SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2015; **8**:2981–2991
Published online 10 February 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1225

RESEARCH ARTICLE

Cryptanalysis and provable improvement of a chaotic maps-based mobile dynamic ID authenticated key agreement scheme

Hongfeng Zhu*

Software College, Shenyang Normal University, No. 253, Huanghe Bei Street, Huanggu District, Shenyang, P.C 110034, China

ABSTRACT

In recent years, due to the wide applications of client–server architecture, the problem of only legal users have access to use the various remote services has attracted much attention. Consequently, many chaotic maps-based authenticated key agreement schemes using static ID have been widely used. However, static ID authentication schemes cannot provide user anonymity. It is a better choice to utilize dynamic ID authentication scheme. Recently, Lin proposed a chaotic maps-based mobile dynamic ID authenticated key agreement scheme and proved that it was secure against existential active attacks. Unfortunately, in this paper, we show that Lin's scheme cannot resist dictionary attack, user spoofing attack, and denial of service attack. Moreover, the paper first proposed an attack method called exclusive-or operation with pad operation leaking attack, which can lead to the worst case scenario: an adversary can get the session key without being detected. In addition, in the password-change phase of Lin's scheme, there is no authenticated process for the user. In other words, even if anyone else inputs the two uncorrelated passwords, the mobile device will continue to update the password, which leads to the consequence that the legal user cannot log in forever. Finally, we proposed an improved protocol based on chaotic maps with provable security under the random oracle model. Compared with previous related works, the improved protocol not only can withstand existential active attacks, but also has better computational efficiency.

KEYWORDS

dynamic ID; user spoofing attack; dictionary attack; chaotic maps

***Correspondence**

Hongfeng Zhu, Software College, Shenyang Normal University, No. 253, Huanghe Bei Street, Huanggu District, Shenyang, P.C 110034-China

E-mail: zhuhongfeng1978@163.com

1. INTRODUCTION

Nowadays, many chaotic maps-based authenticated key agreement schemes using static ID [1–3] have been widely used. However, static ID authentication schemes cannot provide user anonymity. Many authentication schemes using dynamic ID [4–8] have been proposed. In 2004, Das *et al.* [5] proposed a dynamic ID authentication scheme. However, the scheme is insecure for its independent of the password and cannot provide mutual authentication and resist impersonate remote server attack. In 2009, Wang *et al.* [6] proposed an enhanced password authentication scheme using smart cards. The proposed scheme keeps the merits of [5], and it is more secure and practical. However, the scheme of Wang *et al.* [6] cannot provide anonymity of a user during authentication and resist insider attack. To address these security flaws, in 2011, Khan *et al.* [7] proposed an enhanced authentication scheme. In 2012, Tang *et al.* [8] proposed a dynamic ID-based remote user authentication with key agreement scheme. In 2014, Lin [4] proposed a chaotic maps-based mobile dynamic ID-authenticated key agreement scheme and proved that it is secure against existential active attacks. Unfortunately, in this paper, we point out Lin's scheme cannot resist dictionary attack, user spoofing attack, and denial of service (DoS) attack, and cannot provide secure password update protocol.

In this paper, after cryptanalysis Lin's scheme, we give an improved password-authenticated key agreement scheme, which is based on chaotic maps for mobile device environment. The improved protocol is based on chaotic maps without using modular exponentiation and scalar multiplication on an elliptic curve. At the same time, our improved protocol can resist all common attacks, such as impersonation attacks and man-in-the-middle attacks.

The organization of the paper is described as follows. Section 2 reviews Lin's scheme. Section 3 shows the security flaws of Lin's scheme. The improved protocol is described in Section 4. The security and efficiency analysis is given in Section 5. This paper is finally concluded in Section 6.

2. REVIEW OF LIN'S SCHEME

Lin's scheme [4] consists of four phases: registration phase, login phase, verification phase, and password-change phase. We review Lin's scheme in the left part of Figure 1. The symbols involved are defined in Table I.

2.1. Registration phase

Step 1 U_a chooses his password PW_a and a random integer number t to compute $W_a = PW_a \oplus t$, and then sends (ID_a, W_a) to B via a secure channel.

Step 2 After receiving (ID_a, W_a), B computes $H_a = h(s, ID_a)$, $n_a = h(W_a, ID_a) \oplus (H_a \| x \| T_s(x))$, and sends n_a to U_a via a secure channel.

Step 3 After receiving n_a, U_a computes $N_a = h(ID_a, PW_a) \oplus n_a \oplus h(W_a, ID_a)$, and stores N_a in MD.

2.2. Login phase

Step 4 U_a enters (ID_a, PW_a), and then MD chooses a random integer k to compute $H_a \| x \| T_s(x) = N_a \oplus h(ID_a, PW_a)$, $Z = T_k(T_s(x))$, $CID_a = ID_a \oplus (H_a \| T_1 \| Z)$, $C = T_k(x)$, $R = H_a \oplus Z$, $V_a = h(CID_a, C, H_a, R, T_1)$. The login request (CID_a, C, V_a, R, T_1) is sent to B.

2.3. Verification phase

Step 5 After receiving (CID_a, C, V_a, R, T_1), B checks if $T_2 - T_1 \le \Delta T$. If it holds, B computes $Z = T_s(C)$, $H_a = R \oplus Z$, $ID_a = CID_a \oplus (H_a \| T_1 \| Z)$, $V_a' = h(CID_a, C, H_a, R, T_1)$, and then verifies whether $V_a' = V_a$. If it holds, B computes $\lambda = h(H_a, CID_a, V_a, T_1, T_2)$, $V_s = h(\lambda, H_a, T_1, T_2)$, and sends (V_s, T_2) to U_a.

