M_3) in the game $\boldsymbol{G}_3$.

Case 1. Considering *Send*(U, M_1) oracle query, we must carefully analyze the elements of message M_1. The hash values $h(c||x||T_i(x)||c_0) \in L_A$ must hold, otherwise the session will be terminated. The maximum calculated probability is up to $(q_h/2^l)$. Again, it must be that $h(ID||n_1) \in L_A$ whose probability is at most $(q_h/2^l)$. Finally, the message $M_1 \in L_T$ should hold, or the session will stop. For this, the probability is $(q_s/2^{3l})$.

Case 2. Considering *Send*(S, M_2) oracle query, M_2 consists of w_2 andθ_2. The hash values $h(c_0^* \| (T_j(x) \oplus h(n_2)) \| h(\mathrm{ID}^* \| T_i(x) \| T_j(x) \| n_1' \| n_2 \| T_j(T_i(x)))) \in L_A$ must hold; otherwise, the session will be terminated. The maximum probability is up to $(q_h/2^l)$. The probability of value $h(\mathrm{ID}^* \| T_i(x) \| T_j(x) \| n_1' \| n_2 \| T_j(T_i(x)))$ falling within the list L_A is at most $(q_h/2^l)$. Finally, the message M_2 should fall within L_T, or the session will terminate. The maximum probability is $(q_s/2^{2l})$.

Table 3: Simulation of hash, reveal, test, corrupt, and execute oracle queries.

(i) *Hash* simulation query performs as follows:
If the record (*; h) is found in the list L_h corresponding to the hash query $h(*)$, return the hash function h. Otherwise, select a string $h \in \{0,1\}^l$ and add (*; h) into L_h. If the query is initiated by A, (*; h) is stored in L_A.
(ii) *Reveal*(U^i/S^j) simulation query performs as follows:
If U^i/S^j is in the accepting state, the current session key SK formed by U^i/S^j and its partner is returned.
(iii) *Test*(U^i/S^j) simulation query performs as follows:
Through *Reveal*(U^i/S^j) query, obtain the current session SK and then flip an unbiased coin b. If $b = 1$, return SK. Otherwise, return a random string from $\{0,1\}^l$.
(iv) *Corrupt*(U, a) simulation query performs as follows:
If $a = 1$, the query returns the password PW of U. If $a = 2$, the query returns the secret information stored in the user smart card.
(v) Simulation of *Execute*(U^i, S^j) query occurs in succession with the simulation of *Send* queries as shown below.
$\{e, w_1, \theta_1\} \leftarrow$Send($U^i$; **start**), $\{w_2, \theta_2\} \leftarrow$Send($S^j$; $\{e, w_1, \theta_1\}$) and $\{\theta_3\} \leftarrow$Send(U^i; $\{w_2, \theta_2\}$). Finally, $M_1 = \{e, w_1, \theta_1\}$, $M_2 = \{w_2, \theta_2\}$, and $M_3 = \{\theta_3\}$ are returned.

Table 4: Simulation of *send* oracle queries.

(i) On a query Send(U^i; start), assuming U^i is in the correct state, we proceed as follows:
Choose a positive integer i and a real number x over $(-\infty, +\infty)$ and compute $T_i(x)$, $e = (cn_1)^2 \bmod n$, $w_1 = h(n_1) \oplus (x \| T_i(x))$, $\theta_1 = h(c||x||T_i(x)||h(ID||n_1))$. Then, the answer $\{e, w_1, \theta_1\}$ to the query is returned.
(ii) On a query Send(S^j; $\{e, w_1, \theta_1\}$), assuming S^j is in the correct state, we proceed as follows:
Solve the square roots of e and obtain c', n_1'. Decrypt c' and get (ID^*, n^*). Compute $(x \| T_i(x)) = h(n_1') \oplus w_1$, $c_0{}^* = h(ID^* || n_1')$, and check if the received $\theta_1 = h(c'||x||T_i(x)||c_0{}^*)$. If the equation does not hold, the server instance terminates without accepting. Otherwise, choose randomly a nonce n_2, a positive integer j, and compute $w_2 = h(\mathrm{ID}^* \| n_1') \oplus (n_2 \| T_j(x))$, $SK = h(\mathrm{ID}^* || T_i(x) || T_j(x) || n_1' || n_2 || T_j(T_i(x)))$, $\theta_2 = h(c_0{}^* || (T_j(x) \oplus h(n_2)) || SK)$. Then, the answer $\{w_2, \theta_2\}$ to the query is returned.
(iii) On a query Send(U^i; $\{w_2, \theta_2\}$), assuming U^i is in the correct state, we proceed as follows:
Compute $(n_2 \| T_j(x)) = w_2 \oplus h(\mathrm{ID} \| n_1)$, $\mathrm{SK}^* = \mathrm{h}(\mathrm{ID}||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))$, and check whether $\theta_2 = \mathrm{h}(\mathrm{c}||(T_j(\mathrm{x}) \oplus h(n_2))||\mathrm{SK}^*)$. If the equation does not hold, the user instance terminates without accepting. Otherwise, compute $\theta_3 = \mathrm{h}(\mathrm{SK}^*||(n_1 \oplus n_2)||c_0)$, authenticate S^j, and establish SK as the session key. Then, the answer $\{\theta_3\}$ to the query is returned.
(iv) On a query Send(S^j; $\{\theta_3\}$), assuming S^j is in the correct state, we proceed as follows:
Check if $\theta_3 = h(SK||(n_1' \oplus n_2)||c_0^*)$. If the equation does not hold, the server instance terminates without accepting and aborts the session. Otherwise, S accepts the session key SK.

Case 3. To respond *Send*(U, M_3) oracle query, $h(h(ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))||(n_1 \oplus n_2)||c_0) \in LA$ must hold with the total maximum probability $(q_h/2^l)$. Finally, for a transcript $M_3 \in L_T$, we have the maximum probability as $(q_s/2^l)$.
Considering the three cases, we have,

Game G_4: In this game, when the session key SK is required to compute, we replace the random hash oracle H_1 with private oracle H'. That is, the session key is determined without querying the hash oracle. Moreover, we do not use $T_j(T_i(x))$ or $T_i(T_j(x))$ to compute $SK = H'(ID||T_i(x)||T_j(x)||n_1||n_2)$. Thus, the session key is completely independent of *hash oracle* and $T_j(T_i(x))$ or $T_i(T_j(x))$. Games $\mathbf{G_4}$ and $\mathbf{G_3}$ are perfectly indistinguishable unless the following event AskH_1 occurs: the adversary A queries the hash function on $ID||T_i(x)||T_j(x)||n_1||n_2||T_j(T_i(x))$ or on the message $ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x))$. Hence, we have

$$\left|\Pr(\mathrm{Succ}_4) - \Pr(\mathrm{Succ}_3)\right| \le \Pr[\mathrm{AskH}_1]. \tag{23}$$

Game G_5: In this game, we simulate the executions using the random self-reducibility of the CPCDH problem, given one CPCDH instance $(T_i(x), T_j(x))$. We choose randomly two integers $u, v$and compute $T_u(T_i(x))$, $T_v(T_j(x))$. The event AskH_2 means that the adversary A had queried the random hash oracle H_1 on $ID||T_i(x)||T_j(x)||n_1||n_2||Z$, where $Z = \mathrm{CPCDH}(T_u(T_i(x)), T_v(T_j(x)))$. It is easy to know that the equation CPCDH $(T_u(T_i(x)), T_v(T_j(x))) = T_{uv}(\mathrm{CPCDH}(T_i(x), T_j(x)))$ holds. We have

$$\begin{aligned} &\left|\Pr(\mathrm{Succ}_5) - \Pr(\mathrm{Succ}_4)\right|, \\ &\Pr[\mathrm{AskH}_1] = \Pr[\mathrm{AskH}_2]. \end{aligned} \tag{24}$$

