A secure key agreement protocol based on chaotic maps*

Wang Xing-Yuan(王兴元)† and Luan Da-Peng(栾大朋)

Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, *China*

(Received 6 March 2013; revised manuscript received 22 May 2013)

To guarantee the security of communication in the public channel, many key agreement protocols have been proposed. Recently, Gong *et al*. proposed a key agreement protocol based on chaotic maps with password sharing. In this paper, Gong *et al*.'s protocol is analyzed, and we find that this protocol exhibits key management issues and potential security problems. Furthermore, the paper presents a new key agreement protocol based on enhanced Chebyshev polynomials to overcome these problems. Through our analysis, our key agreement protocol not only provides mutual authentication and the ability to resist a variety of common attacks, but also solve the problems of key management and security issues existing in Gong *et al*.'s protocol.

Keywords: chaos, Chebyshev polynomials, key agreement protocol, security

PACS: 05.45.Vx, 05.45.–a **DOI:** 10.1088/1674-1056/22/11/110503

1. Introduction

A chaotic system is characterized by sensitive dependence on initial conditions, pseudo-randomness, and ergodicity.[1] In the past few years, cryptography based on chaos theory has been studied widely as chaos features characterize excellent properties of diffusion and confusion, which are required in cryptosystems. In general, the cryptosystem is divided into two categories: secret key cryptosystem and public key cryptosystem. The secret key cryptosystem is also referred to as the symmetric encryption system. Much work has been done by incorporating chaos theory into the design of symmetric encryption schemes,[2–5] S boxes,[6] and hash functions.[7] The public key cryptosystem is different from the secret key cryptosystem, because the encryption key and the decryption key are not the same. Generally, the recipient has a public key that each sender can easily obtain, while anther key called the private key is only known to the recipient itself. One only knows the public key and intends to calculate the private key, which is not feasible, or computationally infeasible. Compared with many studies on symmetric encryption, the public key system and the key agreement protocol based on the chaos theory have not received the attention they deserve.

The key agreement protocol is a key establishment technique where a shared secret key is derived by two or more specified parties as a function of information contributed by, or associated with, each of these, such that no party can predetermine the resulting value.[8] With a shared key, communication parties can use the shared key to carry out the transmission of information in the subsequent communication. Therefore, designing cryptographic protocols is an important goal of information security over the public channel.

In recent years, Kocarev *et al*. proposed a public key encryption using the semi-group property of the Chebyshev chaotic map.[9] Shortly after, Bergamo *et al*. demonstrated that Kocarev *et al*.'s protocol is not safe against their cipher attack.[10] As the Chebyshev polynomial has the periodicity of cosine function, an attacker can efficiently recover plaintext even without the private key. In 2005, Bose proposed a key agreement protocol[11] over an insecure channel which used a plurality of chaotic systems and a set of linear functions. However, Wang *et al*. pointed out that the system is also unsafe and risks a successful attack.[12] In addition, Zhang also proved that Bose's protocol is neither safe nor efficient.[13] Xiao *et al*. proposed a chaos-based key agreement protocol,[14] which utilized an efficient chaotic public-key cryptosystem and claimed that its security is enhanced to resist the common attacks. Unfortunately, Han[15] analyzed this protocol and proposed different approaches to prevent the establishment of the session key. Xiang *et al*. also made security analysis on Xiao *et al*'s protocol[16] and pointed out its security issues. Chang *et al*. proposed a new key agreement protocol using passphrase,[17] which works with clock synchronization. Later on, they proposed a chaos-based key agreement protocol which can be regarded as an improved version of Chang *et al*.'s and claimed that the protocol can work without clock synchronization.[18] However, Yoon *et al*. pointed out that the protocol with nonce cannot resist replay attack.[19] To preserve user anonymity and improve security, Tseng *et al*. proposed a new key agreement protocol based on Chebyshev chaotic maps.[20] In 2011, however, Niu and Wang demonstrated that when the attacker is

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61370145, 61173183, and 60973152), the Doctoral Program Foundation of Institution of Higher Education of China (Grant No. 20070141014), the Program for Excellent Talents in Universities of Liaoning Province, China (Grant No. LR2012003), the Natural Science Foundation of Liaoning Province, China (Grant No. 20082165), and the Fundamental Research Funds for the Central Universities of China (Grant No. DUT12JB06).