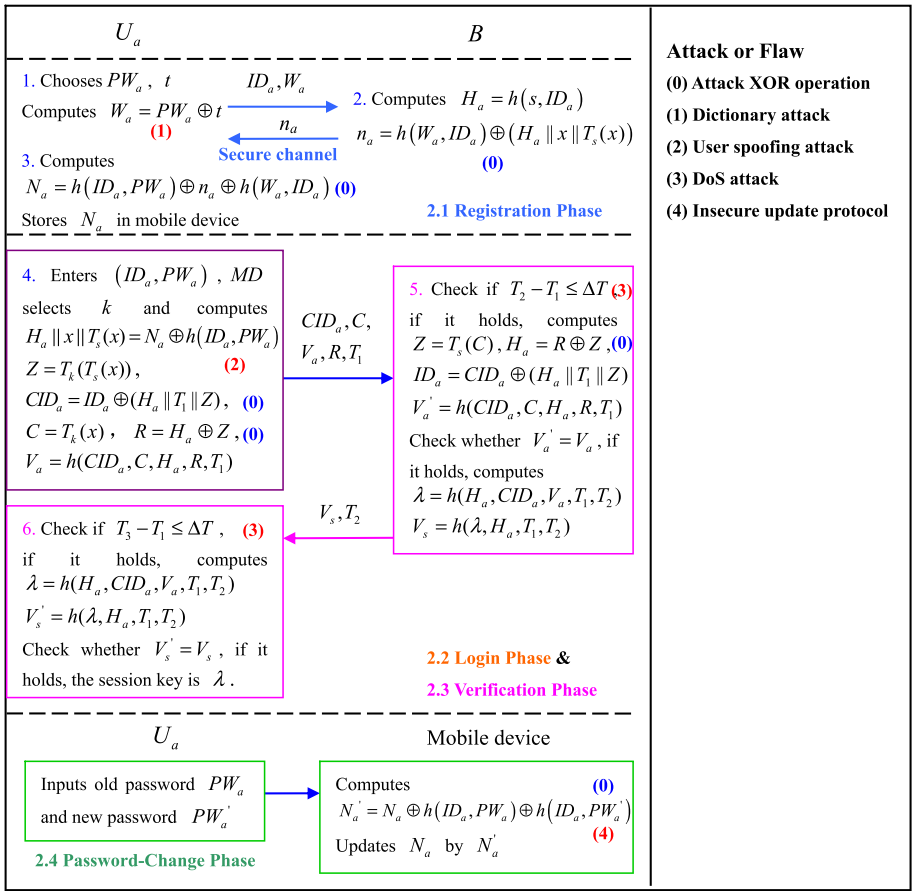

Figure 1. Lin's scheme and security flaws. XOR, exclusive-or operation; DoS, denial of service.

Security Comm. Networks 2015; **8**:2981–2991 © 2015 John Wiley & Sons, Ltd.
DOI: 10.1002/sec

Table I. Symbols.

Symbol	Definition
U_a, B, MD, Trudy	A user, a server, a mobile device, and an adversary, respectively
ID_a, ID_b	the identities of user U_a and server B, respectively
PW_a	the password of user U_a
$(x, T_s(x))$, s	the public key and secret key of server B
T_1	the current timestamp of user U_a and server B, respectively
T_2, T_3	the time of server B and user U_a receiving the message, respectively
ΔT	the time threshold
$h(\cdot)$	a collision-resistant one-way hash function
t, k R, r	random integer number
λ or SK	the session key
$\oplus$, $\|\|$	XOR operation and concatenation operation, respectively
$E_K()/D_K()$	a pair of secure symmetric encryption/decryption functions with the key K

Step 6 After receiving (V_s, T_2), U_a checks if $T_3 - T_1 \leq \Delta T$. If it holds, U_a computes $\lambda = h(H_a, CID_a, V_a, T_1, T_2)$, $V_s^{'} = h(\lambda, H_a, T_1, T_2)$, and then verifies whether $V_s^{'} = V_s$. If holds, U_a and B establish a shared secret key λ for subsequent communication.

2.4. Password-change phase

For changing the old password, U_a inputs his old and new passwords $(PW_a, PW_a^{'})$, and then MD computes $N_a^{'} = N_a \oplus h(ID_a, PW_a) \oplus h(ID_a, PW_a^{'})$. MD updates N_a as $N_a^{'}$.

3. SECURITY FLAWS IN LIN'S SCHEME

In Figure 1, we show some security flaws in Lin's scheme. Assume that Trudy is an adversary, who can obtain any messages transmitted among U_a, B, and MD.

3.1. Security flaw in the whole phase — attack exclusive-or operation

First of all, there exists a kind of wrong method about using $\oplus$ in the whole Lin's scheme. The exclusive-or (XOR) operation must assure the same binary digits on both sides of $\oplus$ (Figure 1(**0**)).

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

Case 1: Extended a.

However, a may be the *ID* of a user (such as in literature [4]), so the *ID* of a user is not practical and friendly enough.

Case 2: Shorten b.

However, b may be a random number (such as in literature [4]), if b is shortened, it can be easily guessed. And if the protocol transmits a (may be the *ID*) in plaintext, anyone will get the b.

Case 3: Pad a.

Definition 3.1. Leak attack.

Leak attack is a kind of intercept attack in which the attackers use various technologies to obtain the useful information from the messages eavesdropped from public channels.

Definition 3.2. The XOR with pad operation leaking attack.

This kind of attack is due to the use of XOR operation in a wrong way, which will lead to leak some sensitive information, and finally an adversary can obtain part of useful information, even the session key is not being detected.

In literature [4], Trudy can launch an XOR with pad operation leaking attack.

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

Assume that 0 is used to pad a. In the login phase, $CID_a = ID_a \oplus (H_a||T_1||Z)$, the length of ID_a is shorter than $(H_a||T_1||Z)$, then Trudy can use 0 to pad ID_a until the length of $(ID_a, 0...0)$ is equal to CID_a. Then Trudy computes $(H_a||T_1||Z)^* = CID_a \oplus (ID_a, 0...0)$. Compare $(H_a||T_1||Z)^*$ with $(H_a||T_1||Z)$, some parts of the middle of them must be equal to T_1 or the back part of T_1 (T_1 can be easily known.). Because $||$ is a concatenation operation, then Trudy can successfully extract Z. Then Trudy can compute $H_a = R \oplus Z$, and obtains the session key $\lambda = h(H_a, CID_a, V_a, T_1, T_2)$ because the messages $\{CID_a, V_a, T_1, T_2\}$ were transmitted on the public channel. Using this method, Trudy can successfully obtain the session key λ.