According to the definition of the event AskH, the accumulated probability is at least $\Pr[\mathrm{AskH}_2]/q_h$. Thus, we have

$$\Pr[\mathrm{AskH}_2] \le q_h \mathrm{Adv}^{\mathrm{CPCDH}}(t). \tag{25}$$

In Game G_5, Diffie–Hellman keys SK are random and independent of passwords and ephemeral keys. So, there are two possible cases where the adversary distinguishes the real session key from the random key as follows:

Case 1. The adversary queries the hash oracle on $ID||T_i(x)||T_j(x)||n_1||n_2||T_j(T_i(x))$. The probability that this event occurs is $(q_h/2^l)$.

Case 2. The adversary asks the *Send* (U^i, m) query and successfully impersonates a user. If the Corrupt$(U, 1)$ has been made, it implies that the Corrupt $(U, 2)$ has not been made. To impersonate the user, the adversary has to obtain the parameter c and the identity. The probability that the event happens is $\mathrm{Adv}^{\mathrm{IF}}(t')(q_s/|W|)$. On the contrary, if the Corrupt $(U,2)$ has been made, it is not allowed to reveal the static key PW of the user. Thus, in order to impersonate the user, the adversary has to obtain some information on the password of the user. The success probability of the adversary in the q_s sessions is $(q_s/(|D|+|W|))$. If the adversary just makes an attempt at random to impersonate the user by computing θ_3 and succeeds, it will make the difference; but, the probability for qs sessions is less than. $(q_s/2^l)$

Hence, we have

$$\Pr[\mathrm{Succ}_5] \le \frac{1}{2} + \frac{q_h}{2^l} + \mathrm{Adv}(t')\frac{q_s}{|W|} + \frac{q_s}{|D|+|W|} + \frac{q_s}{2^l}. \quad (26)$$

Using the triangular inequality and equations (19)–(26), we have the following:

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{TFAKA-2}}(A) &= \left|\Pr\left[\mathrm{Succ}_0 - \frac{1}{2}\right]\right| \\
&\le \sum_{i=0}^{4}\left|\Pr[\mathrm{Succ}_i] - \Pr[\mathrm{Succ}_{i+1}]\right| \\
&\quad + \left|\Pr[\mathrm{Succ}_5] - \frac{1}{2}\right| \\
&\le \frac{\left(q_s^2+q_e^2\right)}{2^{n+1}} + \frac{\left(q_s^2+q_e^2\right)}{2(p_0-1)} + \frac{q_h^2}{2^{l+1}} + \frac{3q_h}{2^{l-1}} \\
&\quad + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} + \frac{q_s}{2^{3l}} + \frac{q_s}{|D|+|W|} \\
&\quad + q_h\mathrm{Adv}^{\mathrm{CPCDH}}(t) + \mathrm{Adv}^{\mathrm{IF}}(t')\frac{q_s}{|W|}.
\end{aligned} \quad (27)$$

Thus, we have completed the proof of the theorem.

The above theorem is about the security of the proposed TAKACP-2 scheme based on the extended Chebyshev polynomials defined on the interval $[-\infty, +\infty]$. To complete the security proof of the TAKACP-1 scheme based on Chebyshev polynomials over the interval [–1, 1], one only needs to delete the CPCDH simulation in Game G_5 of the proof of Theorem 1. We only state the results as Theorem 2 without detailed proof. □

Theorem 2. *Let D(W) be a uniformly distributed password (identity) dictionary of size |D|(|W|). Let A (including A_1 and A_2) be the polynomial time-bound adversary against the semantic security of the TAKACP-1 scheme. Suppose A makes Send queries q_s times, Executes queries q_e times, and hash oracle queries q_h times at most. Then, we have*

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{TFAKA-2}}(A) \le{}& \frac{\left(q_s^2+q_e^2\right)}{2^{n+1}} + \frac{q_h^2}{2^{l+1}} + \frac{3q_h}{2^{l-1}} + \frac{q_s}{2^{l-1}} + \frac{q_s}{2^{2l}} \\
&+ \frac{q_s}{2^{3l}} + \frac{q_s}{|D|+|W|} + \mathrm{Adv}^{\mathrm{IF}}(t')\frac{q_s}{|W|},
\end{aligned} \quad (28)$$

where l refers to the string length of hash results, $Adv^{CMCDH}(t)$ is the success probability of any probabilistic polynomial-time Turing machine within the time upper bound t in solving CPCDH problems, and $Adv^{IF}(t')$ is the probability of any probabilistic polynomial-time Turing machine in solving the square root with composite number module within time upper bound t'.

Theorem 3. *The proposed TAKACP-1(TAKACP-2)scheme achieves the property of the medium unlinkability.*

Proof. Consider that the insider adversary $\boldsymbol{A}_1$ would attempt to violate the user anonymity of the proposed schemes. Further suppose that the Corrupt $(U, 2)$ has been made. The smart card of the user U is compromised. The adversary $\boldsymbol{A}$ has extracted the elements $\{d_1, d_2, n\}$ for *TAKACP-1*($\{d_1, d_2, n, p_0\}$ for *TAKACP-2*) in the smart card. Since $d_1 = E_s(ID||n) \oplus h(ID||PW)$, $d_2 = h((ID \oplus PW)||d_1||n)$ in *TAKACP-1*, or $d_1 = E_s(ID||n||p_0) \oplus h(ID||PW)$, $d_2 = h((ID \oplus PW)||d_1||(n \oplus p_0))$ in *TAKACP-2*, *A* cannot divide d_1 or d_2 into the exclusive-OR items $E_s(ID||n)$ or $E_s(ID||n||p_0)$, $h(ID||PW)$, $h(ID \oplus PW)$ correctly. In essence, even if $\boldsymbol{A}$ has $E_s(ID||n)$, $\boldsymbol{A}$ cannot still recover *ID* from it without the master key *s*. Since $d_2 = h((ID \oplus PW)||d_1||n)$ or $h((ID \oplus PW)||d_1||\ (n \oplus p_0))$, owing to the one-way hash function, $\boldsymbol{A}_1$ cannot derive *ID* or *PW* from d_2.