†Corresponding author. E-mail: wangxy@dlut.edu.cn

© 2013 Chinese Physical Society and IOP Publishing Ltd

a legal user, Tseng *et al.*'s protocol is vulnerable to the insider attack.[21] They proposed a new protocol to overcome these weaknesses. Unfortunately, Xue *et al.*[22] and Yoon[23] pointed out the security problems in Niu-Wang's protocol, respectively. Yoon found that the protocol could not resist the Denial of Service (Dos) attack. In order to overcome the weaknesses in Niu-Wang's protocol, some other key agreement protocols using a smart card have been proposed,[22,24] while this type of protocol has some limitations. Recently, Gong *et al.* proposed a new chaos-based key agreement protocol without using smart cards,[25] and claimed that the protocol is safe. However, according to our analysis, Gong *et al.*'s protocol has the following security problems: (1) the shared password security faces threat of stolen-verifier attack; (2) the attacker may replay the intercepted messages, and the server will cost much resource to face the forged message flood; (3) the shared secret value in the whole network will cause the key management problems when the network scale becomes large enough. In order to overcome the weaknesses, in this paper we propose a new key agreement protocol.

This paper will be organized as follows. Section 2 gives a description of the Chebyshev chaotic map and some properties of it. Section 3 briefly reviews Gong *et al.*'s protocol. In Section 4, the vulnerability of Gong *et al.*'s protocol is analyzed. We propose a new key agreement protocol in Section 5. The security analysis of our protocol will be given in Section 6. In the last section, we conclude the paper.

2. Preliminaries

This section introduces some knowledge about the Chebyshev polynomial with semi-group property and some hard problems based on the Chebyshev chaotic map.

Definition 1 Let n be an integer, and let x be a variable taking values over the interval $[-1,1]$. The Chebyshev polynomial map $T_n : R \to R$ is defined by the following recursive relation:

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

where $T_0(x) = 1$ and $T_1(x) = x$.

According to Definition 1, the first few Chebyshev polynomials are

$$T_2(x) = 2x^2 - 1, \quad T_3(x) = 4x^3 - 3x,$$
$$T_4(x) = 8x^4 - 8x^2 + 1.$$

When $n > 1$, the Chebyshev polynomial map $T_n : T_n([-1,1]) \to [-1,1]$ of degree n is a chaotic map with its invariant density $f^*(x) = 1/(\pi\sqrt{1-x^2})$, for positive Lyapunov exponent $\ln n$.

Definition 2 Let n be an integer, and let x be a variable taking values over the interval $[-1,1]$. The Chebyshev polynomial $T_n(x) : [-1,1] \to [-1,1]$ is defined as

$$T_n(x) = \cos(n \arccos(x)), \quad -1 \leq x \leq 1.$$

If we only know the pair $(x, T_n(x))$, where $T_n(x)$ is a polynomial of degree n, then it is difficult to compute the degree n of the Chebyshev polynomial. The only way to solve this problem is through the brute-guess method, specifically, by computing $T_k(x)$ until we find k such that $T_k(x) = T_n(x)$, where $k = 2,3,4,\ldots$. However if n is a large enough number, the approach is impossible.

Definition 3 One of the most important properties of Chebyshev polynomials is the so-called semi-group property, namely,

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

Obviously, we can draw a conclusion: Chebyshev polynomials commute under composition. Thus,

$$T_r(T_s(x)) = T_s(T_r(x)).$$

Definition 4 There are several Chebyshev polynomials with the same value x but different degrees passing through the same point due to the periodicity of the cosine function. To enhance the property of the Chebyshev chaotic map, Zhang[13] proved that the semi-group property still holds for Chebyshev polynomials defined over interval $(-\infty, +\infty)$. We use the following enhanced Chebyshev polynomials:

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

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

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

Definition 5 The discrete logarithm problem (DLP). Given an element α, the task of DLP is to find the integer such that $T_r(x) \equiv \alpha$.

Definition 6 The Diffie–Hellman problem (DHP). Given the elements $T_r(x)$ and $T_s(x)$, the task of DHP is to compute $T_{rs}(x)$.

It is commonly believed that there is no polynomial-time algorithm to solve the DLP and DHP with non-negligible probability.

3. Review of Gong *et al.*'s key agreement protocol

In this section, Gong *et al.*'s key agreement protocol in Ref. [25] is reviewed.

3.1. Notations

In order to facilitate the understanding of the subsequent reading, symbols commonly used are provided in Table 1 with their meanings.

Table 1. Notations in Gong *et al.*'s protocol.