Finally, Trudy reveals all secrets including the session key. Assuming that the T_1 has l bits, the success probability of the given attack is $Adv_{success}^{l} = (1 - {}^{1}/_{2^l})$. For example, for $l \geq 8$, then the success probability is almost 1. And in general, the timestamp T_1 has at least 40 bits. Figure 2 describes the process of how to get H_a in the details.

3.2. Security flaw in registration phase — dictionary attack

In Step 1 of the registration phase, the protocol cannot resist dictionary attack. We know that PW_a is usually short

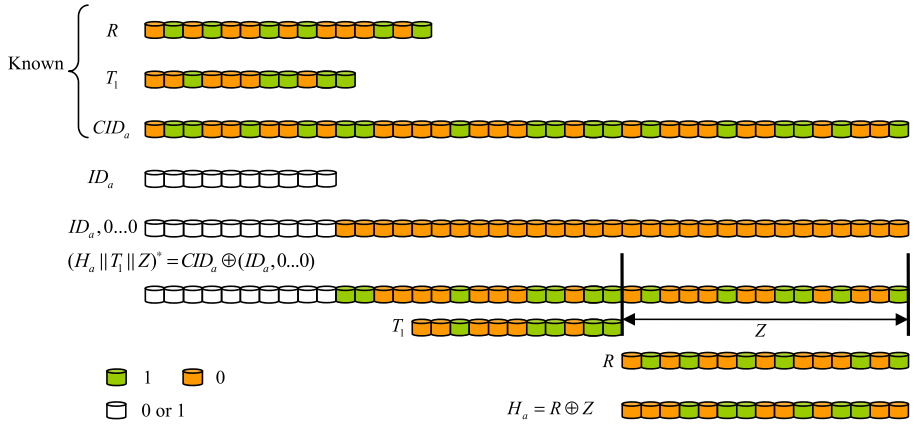

Figure 2. The process of how to obtain H_a.

and easily remembered. However, U_a computes $W_a = PW_a \oplus t$. If the curious server starts a dictionary attack (Figure 1(**1**)) then it can easily extract W_a as the length of t is short. Because t can be XORed with PW_a, and the lengths of t and PW_a should be the same.

3.3. Security flaw in login phase — user spoofing attack

In the login phase, the protocol cannot resist user spoofing attack. After U_a enters (ID_a, PW_a), the mobile device MD selects k and computes $H_a \| x \| T_s(x) = N_a \oplus h(ID_a, PW_a)$. However, Trudy can extract H_a (Figure 1(**2**)) from MD by some physical methods because $(x, T_s(x))$ is encapsulated in MD. Knowing H_a, she can masquerade as U_a to start a user spoofing attack anytime, even if she does not know the master secret key s.

3.4. Security flaw in verification phase — denial of service attack and ID_a exposed

In the verification phase, the protocol cannot resist DoS attack. After U_a sends the login request (CID_a, C, V_a, R, T_1), Trudy intercepts the message and delays to send it to B, or Trudy just alters T_1 to earlier time. When B receives it and checks if $T_2 - T_1 \leq \Delta T$ (Figure 1(**3**)) and it does not hold, the request is refused. Trudy succeeds to start a DoS attack. The same procedure happens in **Step 2** of the verification phase (Figure 1(**3**)).

In addition, in Step 1 of the verification phase, B computes $ID_a = CID_a \oplus (H_a \| T_1 \| Z)$. However, B does not use ID_a to verify any messages. The static ID_a is sensitive and protected in the dynamic CID_a, if ID_a is exposed, the scheme is insecure.

3.5. Security flaw in password-change phase — insecure update protocol

In the password-change phase, the protocol cannot provide a secure update protocol (Figure 1(**4**)). Obviously, if Trudy or any friend of U_a inputs (PW_a', PW_a''), MD computes $N_a' = N_a \oplus h(ID_a, PW_a') \oplus h(ID_a, PW_a'')$, and then N_a can be replaced by N_a' due to without the verification of old password PW_a, which will lead to the legal user cannot log in forever.

4. OUR IMPROVED PROTOCOL

Similar to Lin's protocol [4], our protocol also consists of four phases: registration phase, login phase, verification phase, and password-change phase. The details are described in Figure 3.

4.1. Registration phase

Step 1 Alice chooses his password PW_a and MD produces a random t to compute $W_a = h(PW_a \| t)$, and then sends (ID_a, W_a) to B via a secure channel.

Step 2 After receiving (ID_a, W_a), B computes $H_a = h(s \| ID_a)$, $n_a = h(W_a \| ID_a) \oplus H_a$, and sends $n_a, x, T_s(x)$ to Alice via a secure channel.

Step 3 After receiving $n_a, x, T_s(x)$, Alice computes

$$N_a = h(ID_a \| PW_a) \oplus n_a \oplus h(W_a \| ID_a) = h(ID_a \| PW_a) \oplus H_a,$$

and stores $N_a, x, T_s(x)$ in MD.

4.2. Login phase and verification phase

Step 4 Alice enters (ID_a, PW_a), and then MD chooses two random integers k, R to compute $H_a = N_a \oplus h(ID_a \| PW_a) = h(s \| ID_a)$, $T_k(x)$, $K_{AS} = T_k T_s(x)$, $H_A = h(H_a \| T_k(x) \| ID_a \| ID_b \| R)$, $C = E_{K_{AS}}(H_A \| ID_a \| ID_b \| R \| TS)$, where TS is a time stamp. The login request $C, T_k(x)$ is sent to B.