Now consider the authenticated key exchange phase. Assume that $M_{0,1} = \{e_0, w_{0,1}, \theta_{0,1}\}$ and $M_{1,1} = \{e_1, w_{1,1}, \theta_{1,1}\}$ are two requesting messages produced by one user in two different authentication sessions, where

$$\begin{aligned}
e_{0,1} &= \left(cn_{0,1}\right)^2 \mathrm{mod} n, \\
w_{0,1} &= h\left(n_{0,1}\right) \oplus \left(x \| T_{0i}(x)\right), \\
\theta_{0,1} &= h\left(c \| x \| T_{0i}(x) \| h\left(\mathrm{ID} \| n_{0,1}\right)\right), \\
e_{1,1} &= \left(cn_{1,1}\right)^2 \mathrm{mod} n, \\
w_{1,1} &= h\left(n_{1,1}\right) \oplus \left(y \| T_{1i}(y)\right), \\
\theta_{1,1} &= h\left(c \| y \| T_{1i}(y) \| h \| \left(\mathrm{ID} \| n_{1,1}\right)\right).
\end{aligned} \quad (29)$$

Due to the usage of the random integers $\{n_{0,1}, n_{1,1}\}$, $e_{0,1}$ is independent of $e_{1,1}$. Likewise, owing to the randomness of $\{x, y, 0_i, 1_i\}$, $T_{0i}(x)$ is independent of $T_{1i}(y)$. Thus, $w_{0,1}$ is independent of $w_{1,1}$. As *h()* is a secure cryptographic hash function, the same is true for $\theta_{0,1}$and $\theta_{1,1}$. Therefore, $\boldsymbol{A}$ believes that $M_{0,1}$ and $M_{1,1}$ are independent of each other. We can make similar analysis of the response messages $M_{0,2} = \{w_{0,2}, \theta_{0,2}\}$, $M_{1,2} = \{w_{1,2}, \theta_{1,2}\}$, and the confirmation

messages $M_{0,3} = \{w_{0,3}, \theta_{0,3}\}$, $M_{1,3} = \{w_{1,3}, \theta_{1,3}\}$. From the above analysis, we have $\Pr[A^{\text{Decide}}] = 1/2$. Thus, $\text{Adv}^{\text{Strong-Unlin}}(A) = 0$.

Therefore, our protocols achieve medium unlinkability. Any adversary (but A_0) is unable to link two different protocol sessions to the same user. □

6.2. Authentication Proof Based on BAN-Logic. In this section, we introduce the well-popular Burrows-Abadi-Needham Logic (BAN-logic) to validate the authentication of the proposed protocols. By using BAN logic, we also try to find out flaws in the proposed schemes and deal with authentication issues among the participants. The formal verification of the BAN logic demonstrates that the proposed protocols achieve mutual authentication and allow the user and the server to establish session keys. It is well-known that BAN logic [46] is the widely used logical analysis method of reasoning the beliefs of participants in an authentication protocol [47, 48]. BAN logic uses a set of postulates to analyze and verify authentication schemes. BAN logic has three elementary items, i.e., formulas/statements, principals, and keys. Let X and Y be two statements, P and Q be principals, K be the symbol for a key. The basic expressions of BAN logic are described in Table 5. More details can be found in [46–48].

The main logical postulates of the *BAN* logic are listed as follows:

Message-meaning$_K$ rule: $(P| \equiv P \overset{K}{\leftrightarrow} Q, P \triangleleft \{X\}_K / P| \equiv Q| \sim X)$, $(P| \equiv P \Leftrightarrow Q, P \triangleleft \langle X \rangle_K / P| \equiv Q| \sim X)$: If P believes that it shares K with Q and sees X encrypted by K (or X combined with K), then P believes that Q once said X.

The nonce-verification rule: $(P| \equiv \#(X), P| \equiv Q| \sim X / P| \equiv Q| \equiv X)$

If P believes that X could have been uttered only recently and Q once said X, then P believes that Q believes X.

The freshness propagation rule: $(P| \equiv \#(X) / P| \equiv \#(X, Y))$

If P believes that X is fresh, then P also believes that (X, Y) is fresh.

The jurisdiction rule: $(P| \equiv Q| \Rightarrow X, P| \equiv Q| \equiv X / P| \equiv X)$

If P believes that Q has authority over X and Q believes X, then P trusts Q on the truth of X.

The message decryption rule: $(P| \equiv P \overset{K}{\leftrightarrow} Q, P \triangleleft \{X\}_K / P \triangleleft X)$

If P believes that it shares K with Q and sees encrypted X by K, then P sees X.

In the following, we apply BAN logic to analyze the TAKACP-1 scheme. Similar analysis can be applied to the TAKACP-2 scheme. According to the analytic procedures of the BAN logic, the proposed TAKACP-1 scheme must satisfy the following goals:

Goal (1): $U| \equiv S| \equiv U \overset{\text{SK}}{\leftrightarrow} S$,

Goal (2): $U| \equiv U \overset{\text{SK}}{\leftrightarrow} S$,

Goal (3): $S| \equiv U| \equiv U \overset{\text{SK}}{\leftrightarrow} S$,

Goal (4): $S| \equiv U \overset{\text{SK}}{\leftrightarrow} S$.

The generic form of the proposed TAKACP-1 scheme is described below.

From message M_1, $U \longrightarrow S$: $\{c, n_1\}_n$, $\langle x, T_i(x) \rangle_{h(n_1)}$, $< c_0, x, T_i(x) >_c$.

From message M_2, $S \longrightarrow U$: $\langle n_2, T_j(x) \rangle_{h(\text{ID}^* \| n_1')}$, $\langle T_{j'}(x), h(n_2), \text{SK} \rangle_{c_0}$.

From message M_3, $U \longrightarrow S$: $\langle n_1 \oplus n_2, \text{SK} \rangle_{c_0}$.

The idealized form of the proposed TAKACP-1 scheme in the language of formal logic is given below.

Message M_1: $U \longrightarrow S$: $\langle x, T_i(x), U \overset{c_0}{\leftrightarrow} S, U \overset{n}{\Leftrightarrow} S \rangle_{U \overset{c}{\leftrightarrow} S}$.

Message M_2: $S \longrightarrow U$: $\langle n_2, T_j(x), U \overset{\text{SK}}{\leftrightarrow} S \rangle_{U \overset{c_0}{\leftrightarrow} S}$.

Message M_3: U $\longrightarrow$ S: $\langle n_1, n_2, U \overset{\text{SK}}{\leftrightarrow} S \rangle_{U \overset{c_0}{\leftrightarrow} S}$.

We make the assumptions about the initial state to analyze the proposed TAKACP-1 scheme:

$\boldsymbol{H_1}$: $U| \equiv \#(n_1)$, $U| \equiv \#(T_i(x))$;

$\boldsymbol{H_2}$: $S| \equiv \#(n_2)$, $S| \equiv \#(T_j(x))$;

$\boldsymbol{H_3}$: $S| \equiv U \overset{c}{\leftrightarrow} S$;

$\boldsymbol{H_4}$: $U| \equiv U \overset{c_0}{\leftrightarrow} S$;

$\boldsymbol{H_5}$: $S| \equiv U| \Rightarrow U \overset{c_0}{\leftrightarrow} S$;

$\boldsymbol{H_6}$: $U| \equiv S| \Rightarrow U \overset{\text{SK}}{\leftrightarrow} S$;

$\boldsymbol{H_7}$: $S| \equiv U| \Rightarrow U \overset{\text{SK}}{\leftrightarrow} S$.

According to the BAN logic and the assumptions, we give the main proof as follows:

From message M_1, we have

S_1: $S \triangleleft \langle x, T_i(x), U \overset{c_0}{\leftrightarrow} S, U \overset{n}{\Leftrightarrow} S \rangle_{U \overset{c}{\leftrightarrow} S}$.

From S_1, H_3, and Rule (1), we have

S_2: $S| \equiv U| \sim \langle x, T_i(x), U \overset{c_0}{\leftrightarrow} S, U \overset{n}{\Leftrightarrow} S \rangle$.