Symbol	Definition
A	User
B	Server
$\mathrm{ID_A}$	User A's identity
$\mathrm{PW_A}$	User A's password
H	A one-way hash function based on chaotic maps
$r_\mathrm{A}, r_\mathrm{B}$	Random number generated by A and B, respectively
$T_n(x)$	Chebyshev polynomial of degree n
$\oplus$	Bit exclusive-or operation

3.2. Gong *et al*'s key agreement protocol

Assume that A and B share the hash value $h_\mathrm{pw} = H(\mathrm{ID_A}, \mathrm{PW_A})$. Before performing the protocol, the server first publishes the public parameter. B selects a large prime number N, and generates a random number α. The public parameter is $\{\alpha, N, H\}$. Gong *et al.*'s protocol has four steps.

(1) A generates a random number r_A, computes $X = T_{r_\mathrm{A}}(\alpha) \bmod N, \bar{X} = X \oplus h_\mathrm{pw}$, and then sends $M_1 = \{\mathrm{ID_A}, \bar{X}\}$ to B.

(2) After receiving message M_1, B generates a random number r_B, computes $X = \bar{X} \oplus h_\mathrm{pw} = T_{r_\mathrm{A}}(\alpha), Y = T_{r_\mathrm{B}}(\alpha) \bmod N$, $K_\mathrm{B} = T_{r_\mathrm{B}}(X) \bmod N$, and $V_1 = H(Y, K_\mathrm{B})$, and then sends $M_2 = \{Y, V_1\}$ to A.

(3) When receiving M_2, A computes $K_\mathrm{A} = T_{r_\mathrm{A}}(Y) \bmod N$, and verifies whether V_1 is equal to $H(Y, K_\mathrm{A})$ or not. If it does not hold true, A stops here. Otherwise, B is authenticated. A computes $V_2 = H(X, Y, K_\mathrm{A})$, and the session key $\mathrm{SK_A} = H(K_\mathrm{A})$. A sends $M_3 = \{V_2\}$ to B.

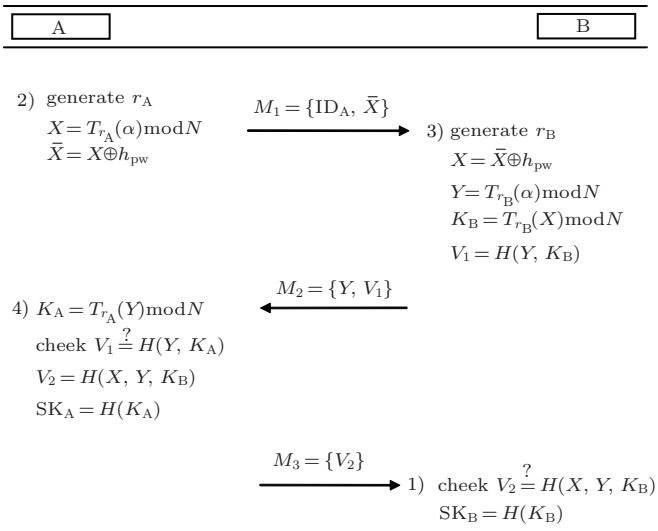


Fig. 1. Gong *et al.*'s protocol

(4) On receiving M_3, B checks whether V_2 is equal to $H(X, Y, K_\mathrm{A})$. If not, B stops the session. Otherwise, A is authenticated. B computes this session key: $\mathrm{SK_B} = H(K_\mathrm{B})$.

According to the information in Section 2 about Chebyshev polynomial, we know $K_\mathrm{A} = K_\mathrm{B}$, so $\mathrm{SK_A} = \mathrm{SK_B}$, which is the session key. Figure 1 describes the protocol performing process.

4. Security analysis of Gong *et al.*'s protocol

In Ref. [25], the authors claimed that their protocol provides mutual authentication, and resistance to replay attack, password-guessing attack, man-in-the-middle attack, and modification attack. However, in this paper we still find their protocol has unsatisfactory security weaknesses, and problems under some circumstances.

On the basis of the authors' own security analysis, the scheme introduced has provided mutual authentication and the ability to resist replay attack, man-in-the-middle attack, and modification attack as the attack cannot generate the legal message V_1 and V_2. In fact, the validation of these claims is completely dependent on the security of h_pw, which is shared between the communicating entities. More precisely, the security of the whole protocol relies on the security of $\mathrm{PW_A}$ for h_pw is the hash value of $\mathrm{ID_A}$ and $\mathrm{PW_A}$, where $\mathrm{ID_A}$ is often seen as public knowledge or easy to obtain. Therefore, if there is any possibility of leakage of $\mathrm{PW_A}$, the security of the protocol is compromised. Considering user's $\mathrm{PW_A}$ particularity, $\mathrm{PW_A}$ usually is not long and easily remembered for convenience. This further reduces the difficulty in finding $\mathrm{PW_A}$.