DOI: 10.1002/sec

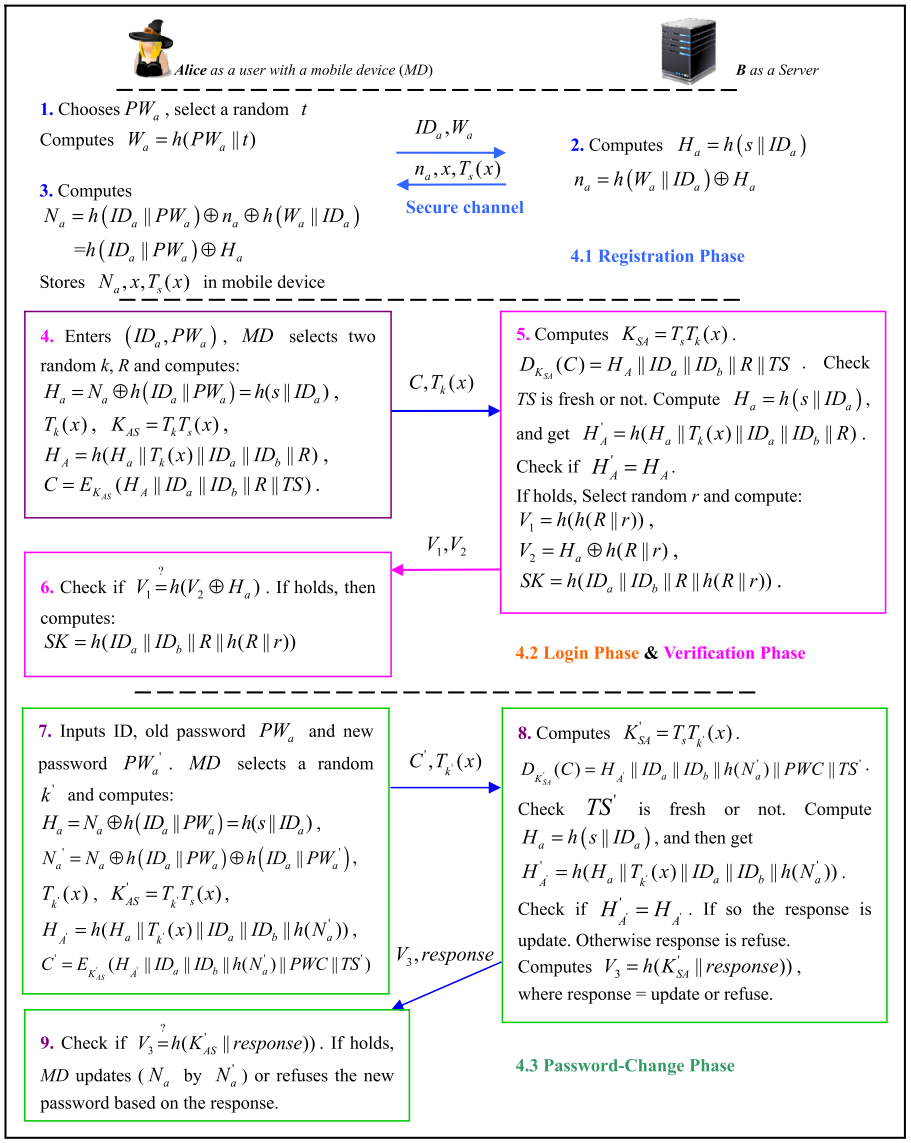

Figure 3. Our proposed protocol.

Step 5 After receiving $C, T_k(x)$, B computes $K_{SA} = T_s T_k(x)$ based on $T_k(x)$ and his secret key s. Then B decrypts C to obtain $H_A||ID_a||ID_b||R||TS$. B checks if the TS is fresh or not. Then B computes $H_a = h(s||ID_a)$, and then computes $H_A' = h(H_a||T_k(x)||ID_a||ID_b||R)$. Finally B checks if $H_A' = H_A$. If holds, B selects random r and computes $V_1 = h(h(R||r))$, $V_2 = H_a \oplus h(R||r)$, $SK = h(ID_a||ID_b||R||h(R||r))$, and sends V_1, V_2 to Alice. Otherwise, B terminates the process.

Step 6 After receiving V_1, V_2, MD checks if $V_1 \stackrel{?}{=} h(V_2 \oplus H_a)$. If holds, MD computes $SK = h(ID_a||ID_b||R||h(R||r))$ as the session key for subsequent communication. Otherwise, MD terminates the process.

4.3. Password-change phase

For changing the old password, Alice must communicate with server B to authenticate the old password. This is because there is no authenticated information stored in MD. No authenticated information stored in MD is to resist the offline guessing attack when MD is lost.

Step 7 Alice inputs his old and new passwords (PW_a, PW_a') with ID, then MD selects a random k' and computes: $H_a = N_a \oplus h(ID_a||PW_a) = h(s||ID_a)$, $N_a' = N_a \oplus h(ID_a||PW_a) \oplus h(ID_a||PW_a')$, $T_{k'}(x)$, $K_{AS}' = T_{k'} T_s(x)$, $H_{A'} = h(H_a||T_{k'}(x)||ID_a||ID_b||h(N_a'))$, and $C' = E_{K_{AS}'}(H_{A'}||ID_a||ID_b||h(N_a')||PWC||TS')$, where the PWC is an authenticator to help the server

recognize the password-change request, and $TS^{'}$ is a time stamp. *MD* sends $C^{'}, T_{k^{'}}(\mathrm{x})$ to server *B*.

Step 8 Server *B* computes $K_{SA}^{'} = T_s T_{k^{'}}(\mathrm{x})$ to decrypt $D_{K_{SA}^{'}}(C) = H_{A^{'}}||\mathrm{ID_a}||\mathrm{ID_b}||h(N_a^{'})||\mathrm{PWC}||TS^{'}$. *B* checks if the $TS^{'}$ is fresh or not. Then based on the authenticator *PWC*, Server *B* knows the request is the password-change phase. Next, *B* computes $H_a = h(\mathrm{s}||\mathrm{ID_a})$, and then computes $H_{A^{'}}^{'} = h(H_a||T_{k^{'}}(\mathrm{x})||\mathrm{ID_a}||\mathrm{ID_b}||h(N_a^{'}))$. *B* checks if $H_{A^{'}}^{'} = H_{A^{'}}$. If it holds, the response is update. Otherwise, the response is refuse. Finally, *B* computes $V_3 = h(K_{SA}^{'}||\mathrm{response}))$ where response = update or refuse and sends $\{V_3, \mathrm{response}\}$ to Alice.