From S_2, H_2, Rule (2) and Rule (3), we have

S_3: $S| \equiv U| \equiv U \overset{c_0}{\leftrightarrow} S$.

From S_3, H_5, and Rule (4), we have

S_4: $S| \equiv U \overset{c_0}{\leftrightarrow} S$.

From message M_2, we have

S_5: $U \triangleleft \langle n_2, T_j(x), U \overset{\text{SK}}{\leftrightarrow} S \rangle_{U \overset{c_0}{\leftrightarrow} S}$.

From S_5, H_4, and Rule (1), we have

S_6: $U| \equiv S| \sim \langle n_2, T_j(x), U \overset{\text{SK}}{\leftrightarrow} S \rangle$.

From S_6, H_1, Rule (2), and Rule (3), we have

S_7: $U| \equiv S| \equiv U \overset{\text{SK}}{\leftrightarrow} S$(**Goal** (1)).

From S_7, H_6, and Rule (4), we have

S_8: $U| \equiv U \overset{\text{SK}}{\leftrightarrow} S$(**Goal** (2)).

From message M_3, we have

S_9: $S \triangleleft \langle n_1, n_2, U \overset{\text{SK}}{\leftrightarrow} S \rangle_{U \overset{c_0}{\leftrightarrow} S}$.

From S_9, S_4, and Rule (1), we have

S_{10}: $S| \equiv U| \sim \langle n_1, n_2, U \overset{\text{SK}}{\leftrightarrow} S \rangle$.

Table 5: Notations of BAN logic.

Symbol	Descript
$P\mid\equiv X$	The principal P believes a statement X, or P would be entitled to believe X.
$P\triangleleft X$	P sees X. P has received a message containing X and can read and repeat X (possibly after doing some decryption).
$P\mid\sim X$	P once said X. P at some time sent a message containing X. It is not known whether this is a replay, though it is known that P believed X when he or she sent it.
$P\mid\equiv X$	P has jurisdiction over X. P is an authority on X and is trusted on this matter.
$\#(X)$	The formula X is fresh. That is, X has never been sent in a message at any time before the current run of the protocol
$P\overset{K}{\leftrightarrow}Q$	K is a key shared between P and Q. P and Q may use K to communicate. And K is good since it can never be discovered by any principal except P or Q, or a principal trusted by either P or Q.
$P\overset{X}{\Leftrightarrow}Q$	The formula X is a shared key known only to P and Q, possibly to principals trusted by them.
$\{X\}_K$	The formula X is encrypted by K.
$<X>_Y$	This represents X combined with the formula Y. It is intended that Y be a secret and that its presence proves the identity of whoever utters $<X>_Y$. X is simply concatenated with Y while Y plays a role as proof or origin for X.

From S_{10}, H_2, Rule (2), and Rule (3), we have

S_{11}: $S\mid\equiv U\mid\equiv U\overset{\mathrm{SK}}{\leftrightarrow}S$(**Goal** (3)).

From S_{11}, H_7, and Rule (4), we have

S_{12}: $S\mid\equiv U\overset{\mathrm{SK}}{\leftrightarrow}S$ (**Goal** (4)).

6.3. Informal Security Analysis. In this section, we analyze the security of the proposed protocols. We will show that the proposed protocols satisfy the essential security requirements, including the ability to provide medium unlinkability, the contributory property of key agreements, session key security, two-factor secrecy, and free updating of passwords. Furthermore, we confirm that the proposed schemes can withstand replay attacks and password-guessing attacks. In the following, we will not expound the two-factor security since its proof has been given in Part A and Part B of Section 4, respectively.

6.3.1. Medium Unlinkability. In Part A of Section 4, we have demonstrated that our protocols can provide medium unlinkability. Now, we compare the privacy-preserving of our protocols with that of the schemes of Guo-Chang [19], Lin [20], and Sun et al. [24].

In the Guo-Chang scheme, since the user identity is encrypted with the master key of the server into the login request R, any adversary A cannot reveal the user identity. So $\Pr[A^{Guess}] = 1/N$. However, R is unchanged until the user updates the password. Thus, any outside adversary can distinguish whether the users in two authentication sessions are identical. That is, $\Pr[A_2^{\mathrm{Decide}}] = 1$. Thus, we have $\mathrm{Adv}^{\mathrm{Anon-Alone}}(A) = 0$ and $\mathrm{Adv}^{\mathrm{Weak-Unlin}}(A) \geq 1/2$. Hence, the Guo-Chang scheme provides *anonymity-alone* but *weak unlinkability*. Similarly, since every request of the user contains the unchanged element IM, Sun et al.'s scheme [24] only achieves the property *anonymity-alone*.

The Lin scheme can provide weak unlinkability. Specifically, although the elements Q and R are kept unchanged, they are transmitted in the ciphertext $E_v(Q, R, T_1)$. Since any outside adversary $\boldsymbol{A_2}$ cannot calculate the key v, they cannot obtain R. But any inside adversary $\boldsymbol{A_1}$ can compute v and acquire Q, R. For every login of the user, the parameters $\{Q, R\}$ are static until the user changes its password or identity. Thus, $\boldsymbol{A_1}$ can still decide whether different login requests are from the same user or not. Then, we have $\Pr[A_{1,2}^{\mathrm{Decide}}] = 1$. Hence, $Adv^{Medium\text{-}Unlin}(A) > 1/2$. The Lin scheme cannot provide *medium unlinkability*.

In the proposed protocols, i and x are chosen randomly by every individual user. Therefore, no unchanged login message can be derived by the other registered users. Thus, we obtain that $Adv^{Medium\text{-}Unlin}(A) = 0$. However, each time the user logins to the server, the server validates his or her identity. So, the advantage $Adv^{Strong\text{-}Unlin}(A)$ is non-negligible. The proposed schemes cannot achieve *the medium unlinkability*.

6.3.2. Contributory Property of Key Agreement. In the proposed protocols, the session key SK is $h(ID||T_i(x)||T_j(x)||n_1||n_2||T_i(T_j(x)))$. Only for the *TAKACP*-1 scheme, the server can use the method mentioned in [18, 44] to compute i^* and j^*, thus satisfying $T_i(x) = T_{i^*}(x)$, $T_j(x) = T_{j^*}(x)$, where $T_{i^*}(T_{j^*}(x))$ represents a previous parameter. Since x, $T_i(x)$and n_1 are randomly selected by the user and SK contains $T_i(x)$ and n_1, the server still fails to predetermine a session key. Likewise, since $T_j(x)$ and n_2 are randomly selected by the server and SK contains $T_j(x)$ and n_2, the user cannot predetermine a session key. Notably, neither the server nor the user can determine the specific session key alone in advance. Therefore, the proposed protocols satisfy the contributory property of key agreements.

6.3.3. Session Key Security. Firstly, since i, j, n_1 and n_2 are selected randomly in every run of the protocols, the session key $SK = h(ID||T_i(x)||\ T_j(x)||n_1||n_2||T_i(T_j(x)))$ is independent of the previously generated session keys. Thus, the proposed protocols can resist against known-key attacks.