Step 9 After Alice's *MD* receiving V_3, response, *MD* checks if $V_3 =^{?} h(K_{AS}^{'}||\mathrm{response}))$. If it holds, *MD* updates (N_a by $N_a^{'}$) or refuses the new password based on the response.

5. SECURITY CONSIDERATION AND EFFICIENCY ANALYSIS

5.1. Security features analysis

The section analyzes the security of our proposed protocol. First of all, using XOR operation in a wrong way can lead to many flaws. Let us assume that there are three secure components, including the two problems Chaotic Maps-Based Discrete Logarithm problem (CMBDLP) and Chaotic. Maps-Based Diffie–Hellman problem (CMBDHP) cannot be solved in polynomial-time, a secure one-way hash function, and a secure symmetric encryption. Assume that the adversary has full control over the insecure channel including eavesdropping, recording, intercepting, and modifying the transmitted messages. The definitions and analysis of the security requirements will be illustrated in Appendix A. The provable security under the random oracle mode will be illustrated in Appendix B. From Table II, we can see that the proposed scheme not only provides secure session key agreement and perfect forward secrecy, but also prevents the Key Compromise Impersonation (KCI) attacks. As a result, the proposed scheme is more secure and has much functionality compared with the recent related scheme.

Table II. Architecture and security of our proposed protocol.

Security requirements	Lin's scheme [4] (2014.5)	Our improved scheme
No verification table	Provided	Provided
Single registration	Provided	Provided
Mutual authentication	Provided	Provided
Impersonation attack	Not secure	Provided
Man-in-the-middle attack	Provided	Provided
Replay attack	Provided	Provided
Known-key security	Provided	Provided
Perfect forward secrecy	Provided	Provided
Data integrity	Provided	Provided
Guessing attacks	Not secure	Provided
Session key security	Provided	Provided
KCI attacks	N/A	Provided
Stolen mobile device attack	Provided	Provided
Stolen-verifier attack	Provided	Provided
Using XOR operation	Abnormal	Normal
XOR with pad operation leaking attack	Not secure	Provided
ID-theft attack	Provided	Provided
Curious server for getting password	Not secure	Provided
Secure update password	Not secure	Provided

N/A, 'not available' or 'not support'; KCI, Key Compromise Impersonation; XOR, exclusive-or operation.

5.2. Efficiency analysis

Compared with RSA and elliptic curve cryptography, Chebyshev polynomial computation problem offers smaller key sizes, faster computation, as well as memory, energy, and bandwidth savings. In our proposed protocol, no time-consuming modular exponentiation, and scalar multiplication on elliptic curves are needed. However, Xiao *et al.* [9] and Wang [10] proposed several methods to solve the Chebyshev polynomial computation problem.

For convenience, some notations are defined as follows:

- T_h: The time for executing the hash function;
- T_s: The time for executing the symmetric key cryptography;
- T_X: The time for executing the XOR operation; and
- T_C: The time for executing the $T_n(x)$ mod p in Chebyshev polynomial using the algorithm in literature [11].

Based on the literatures [11,12], the computational time of a one-way hashing operation, symmetric encryption/decryption operation, and Chebyshev polynomial operation is 0.2 ms, 0.45 ms, and 32.2 ms, separately. Moreover, the computational cost of XOR operation could be ignored when compared with other operations. Table III shows performance comparisons between our proposed scheme with some related ones including Niu-Wang's scheme (NW) [13] and Xue-Hong's scheme (XH) [12]. Because there are many

Table III. Efficiency of our proposed scheme with login and verification phase.

Roles	XH [12] (2012)	NW [13] (2011)	Our improved scheme
User	$2T_C + 3T_h + 3T_s \approx 66.35$ ms	$2T_C + 2T_h + 2T_s \approx 65.7$ ms	$T_C + 4T_h + 1T_s \approx 33.45$ ms
Server	$2T_C + 5T_h + 3T_s \approx 66.75$ ms	$2T_C + 2T_h + 2T_s \approx 65.7$ ms	$T_C + 5T_h + 1T_s \approx 33.25$ ms
Total	$2T_C + 5T_h + 3T_s \approx 133.1$ ms	$4T_C + 4T_h + 4T_s \approx 131.4$ ms	$2T_C + 9T_h + 2T_s \approx 66.7$ ms

XH, Xue-Hong's scheme; NW, Niu-Wang's scheme.

flaws and wrong ways with XOR operation, there are no performance comparisons with Lin's [4] scheme. Therefore, as in **Table** III, we can draw a conclusion that the proposed scheme has the lowest computational costs and is well suited to the mobile device settings.

6. CONCLUSION

Recently, Lin [4] proposed a chaotic maps-based mobile dynamic ID authenticated key agreement scheme and proved that it is secure against existential active attacks. In our paper, we have showed clearly that Lin's scheme is insecure against dictionary attack, user spoofing attack, and DoS attack. Most of all, using XOR operation in a wrong way can lead to these flaws. In addition, Lin's scheme cannot provide secure password update protocol. To overcome the weakness, we also proposed an improved protocol based on chaotic maps using dynamic ID in mobile client–server environment. Analysis shows our protocol could overcome the weakness in their protocol at decreasing the half-computational cost approximately. Therefore, our protocol is more suitable for practical applications.