Secondly, we demonstrate that the proposed protocols can prevent any inside adversary from computing the session keys. Consider that an inside adversary $\boldsymbol{A}_1$ has eavesdropped the communication messages $\{e, w_1, \theta_1, w_2, \theta_2, \theta_3\}$ between the user and the server. Since $\boldsymbol{A}_1$ is a legal user of the same server, $\boldsymbol{A}_1$ knows n. However, n_1 cannot be derived from e without the server's private keys p and q, where $e = (cn_1)^2 \bmod n$, owing to the quadratic residue assumption. Thus, the adversary cannot still recover $x||T_i(x)$ from w_1 without the knowledge of n_1, where $w_1 = h(n_1)\oplus(x\|T_i(x))$. Moreover, $\boldsymbol{A}$ cannot work out $h(ID||n_1)$. $\boldsymbol{A}_1$ cannot obtain

$n_2||T_j(x)$ from w_2, since $w_2 = h(\mathrm{ID}^*\|n_1')\oplus(n_2\|T_j(x))$. Since $\theta_1 = h(c\|x\|T_i(x)\|c_0)$, $\theta_2 = h(c_0^*\|(T_j(x)\oplus h(n_2))\|\mathrm{SK})$, and $\theta_3 = h(\mathrm{SK}^*\|(n_1\oplus n_2)\|c_0)$, due to the one-way property of the hash function, $\boldsymbol{A}_1$ cannot determine $T_i(x)$, $T_j(x)$, or SK. In a word, the session key cannot be derived from the messages transmitted over the public channel. The proposed schemes achieve session key security.

6.3.4. Free Updating of Password. As described in Part C of Section 5, a user can freely update password without any interaction with the server during the password change phase.

6.3.5. Resistance to Password Guessing Attacks. In the proposed TAKACP protocols, the login request message e is information that involves password PW. An inside adversary $\boldsymbol{A}_1$ may guess the password through the equation $c = d_1\oplus h(ID||PW)$. However, an inside adversary $\boldsymbol{A}_1$ cannot still obtain c from e since $e = (cn_1)^2 \bmod n$. Therefore, the proposed protocols can resist password guessing attacks.

6.3.6. Resistance to Replay Attacks. The proposed schemes maintain freshness by using two nonces and two chaotic maps. Specifically, the proposed protocols guarantee the freshness of messages by using $T_i(x)$ and n_1 in Step A1, $T_j(x)$ and n_2 in Step A2, and $\{T_i(x), T_j(x), n_1, n_2\}$ in Steps A3 and A4, respectively. Since n_1 is protected by the quadratic residues, only the server and the user itself know it. $T_i(x)$, $T_j(x)$, and n_2 are contained in w_1 and w_2. They can be calculated only when one knows the nonce n_1. Therefore, the proposed schemes can prevent replaying attacks.

7. Security, Functionality, and Performance Comparison

In this section, we will make a comparison with the related TAKACP protocols in terms of security, functionality, and performance.

7.1. Security Comparison. We compare the security of our proposed TAKACP schemes with respect to the related authenticated key agreement schemes [19–25, 27–30]. Table 6 summarizes the security properties of the proposed schemes and illustrates the comparison result.

As is indicated in Table 6, the proposed schemes are highly secure as compared to the related authenticated key agreement schemes [19–25, 27–30]. Especially, the proposed schemes can deal with several imperative security issues which most of the authenticated key agreement protocols based on the Chebyshev polynomials defined on the interval [−1,+1] suffer from. For example, the proposed schemes eliminate their weaknesses of the Lin scheme and the Guo-Chang scheme. In contrast, the authentication protocols [22, 23, 25, 28, 29] cannot preserve user anonymity. The protocol in [27] cannot provide the contributory property of key agreement since the session key is determined by the user. The authentication protocols [19, 20, 23] even cannot provide the session key security. Those protocols presented in [19–21, 23] cannot provide free updating of passwords. The Guo-Chang scheme achieves the anonymity-alone, while the Lin scheme provides weak unlinkability. The proposed schemes achieve the property, medium unlinkability. Note that designing the two-factor authentication protocol with strong unlinkability is still challenging.

7.2. Performance Comparison. In this section, we evaluate the performance of the proposed schemes and make a comparison with the related authenticated key agreement schemes [19–25] in terms of the communication cost, storage, and computational overhead.

We suppose that the block size of secure symmetric cryptosystems is 128 bits, and the output size of a secure one-way hash function is 256 bits. In order to make the factoring problems infeasible in practical implementation, let the module n be an integer of 1024 bits. Since the registration of our schemes is based on a one-way hash function, the password length can be 128 bits. Suppose that the size of ID is 64 bits. In our proposed scheme, the cryptographic parameters $\{c, d\}$ must be stored in the smart card. The length of this information is $1152 + 128 = 1280$ bits, where d can be 128 bits and c must be encrypted in nine blocks. The size of the information stored in the smart card is $64 + 64 + 128 \times 7 + 1024 = 2048$ bits in Fan et al.'s scheme [22], $128 \times 3 + 128 + 64 + 64 + 256 = 896$ bits in Juang et al.'s scheme [23], $128 + 128 \times 3 = 512$ bits in Sun et al.'s scheme [24], $128 \times 3 + 128 + 64 + 64 = 640$ bits in Li et al.'s scheme [25], $128 \times 3 + 256 + 256 = 896$ bits in Guo et al.'s scheme [19], $128 \times 3 + 256 + 128 + 128 = 896$ bits in Lin's scheme [20], and $128 \times 2 + 128 + 128 = 512$ bits in Lee's scheme [21]. As is shown in Table 7, during the registration, the smart card needs a little larger storage space in the proposed schemes than those in other schemes [19–25, 28, 29]. However, it is practically insignificant considering the fact that most current mobile devices, including 4G cellular phones, personal digital assistants (PDAs), and notebook computers, have over a few hundred MB or a few GB of available memory.

In our proposed schemes, the messages transmitted in the registration phase are $\{ID, d\}$. The communication cost of the login protocol is $64 + 256 = 320$ bits. The total size of messages transmitted during the authenticated key exchange phase for cryptographic parameters $\{e, w_1, \theta_1\}$, $\{w_2, \theta_2\}$, and $\{\theta_3\}$ is $(1024 + 256 + 256) + (256 + 256) + 256 = 2294$ bits. The authenticated key exchange phase requires three rounds of message transmission. During the password change, the proposed schemes require no message transmission between the user and the server since the server is not involved with the phase. Let the module number of the elliptic curve be an integer of 163 bits in Juang et al.'s scheme [23] and Sun et al.'s scheme [24]. Let the time stamp be a string of 32 bits in Guo et al.'s scheme [19], Lin's scheme [20], and Lee's scheme [21]. In Juang et al.'s scheme, the communication cost of the login phase about cryptographic parameters $\{b_i, E_{V_i}(e)\}$, $\{u, M_s\}$, and $\{M_U\}$ is $(384 + 384) + (256 + 256) + 256 = 1536$ bits, where $E_{V_i}(e)$ is the encryption of three

Table 6: Security comparison.

	*S*1	*S*2	*S*3	*S*4	*S*5	*S*6	*S*7	*S*8	*S*9	*S*10
Fan et al. [22]	Yes	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Juang et al. [23]	No	No	Yes	Yes	No	Yes	No	Yes	Yes	No
Sun et al. [24]	No	Anonymity-alone	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Li et al. [25]	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Maitra [27]	No	Medium unlinkability	Yes	No	Yes	Yes	Yes	Yes	Yes	No
Chaudhry et al. [28]	Yes	No	Yes	Yes	Yes	No	#	Yes	Yes	No
Guo et al. [19]	Yes	Anonymity-alone	Yes	Yes	No	Yes	No	Yes	Yes	No
Lin [20]	Yes	Weak unlinkability	Yes	Yes	No	Yes	No	Yes	Yes	No
Lee [21]	Yes	Medium unlinkability	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Irshad et al. [29]	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Irshad et al. [30]	No	Medium unlinkability	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Our protocols	Yes	Medium unlinkability	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

*S*1: No password table and verification table; *S*2: User privacy-preserving; *S*3: Mutual authentication; *S*4: Contributory property of key agreement; *S*5: Session key security; *S*6: Two-factor secrecy; *S*7: Free updating of password; *S*8: Resistance to password guessing attacks; *S*9: Resistance to replay attack. *S*10: Key confirmation.

Table 7: Storage cost and communication cost comparison.

	SC_1 (in bits)	$SC_2{}^*$	SC_2 (in bits)	$SC_3{}^*$	SC_3 (in bits)	$SC_4{}^*$	SC_4 (in bits)
Fan et al. [22]	2048	3	512	3	1856	#	#
Juang et al. [23]	896	3	578	3	1536	5	2560
Sun et al. [24]	512	2	192	3	1548	0	0
Li et al. [25]	640	1	576	3	3200	5	5248
Maitra [27]	1600	1	1792	2	4416	0	0
Chaudhry et al. [28]	512	1	512	3	2624	#	#
Guo et al. [19]	896	1	320	2	1792	2	1408
Lin [20]	896	1	320	2	1408	2	1280
Lee [21]	512	1	320	2	1088	2	1440
Irshad et al. [29]	768	1	1344	2	1088	0	0
Irshad et al. [30]	1024	1	1344	5	5184	0	0
Our protocols	1280	1	320	3	2294	0	0

SC_1 denotes the storage cost of the smart card in the registration phase. $SC_2{}^*$ denotes the round number of message transmission in the registration phase. SC_2 denotes the total size of the transmitted message in the registration phase. $SC_3{}^*$ denotes the number of message transmissions in the authenticated key exchange phase. SC_3 denotes the total size of the transmitted message in the authenticated key exchange phase. $SC_4{}^*$ denotes the number of the message transmission in the password change phase. SC_4 denotes the total size of transmitted message in the password change phase. # denotes that the scheme does not provide the functionality of password updating.

blocks. The password change phase of Juang et al.'s scheme [23] requires that the user needs to agree on a session key with the server through the log-in phase in advance and then transmit the messages $E_{S_k}(\mathrm{ID}_i, h(\mathrm{PW}_i^* \| b^*))$and $E_{S_k}(b_i^*)$, respectively. Hence, the size of the message transmitted in the password change phase of Juang et al.'s scheme is 2560 bits. By similar analysis, we can evaluate the communication cost of other related schemes [19–22, 24, 25, 27–30]. The communication cost and storage cost among our schemes and related schemes are shown in Table 7.

As can be seen from Table 7, during the authenticated key exchange phase, the size of the message transmitted between the user and the server in the proposed schemes is a little larger than the size of the message in the schemes [19–24,29]. However, the proposed schemes can provide the user with stronger privacy protection than the schemes in [22–24]. It also can achieve the session key confirmation, but the schemes [19–21,29] cannot provide the function. Moreover, the scheme in [29] cannot preserve the anonymity of the user. During the password change phase in the proposed schemes, no message transmission between the user and the serve is required. Compared with the proposed scheme, the server of the other schemes [19–23,25] is involved with the password change. Furthermore, quite a few messsages will be transmitted between the user and the server.

Now, we evaluate the computation cost of our protocols and related protocols. Let T_c denote the time to execute a Chebyshev polynomial computing. Let T_s represent the time to execute a symmetric encryption/decryption operation. We refer to T_h as the time to execute a one-way hash function operation. Let T_m denote the time to execute a scalar multiplication in the elliptic curve group. T_e represents the time to execute one exponentiation operation. We denote by T_{sq} the time to execute a squaring operation. T_{crt} represents the time to solve the square root through the CRT method. Since the XOR operations cost very little, we neglect it. Since a user is required to register with a server one time, the computational cost in the registration phase is not listed in Table 8.

The proposed protocols protect the random number n_1 and the shared secret c by using the quadratic residues. The user requires one modular squaring operation, and the

Table 8: Computation overhead comparison.

	C_1	C_2	C_3	C_4
Fan et al. [22]	$4T_h + 1T_{sq} \approx 0.0175\ ms$	$3T_h + 1T_s + 1T_{\text{crt}} \approx 0.0238\ ms$	#	#
Juang et al. [23]	$3T_h + 3T_s \approx 0.021\ ms$	$4T_h + 6T_s + 1T_m \approx 0.186\ ms$	$14T_h + 5T_s \approx 0.1015\ ms$	$52T_h + 5T_s \approx 0.2345\ ms$
Sun et al. [24]	$4T_h + 2T_m \approx 0.232\ ms$	$4T_h + 1T_s + 1T_m \approx 0.1335\ ms$	$2T_h \approx 0.007\ ms$	0
Li et al. [25]	$8T_h + 4T_s \approx 0.0385\ ms$	$10T_h + 10T_s + 1T_m \approx 0.249\ ms$	$21T_h + 6T_s \approx 0.1365\ ms$	$70T_h + 9T_s \approx 0.3395\ ms$
Maitra [27]	$8T_h + 6\text{t}_e + 1t_m \approx 157.069\ ms$	$8T_h + 4t_e + 1t_m + 1t_{\text{inv}} \approx 150.029\ ms$	$6T_h \approx 0.042\ ms$	0
Chaudhry et al. [28]	$1T_p + 6T_h + 3T_m + 2T_a + 2T_e + 2t_m + 1t_{inv} \approx 183.7542\ ms$	$3T_p + 5T_h + 4T_m + 3T_a + 2T_e + 1t_m \approx 462.9762\ ms$	#	#
Guo et al. [19]	$2T_h + 2T_s + 3T_c \approx 0.3925\ ms$	$2T_h + 2T_s + 3T_c \approx 0.3925\ ms$	$2T_h + 1T_s + 2T_c \approx 0.2542\ ms$	$3T_s + 1T_c \approx 0.153\ ms$
Lin [20]	$3T_h + 2T_s + 2T_c \approx 0.2745\ ms$	$2T_h + 3T_s + 3T_c \approx 0.403\ ms$	$2T_h + 1T_s + 1T_c \approx 0.1327\ ms$	$3T_s + 1T_c \approx 0.153\ ms$
Lee [21]	$3T_h + 1T_s + 3T_c \approx 0.3945\ ms$	$2T_h + 2T_s + 3T_c \approx 0.3925\ ms$	$4T_h + 1T_s + 2T_c \approx 0.2675\ ms$	$2T_h + 3T_s + 1T_c \approx 0.16\ ms$
Irshad et al. [29]	$7T_h + 4T_c \approx 0.5105\ ms$	$4T_h + 3T_c \approx 0.3785\ ms$	$6T_h \approx 0.042\ ms$	0
Irshad et al. [30]	$11T_h + 3T_c \approx 0.4030\ ms$	$7T_h + 2T_c \approx 0.2675\ ms$	$8T_h \approx 0.056\ ms$	0
Our protocols	$9T_h + 1T_{sq} + 1T_c \approx 0.1565\ ms$	$7T_h + 1T_{crt} + 1T_s + 1T_c \approx 0.1593\ ms$	$2T_h \approx 0.014\ ms$	0

C_1 denotes the computational cost of a user in the authenticated key exchange phase. C_2 refers to the computational cost of the server in the authenticated key exchange phase. C_3/C_4 represents the computational cost of the user/server during the password change phase. N/A represents no requirement of computation. # denotes that the scheme does not provide the functionality of password updating.