REFERENCES

1. Xie Q, Zhao JM, Yu XY. Chaotic maps-based three-party password-authenticated key agreement scheme. *Nonlinear Dynamics* 2013; **74**(4):1021–1027.
2. Guo C, Chang CC. Chaotic maps-based password-authenticated key agreement using smart cards. *Communications in Nonlinear Science and Numerical Simulation* 2013; **18**(6):1433–1440.
3. Chain K, Kuo WC. A new digital signature scheme based on chaotic maps. *Nonlinear Dynamics* 2013; **74**(4):1003–1012.
4. Lin HY. Chaotic map based mobile dynamic ID authenticated key agreement scheme. *Wireless Personal Communications* 2014; **78**(2):1487–1494.
5. Das ML, Saxana A, Gulati VP. A dynamic ID-based remote user authentication scheme. *IEEE Transactions on Consumer Electronics* 2004; **50**(2):629–631.
6. Wang YY, Liu JY, Xiao FX, Dan J. A more efficient and secure dynamic ID-based remote user authentication scheme. *Computer Communications* 2009; **32**(4):583–585.
7. Khan MK, Kim SK, Alghathbar K. Cryptanalysis and security enhancement of a more efficient and secure dynamic ID-based remote user authentication scheme. *Computer Communications* 2011; **34**(3):305–309.
8. Tang HB, Liu XS. Cryptanalysis of a dynamic ID-based remote user authentication with key agreement scheme. *Informational Journal of Communication Systems* 2012; **25**(12):1639–1644.
9. Xiao D, Liao X, Deng S. A novel key agreement protocol based on chaotic maps. *Inf. Sci.* 2007; **177**:1136–1142.
10. Wang X, Zhao J. An improved key agreement protocol based on chaos. *Commun. Nonlinear Sci. Numer. Simul* 2010; **15**:4052–4057.
11. Kocarev L, Lian S. *Chaos-Based Cryptography: Theory, Algorithms and Applications*. Springer: Berlin, 2011; 53–54.
12. Xue K, Hong P. Security improvement on an anonymous key agreement protocol based on chaotic maps. *Communications in Nonlinear Science and Numerical Simulation* 2012; **17**(7):2969–2977.
13. Niu Y, Wang X. An anonymous key agreement protocol based on chaotic maps. *Communications in Nonlinear Science and Numerical Simulation* 2011; **16**(4):1986–199.
14. Kar J, Majhi B. An efficient password security of three party key exchange protocol based on ECDLP, 12*th International Conference on Information Technology* (*ICIT* 2009), Bhubaneswar, India, Tata McGrow Hill Education Private Limited, pp. 75–78, 2009.
15. Katz J, Shin JS. Modeling insider attacks on group key-exchange protocols. I*n*: *Proceedings of the 12th ACM Conference on Computer and Communications Security–CCS*'05, *ACM*, pp. 180–189, 2005.
16. Bellare M, Pointcheval D, Rogaway P. Authenticated key agreement secure against dictionary attacks. In *Proc. of the Advances in Cryptology - EUROCRYPT 2000*, *LNCS*, Vol. **1807**. Springer: Verlag, 2000; 139–155.

APPENDIX A: SECURITY PROOF OF THE PROPOSED SCHEME

(1) Mutual authentication and key agreement

Definition A.1. Mutual authentication and key agreement refers to two parties authenticating each other suitably and getting the session key simultaneously.

Theorem A.1. *The proposed protocol can achieve mutual authentication and key agreement.*

Proof . Server B uses the secret key s to compute $K_{SA} = \mathrm{T}_s T_k(\mathrm{x})$, so the improved scheme allows B-authenticated Alice by checking whether $H_A^{'} = H_A$. Because only Alice's MD can compute the $H_a = h(\mathrm{s}||\mathrm{ID_a})$ by inputting the right $\mathrm{ID_a}$ and $\mathrm{PW_a}$.

Alice authenticates server B uses by checking whether V_1 equals $h(V_2 \oplus H_a)$. Because only B can compute $H_a = h(\mathrm{s}||\mathrm{ID_a})$ based on its secret key s.

As for the key agreement, after authenticating each other, the temporary $T_k(\mathrm{x})$, r, R, and the ID $\mathrm{ID_a}$, $\mathrm{ID_b}$ were already authenticated by B. So finally Alice and B can make the key agreement simultaneously.

(2) Impersonation attack

Definition A.2. An impersonation attack is an attack in which an adversary successfully assumes the identity of one of the legitimate parties in a system or in a communications protocol.

Theorem A.2. *The proposed protocol can resist impersonation attack.*

Proof A.2. An adversary cannot impersonate anyone of the Alice or B. The improved scheme has already authenticated each other between Alice and B (in section Appendix A.(1)) based on the secrets $\mathrm{s}, \mathrm{PW_a}$ and the nonces k, R, r. So there is no way for an adversary to have a chance to carry out impersonation attack.

(3) Man-in-the-middle attack

Definition A.3. The man-in-the-middle attack is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them and making them believe that they are talking directly to each other over a private connection; when in fact, the entire conversation is controlled by the attacker.

Theorem A.3. *The proposed protocol can resist man-in-the-middle attack.*

Proof A.3. Because C, V_1, V_2 contain the participants' identities, a man-in-the-middle attack cannot succeed.

(4) Replay attack

Definition A.4. A replay attack is a form of network attack in which a valid data transmission is maliciously or fraudulently repeated or delayed.

Theorem A.4. *The proposed protocol can resist replay attack.*

Proof . An adversary cannot start a replay attack against our scheme because of the timestamp in each session for verification phase and password-change phase. At the same time, any adversary cannot modify the timestamp because it is encrypted in the $C, C^{'}$. If the adversary wants to launch the replay attack successfully, it must compute and modify $T_k(\mathrm{x})$ and C correctly, which is impossible.

(5) Known-key security

Definition A.5. Known-key security is a protocol that can protect the subsequent session keys from disclosing even if the previous session keys are revealed by the intendant user.

Theorem A.5. *The proposed protocol can achieve known-key security.*

Proof . Because the session key $SK = h(\mathrm{ID_a}||\mathrm{ID_b}||R||h(R||r))$ is dependent on the random nonces k, R, and r, and the generation of nonces is independent in all sessions, an adversary cannot compute the previous and future session keys when the adversary knows one session key. And in the password-change phase, any session key is only used once, so it has known-key security attribute.

(6) Perfect forward secrecy

Definition A.6. An authenticated multiple key establishment protocol provides perfect forward secrecy if the compromise of both of the node's secret keys cannot result in the compromise of previously established session keys [14].