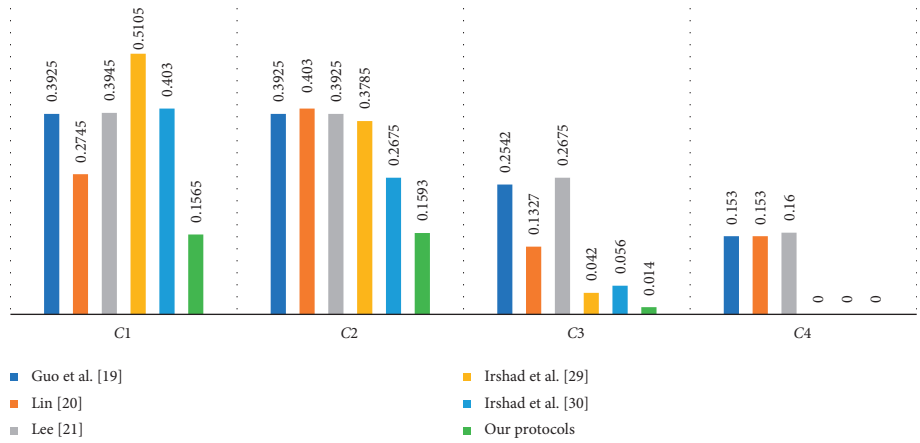

Figure 6: Computation overhead comparison of TAKACP protocols (in milliseconds).

server requires one square root solving operation through the CRT and one symmetric decryption. The proposed protocols require no symmetric encryption operation. It needs only one Chebyshev polynomial computing in the user, which is less than one (two) Chebyshev polynomial computing than those in the Lin scheme (the Guo-Chang scheme). The proposed protocol requires one symmetric encryption operation and one Chebyshev polynomial computing in the server, which is less than two (one) symmetric encryption operations and two (two) Chebyshev polynomial computing than those in the Lin scheme (the Guo-Chang scheme). In the proposed protocols, one modular squaring operation does not affect user efficiency since the implementation of one modular squaring [35] can be reduced to only a few hundred gate-equivalents. In practical implementation of the proposed protocols, to efficiently compute the square roots in Z_n^*, the server does the pre-computation [22]. To be specific, S pre-computes and stores the inverse p' of p modular q and the inverse q' of q modular p. In order to compute the square root of a, one first computes $a_1 = a^{(p+1)/4} \bmod p$ and $a_2 = a^{(q+1)/4}$, then he or she can calculate rapidly the four square roots of a in Z_n^*, for example, $(p'pa_2 + q'qa_1) \bmod n$. Due to $p \equiv q \equiv 3 \pmod 4$, the computation of (a_1, a_2) requires about the same time as performing a modular exponentiation computation in Z_n^*. Consequently, we have $1T_{\text{crt}} \approx 1T_e$.

We have executed these operations by utilizing PyCrypto library in Python language in the computer with 16 GB RAM and a clock speed of 3.60 GHz. The time cost of all operations is as follows: $T_h \approx 0.0035$ ms, $T_c \approx 0.1215$ ms, $T_s \approx 0.0105$ ms, $T_m \approx 0.109$ ms, $T_{\text{sq}} \approx 0.0035$ ms, and $T_{\text{crt}} \approx 0.0028$ ms. Table 8 summarizes the computation cost of our scheme with those described in [19–25, 27–30]. As shown in Table 8 and Figure 6, during the authenticated key exchange phase, both the user and the server in the proposed protocols are required at the lowest computation cost among these two-factor authentication protocols [19–21, 29, 30] based on chaotic maps. In addition, the proposed schemes require no involvement of the server during the password change phase. Moreover, in comparison with the related schemes [19–25], the user is required at a very low computation cost during the password change phase of the proposed schemes.

8. Conclusion

In this paper, we examine the limitations of Lin's chaotic map-based authenticated key agreement protocol. We have proposed two TAKACP protocols with key confirmation. Compared with the Lin protocol and the Guo-Chang protocol, the proposed protocols achieve the following additional merits: session key secrecy, medium unlinkability, and free updating of passwords. The proposed protocols with the enhanced security do not affect the user's or the server's efficiency. Therefore, the proposed protocols are highly feasible for practical implementation.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant nos. 61862028 and 61702238), the Natural Science Foundation of Jiangxi Province (Grant no. 20181BAB202016), and the Science and Technology Project of Provincial Education Department of Jiangxi (Grant nos. GJJ160430 and GJJ180288).

References

[1] Z. Tan, "Secure delegation-based authentication for telecare medicine information systems," *IEEE Access*, vol. 6, no. 1, pp. 26091–26110, 2018.

[2] W. Ding and W. Ping, "Two birds with one stone: two-factor authentication with security beyond conventional bound," *IEEE Transactions on Dependable and Secure*, vol. 15, no. 4, pp. 708–722, 2018.

[3] M. Gupta and N. S. Chaudhari, "Anonymous two factor authentication protocol for roaming service in global mobility network with security beyond traditional limit," *Ad Hoc Networks*, vol. 84, no. 1, pp. 56–67, 2019.

[4] Y. Cao, Q. Zhang, F. Li, S. Yang, and Y. Wang, "PPGPass: nonintrusive and secure mobile two-factor Authentication via wearables," in *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1917–1926, Toronto, Canada, April 2020.

[5] M. Fotouhi, M. Bayat, A. K. Das, H. A. N. Far, S. Morteza Pournaghi, and M.A. Doostari, "A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care IoT," *Computer Network*, vol. 177, Article ID 107333, 2020.

[6] M. F. Ayub, S. Shamshad, K. Mahmood, S. H. Islam, R. M. Parizi, and K.-K. R. Choo, "A provably secure two-factor Authentication scheme for USB storage devices," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 4, pp. 396–405, 2020.

[7] P. Gope and B. Sikdar, "Lightweight and privacy-preserving two-factor authentication scheme for IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580–589, 2019.

[8] Y. J. Niu and X. Y. Wang, "An anonymous key agreement protocol based on chaotic maps," *Communication Nonlinear Science Numerical Simulators*, vol. 16, pp. 1986–1992, 2011.

[9] E.-J. Yoon, "Efficiency and security problems of anonymous key agreement protocol based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 7, pp. 2735–2740, 2012.

[10] 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, 2016.

[11] X. Li, J. Niu, S. Kumari et al., "A novel chaotic maps-based user authentication and key agreement protocol for multi-server environments with provable security," *Wireless Personal Communications*, vol. 89, no. 2, pp. 569–597, 2016.

[12] S. Roy, S. Chatterjee, A. K. Das, S. Chattopadhyay, S. Kumari, and M. Jo, "Chaotic map-based anonymous user authentication scheme with user biometrics and fuzzy extractor for crowdsourcing internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2884–2895, 2018.

[13] Z. W. Tan, "A privacy-preserving multi-server authenticated key-agreement scheme based on Chebyshev chaotic maps," *Security Communication Networks.*vol. 9, pp. 1384–1397, 2016.

[14] V. Sureshkumar, R. Amin, M. S. Obaidat, and I. Karthikeyan, "An enhanced mutual authentication and key establishment protocol for TMIS using chaotic map," *Journal of Information Security and Applications*, vol. 53, Article ID 102539, 2020.

[15] S. Kumari, X. Li, F. Wu, A. K. Das, H. Arshad, and M. K. Khan, "A user friendly mutual authentication and key agreement scheme for wireless sensor networks using chaotic maps," *Future Generation Computer Systems*, vol. 63, pp. 56–75, 2016.

[16] J. Srinivas, A. K. Das, M. Wazid, and N. Kumar, "Anonymous lightweight chaotic map-based authenticated key agreement protocol for industrial internet of things," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 6, pp. 1133–1146, 2020.

[17] S. Chatterjee, S. Roy, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 824–839, 2018.

[18] K. Y. Cheong and T. Koshiba, "More on security of public-key cryptosystems based on chebyshev polynomials," *IEEE Transactions on Circuits & Systems II Express Briefs*, vol. 54, no. 9, pp. 795–799, 2007.

[19] C. Guo and C.-C. Chang, "Chaotic maps-based password-authenticated key agreement using smart cards," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 6, pp. 1433–1440, 2013.

[20] H.-Y. Lin, "Improved chaotic maps-based password-authenticated key agreement using smart cards," *Communications in Nonlinear Science and Numerical Simulation*, vol. 20, no. 2, pp. 482–488, 2015.

[21] T.-F. Lee, C.-H. Hsiao, S.-H. Hwang, and T.-H. Lin, "Enhanced smartcard-based password-authenticated key agreement using extended chaotic maps," *PLoS One*, vol. 12, no. 7, Article ID e0181744, 2017.

[22] C.-I. Fan, Y.-C. Chan, and Z.-K. Zhang, "Robust remote authentication scheme with smart cards," *Computers & Security*, vol. 24, no. 8, pp. 619–628, 2005.

[23] W.-S. Juang, S.-T. Chen, and H.-T. Liaw, "Robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 6, pp. 2551–2556, 2008.

[24] D. Z. Sun, J. P. Huai, J. Z. Sun, J. X. Li, J. W. Zhang, and Z. Y. Feng, "Improvements of Juang 's password-authenticated key agreement scheme using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 2284–2291, 2009.

[25] X. X. Xiangxue Li, W. D. Weidong Qiu, D. Dong Zheng, K. F. Kefei Chen, and J. H. Jianhua Li, "Anonymity enhancement on robust and efficient password-authenticated key agreement using smart cards," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, pp. 793–800, 2010.

[26] Q. Xie, D. S. Wong, G. Wang, X. Tan, K. Chen, and L. Fang, "Provably secure dynamic ID-based anonymous two-factor authenticated key exchange protocol with extended security model," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1382–1392, 2017.

[27] T. Maitra, M. S. Obaidat, R. Amin, S. H. Islam, S. A. Chaudhry, and D. Giri, "A robust ElGamal based password authentication protocol using smart card for client-server communication," *International Journal of Communication Systems*, vol. 30, no. 11, Article ID e3242, 2017.

[28] S. A. Chaudhry, I. L. Kim, S. Rho, M. S. Farash, and T. Shon, "An improved anonymous authentication scheme for distributed mobile cloud computing services," *Cluster Computing*, vol. 22, no. S1, pp. 1595–1609, 2019.

[29] A. Irshad, M. Sher, S. A. Chaudhary, H. Naqvi, and M. S. Farash, "An efficient and anonymous multi-server authenticated key agreement based on chaotic map without engaging Registration Centre," *The Journal of Supercomputing*, vol. 72, no. 4, pp. 1623–1644, 2016.

[30] A. Irshad, S. A. Chaudhry, Q. Xie et al., "An enhanced and provably secure chaotic map-based authenticated key agreement in multi-server architecture," *Arabian Journal for Science and Engineering*, vol. 43, no. 2, pp. 811–828, 2018.

[31] 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.

[32] M. S. Baptista, "Cryptography with chaos," *Physics Letter A*, vol. 240, no. 1–2, pp. 50–54, 1998.

[33] L. Kocarev, "Chaos-based cryptography: a brief overview," *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6–21, 2001.

[34] X. Guo and J. Zhang, "Secure group key agreement protocol based on chaotic hash," *Information Sciences*, vol. 180, no. 20, pp. 4069–4074, 2010.
[35] T.-F. Lee, "Enhancing the security of password authenticated key agreement protocols based on chaotic maps," *Information Sciences*, vol. 290, pp. 63–71, 2015.
[36] K. Xue and P. Hong, "Security improvement on an anonymous key agreement protocol based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 7, pp. 2969–2977, 2012.
[37] Z. Tan, "A chaotic maps-based authenticated key agreement protocol with strong anonymity," *Nonlinear Dynamics*, vol. 72, no. 1-2, pp. 311–320, 2013.
[38] P. Gong, P. Li, and W. Shi, "A secure chaotic maps-based key agreement protocol without using smart cards," *Nonlinear Dynamics*, vol. 70, no. 4, pp. 2401–2406, 2012.
[39] X.-Y. Wang and D.-P. Luan, "A secure key agreement protocol based on chaotic maps," *Chinese Physics B*, vol. 22, no. 11, Article ID 110503, 2013.
[40] H.-Y. Lin, "Chaotic map based mobile dynamic ID authenticated key agreement scheme," *Wireless Personal Communications*, vol. 78, no. 2, pp. 1487–1494, 2014.
[41] S. H. Islam, M. S. Obaidat, and R. Amin, "An anonymous and provably secure authentication scheme for mobile user," *International Journal of Communication Systems*, vol. 29, no. 9, pp. 1529–1544, 2016.
[42] S. H. Islam, "Provably secure dynamic identity-based three-factor password authentication scheme using extended chaotic maps," *Nonlinear Dynamics*, vol. 78, no. 3, pp. 2261–2276, 2014.
[43] X. Hao, J. Wang, Q. Yang, X. Yan, and P. Li, "A chaotic map-based authentication scheme for telecare medicine information systems," *Journal of Medical Systems*, vol. 37, no. 2, pp. 9919–9927, 2013.
[44] P. Bergamo, P. D'Arco, A. De Santis, and L. Kocarev, "Security of public-key cryptosystems based on Chebyshev polynomials," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 7, pp. 1382–1393, 2005.
[45] L. Zhang, "Cryptanalysis of the public key encryption based on multiple chaotic systems," *Chaos, Solitons & Fractals*, vol. 37, no. 3, pp. 669–674, 2008.
[46] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, 1990.
[47] P. Syverson and I. Cervesato, "The logic of authentication protocols," in *FOSAD 2000, LNCS 2171*Springer, Berlin, Heidelberg, 2001.
[48] K. Bicakci and N. Baykal, "One-time passwords: security analysis using BAN logic and integrating with smartcard authentication," *Computer and Information Sciences - ISCIS 2003*, vol. 2869, pp. 794–801, 2003.
[49] M. Ballare and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security (CCS'93)*, pp. 62–73, Fairfox, VA, USA, March 1993.
[50] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," *Advances in Cryptology - EUROCRYPT 2000*, vol. 18, pp. 139–155, 2000.
[51] V. Shoup, "Sequences of games: a tool for taming complexity in security proofs," 2005, http://www.shoup.net.
[52] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.