Theorem A.6. *The proposed protocol can achieve perfect forward secrecy.*

Proof . In the improved scheme, the session key $SK = h(\mathrm{ID_a}||\mathrm{ID_b}||R||h(R||r))$ is related with R and r, which were randomly chosen by Alice and the server B, respectively. So any session key is not related with the secret key (such as $\mathrm{s}, \mathrm{PW_a}$) of each of participants. Furthermore, because of the intractability of the CMBDLP and CMBDHP problem, an adversary cannot compute the previously established session keys.

(7) Data integrity

Definition A.7. Authentication multiple key establishment protocol is said to achieve the property of data integrity if

there is no polynomial time algorithm that can alter or manipulate the transmitted messages.

Theorem A.7. *The proposed protocol can achieve data integrity property.*

Proof . While each participant sends the sensitive data to another participant in the instance protocol by the communication channel, the adversary alters or manipulates the data and cheats one of the honest participants by relying on the wrong session keys.

If the adversary wants to alter or manipulate the message $C, T_k(x)$ of step 4 for cheating server B, the adversary will be detected in step 5. Because the adversary does not have the PW_a of Alice, then the adversary cannot compute $C = E_{K_{AS}}(H_A||ID_a||ID_b||R||TS)$, finally the adversary cannot pass the validation of B. As for V_1, V_2, the adversary cannot alter or manipulate it because V_1, V_2 are just two values of a secure hash, and Alice can verify V_1, V_2 by the local information H_a.

(8) Off-line dictionary/guessing attacks

Definition A.8. In an off-line dictionary/guessing attack, an attacker guesses a password or long-term secret key and verifies his/her guess, but he/she does not need to participate in any communication during the guessing phase. In an undetectable online guessing attack, an attacker searches to verify a guessed password or long-term secret key in an online transaction, and a failed guess cannot be detected and logged by the server.

Theorem A.8. *The proposed protocol can resist off-line dictionary/guessing attacks.*

Proof . In our proposed scheme of the authenticated key exchange phase, the undetectable online guessing attack will fail because after Step 5, B can authenticate Alice. The off-line dictionary/guessing attacks will not work against the proposed scheme because the password pw_A is not transmitted on the public channel at all. Even if the adversary obtains the Alice's *MD* and wants to execute off-line dictionary/guessing attacks, he will fail because the information $N_a = h(ID_a||PW_a) \oplus h(s||ID_a)$ is protected by the server's secret key s. Therefore, the proposed scheme can resist off-line dictionary/guessing attacks.

(9) Session key security

Definition A.9. A communication protocol exhibits session key security if the session key cannot be obtained without any long-term secrets.

Theorem A.9. *The proposed protocol can achieve session key security.*

Proof . In the authenticated key agreement phase and password and shared secret key among servers update phase, a session key $SK = h(ID_a||ID_b||R||h(R||r))$ is generated from R and r. These parameter values are different in each session, and each of them is only known by Alice and B. Whenever the communication ends between B and Alice, the key will immediately self-destruct and will not be reused. Therefore, assuming the attacker has obtained a session key, Alice will be unable to use this session key to decode the information in other communication processes. Because the random point elements R and r are all generated randomly and are protected by the CMBDLP, CMBDHP, a secure one-way hash function and the secure symmetric encryption, a known session key cannot be used to calculate the value of the next session key. Additionally, because the values R and r of the random elements are very large, attackers cannot directly guess the values R and r of the random elements to generate session key. Therefore, the proposed scheme provides session key security.

(10) Key Compromise Impersonation Attacks

Definition A.10. An adversary is said to impersonate a party B to another party A if B is honest, and the protocol instance at A accepts the session with B as one of the session peers, but there exists no such partnered instance at B [15]. In a successful KCI attack, an adversary with the knowledge of the long-term private key of a party A can impersonate B to A.

Theorem A.10. *The proposed protocol can resist* KCI *attack.*

Proof . We assume that an adversary can know Alice's secret password PW_a. Then, an adversary can impersonate server B to cheat Alice and to obtain the session key. But the above-mentioned process will not be achieved, and the attack course terminates at the beginning. Because an adversary cannot own the server's secret key s, and then an adversary cannot decrypt $C = E_{K_{AS}}(H_A||ID_a||ID_b||R||TS)$. Finally, the adversary cannot pass validation of Alice because the adversary cannot compute V_1, V_2. So the key compromise impersonation attacks will fail.

(11) Stolen mobile device attack

Definition A.11. Anyone obtains the mobile device in some way to execute some kind of attacks.

Theorem A.11. *The proposed scheme can resist stolen mobile device attack.*

Proof . It is very clear that the proposed scheme provides identity authentication on-line. Because there is no

authenticated information stored in *MD*, the offline guessing attack when *MD* is lost cannot execute. Therefore, the proposed scheme can resist stolen mobile device attack.

(12) Stolen-verifier attack

Definition A.12. An authentication protocol is vulnerable to the stolen-verifier attack if a malicious adversary can modify user's password stored in the remote server's password table.

Theorem A.12. *The proposed scheme can resist stolen-verifier attack.*

Proof. It is very clear that there is no verifier table including password in the remote server. So either the remote server or an adversary has no way to derive each user's real password.

(13) ID-theft attack

Definition A.13. Identity theft is the fraudulent use of a person's personal identifying information. An identity thief will use another person's personal information such as social security number, date of birth, mother's maiden name, or account numbers to open fraudulent new accounts. They will open new credit card or checking accounts, write drafts on existing checking accounts, charge on existing credit card accounts, or obtain new loans.

Theorem A.13. *The proposed scheme can resist ID-theft attack.*

Proof. All the transmitting information $C, T_k(\mathrm{x})$ and V_1, V_2 on public channel is encrypted by a one-way secure hash function or a pair of secure symmetric encryption/decryption functions. Furthermore, in the improved scheme, we integrate random numbers with user's identity to form a dynamic ID, which varies with each login request. So, the ID-theft attack will fail.

APPENDIX B: PROVABLE SECURITY OF THE PROPOSED SCHEME UNDER THE RANDOM ORACLE MODEL

The adversarial model [16] of a mutual authentication and key agreement protocol is introduced as follows. Assume that a client–server environment contains two types of participants: n users $\mathrm{U} = \{U_1, U_2, \ldots, U_i, \ldots, U_n\}$ and a server S. The ith instance of U_i is denoted by $\prod_U^i$, and the instance of the server is denoted by $\prod_S$. An adversary named A is a probabilistic polynomial time machine. Assume that A is able to potentially control all common communications in the proposed scheme via accessing to a set of oracles (as defined in the succeeding section). The public parameters are known by each participant.

(1) Extract(ID_i) *query*: In *Extract query* model, A is able to obtain the private key of ID_i.
(2) $\mathrm{Send}\left(\prod_c^k, \mathrm{M}\right)$ *query*: In *Send query* model, A can send a message M to the oracle $\prod_c^k$, where $c \in \{U, S\}$. When receiving the message M, $\prod_c^k$ responds to A according to the proposed scheme.
(3) $h(\mathrm{m}_i)$ *query*: In this query, when A makes this hash query with message m_i, the oracle $\prod_c^k$ returns a random number r_1 and records (m_i, r_1) into a list L_H. The list is initially empty.
(4) $\mathrm{Reveal}\left(\prod_c^k\right)$ *query*: In this query model, A can obtain a session key sk from the oracle $\prod_c^k$ if the oracle has accepted. Otherwise, $\prod_c^k$ returns to a null A.
(5) Corrupt(ID_i) *query*: A can issue this query to ID_i and obtains back its private key.
(6) $\mathrm{Test}\left(\prod_c^k\right)$ *query*: When A asks this query to an oracle $\prod_c^k$, the oracle chooses a random bit $b \in \{0, 1\}$. If $b = 1$, then $\prod_c^k$ returns the session key. Otherwise, it returns a random value. This query measures the semantic security of the session key.

In this model, A can make *Send*, *Reveal*, *Corrupt* and *Test* queries. Note that the capabilities of the adversary can make finite queries under adaptive-chosen message attacks.

In the next part, we show that the proposed scheme can provide the secure authentication and key agreement under the computational CMDHP assumption.

Theorem 1. Assume that A can violate the proposed scheme with a non-negligible advantage ε and makes at most $\mathrm{q}_u, \mathrm{q}_s, \mathrm{q}_h$ queries to the oracle of the user $\prod_U^i$, oracle of the server S, and h, respectively. Then we can construct an algorithm to solve the CMDHP problem with a non-negligible advantage.

Proof. We first assume the type of attack, which is impersonating the user to communicate with server. Then we can construct an algorithm to solve the CMDHP problem.

For an instance of CMDHP problem $\{\mathrm{x}, P_1 = T_{k_i}(\mathrm{x}), P_2 = k_i\}$, B simulates the system initializing algorithm to generate the system parameters $\{\mathrm{x}, P_{\mathrm{pub}-u} = P_1, h\}$, h are random oracles controlled by B. Then, B gives the system parameters to A. B interacts with A as follows.

h-query: B maintains a list L_h of tuples (str_i, h_i). When A queries the oracle h on (str_i, h_i), B responds as follows:

If str_i is on L_h, then B responds with h_i. Otherwise, B randomly chooses an integer h_i that is not found in L_h, and adds (str_i, h_i) into L_h, then responds with h_i.

Reveal-query: When the adversary A makes a $\mathrm{Reveal}\left(\prod_c^m\right)$ query, B responds as follows. If $\prod_c^m$ is not

accepted, B responds none. Otherwise, B examines the list L_h and responds with the corresponding h_i.

Send-query: When the adversary A makes a $\text{Send}\left(\prod_c^m, "start"\right)$ query, B responds as follows. If $\prod_c^m = \prod_u^m$, B sets $T_k(x) \leftarrow P_1$, and randomly generates the value C. Otherwise, B generates a random number k^*, and computes $T_k(x) \leftarrow T_{k^*}(x)$, $K_{AS}^* = T_{P_2}(T_{k^*}(x))$, $C^* = E_{K_{AS}^*}\left(H_A^* || \text{ID}_a^* || \text{ID}_b^* || R^* || TS^*\right)$, and responds with $\{C^*, T_{k^*}(x)\}$. The simulation works correctly because A cannot distinguish whether C^* is valid or not unless A knows the identity ID_i and the password PW_i.

When the adversary A makes a $\text{Send}\left(\prod_c^m, (C^*, T_{k^*}(x))\right)$ query, B responds as follows. If $\prod_c^m = \prod_u^m$, B cancels the game. Otherwise, B computes $T_k(x) \leftarrow T_{k^*}(x)$, $K_{AS}^* = T_{P_2}(T_{k^*}(x))$, $C^* = E_{K_{AS}^*}\left(H_A^* || \text{ID}_a^* || \text{ID}_b^* || R^*\right)$, after decrypts C^* by using K_{AS}^*, then checks whether $H_A^{'} = H_A^*$ to authenticate the U_i. If it holds, B computes the session key $SK^* = h\left(\text{ID}_a^* || \text{ID}_b^* || R^* || h(R^* || r^*)\right)$. Then B responds the corresponding messages based on the description of the proposed scheme.

When the adversary A makes a $\text{Send}\left(\prod_c^m, \left(V_1^*, V_2^*\right)\right)$ query, B responds as follows. If $\prod_c^m = \prod_s$, B cancels the game. Otherwise, B computes $SK^* = h\left(\text{ID}_a^* || \text{ID}_b^* || R^* || h(R^* || r^*)\right)$.

If A can violate a user to the authentication, it means that A can obtain the values of $C = E_{K_{AS}}(H_A || \text{ID}_a || \text{ID}_b || R)$ or V_1, V_2, and then obtain the $SK = h(\text{ID}_a || \text{ID}_b || R || h(R || r))$ from the list L_h. Therefore, if A can violate a user to the server authentication, then B is able to solve the CMDHP problem with non-negligible probability. It is contradicting to the intractability of the CMDHP problem. From the above-mentioned analysis, we can see that the probability that A can violate the user to the server authentication is negligible.