4.1. Stolen-verifier attack

Generally, server B must store a table of h_pw in order to verify the identity in the scheme. Supposing that B is hacked or its data is physically accessed by some other ways, the pivotal table h_pw would be exposed. The attacker can easily obtain A's h_pw, because he/she can just find PW such that $H(\mathrm{ID_A}||\mathrm{PW}) = h_\mathrm{pw}$, where $\mathrm{ID_A}$ is easy to obtain. A simple dictionary attack can sometimes accomplish this job because many users choose specific words as passwords for convenience of memorization. It is computationally feasible to enumerate all candidate values in password space to find the solution.

4.2. Clock synchronization

In some environments, the clock synchronization conditions are difficult to meet. However, the protocol apparently does not take into account the problem of clock synchronization. If there is no clock synchronization control, the following problem will occur: if an attacker intercepts the communication line between a user and a server and obtains some messages in the past communications, like message M_1', then the attacker replays M_1' to B in a future key agreement protocol run. Since B does not check the correctness of M_1', B will carry out a series of lookups and calculations, and send V_1 to A on receiving V_1. B will not pass the authentication. If an attacker floods the intercepted messages to B, the computing resource is inevitably wasted and reacts slowly or worse.

4.3. Key management

In Xiao *et al.*'s key agreement protocol and Gong *et al.*'s key agreement protocol, the secret key is assumed to be shared between A and B. However, there is a key management problem: if a key is shared between every two communicating entities, then there would be $n(n-1)/2$ keys, where n is the number of entities in a net. When the number increases, there will be a large amount of keys, which causes the key management problems.

4.4. Other problems

Supposing A wants to establish a connection with B and C, respectively. According to the agreement, if C deliberately publishes the same parameters as B, we know that entity C would share $h_{\rm pw}$ with A, which caused $h_{\rm pw}$ to be known by another entity. It is difficult to guarantee the probability of communication between B and C, or that the person who has the access to C is not interested in $h_{\rm pw}$. Then, C or the other person who obtains $h_{\rm pw}$ through C can impersonate A to communicate with B without being detected by B. Moreover, if there are N entities in a network, the security of $h_{\rm pw}$ cannot be ensured, and the stolen-verifier attack becomes easier as the attacker just needs to break one of the N entities to obtain $h_{\rm pw}$.

5. Our key agreement protocol

Taking Gong *et al.*'s key agreement protocol's security issues and key management problems into account, we propose a new key agreement protocol to solve these problems based on DLP, DHP, and enhanced Chebyshev polynomials. Regarding the structure, our protocol is similar to Neuman–Stubblebine protocol.[26] Assuming that Trend is a trusted third party in the network, for example, Trend may be the authentication center or the key distribution center. Before performing the protocol, Trend shares a different secret key with each participant. Like Gong *et al.*'s protocol, B publishes its public parameters $param = \{\alpha, N, H\}$. The proposed protocol is described as follows.

(1) A $\rightarrow$ Trend : $\mathrm{ID_A}$, $E_{\rm TA}(\mathrm{ID_A}, \mathrm{ID_B}, T_{r_{\rm A}}(\alpha), T_{\rm A})$.

A first selects a random large number $r_{\rm A}$, computes $T_{r_{\rm A}}(\alpha)$, then concatenates $\mathrm{ID_A}$, $\mathrm{ID_B}$ $T_{r_{\rm A}}(\alpha)$ and a timestamp $T_{\rm A}$, and encrypts with the key that A shares with Trend. Finally, A sends $\mathrm{ID_A}$ and $E_{\rm TA}(\mathrm{ID_A}, \mathrm{ID_B}, T_{r_{\rm A}}(\alpha), T_{\rm A})$ to Trend.

(2) Trend $\rightarrow$ B : $E_{\rm TB}(\mathrm{ID_A}, \mathrm{ID_B}, T_{r_{\rm A}}(\alpha), T_{\rm A})$.

On receiving the message, Trend decrypts the message encrypted with the key he shared with A and checks whether A is a valid identity. If not, Trend stops here. Otherwise, Trend checks $|T' - T_{\rm A}| \leq \Delta T$, where T' is the time for Trend to receive the message, and ΔT is an acceptable time delay threshold. If not, Trend stops here. Otherwise, Trend encrypts the concatenated message $E_{\rm TB}(\mathrm{ID_A}, \mathrm{ID_B}, T_{r_{\rm A}}(\alpha), T_{\rm A})$ with the key shared with B, and sends it to B.

(3) B $\rightarrow$ A : $T_{r_{\rm B}}(\alpha)$, $\mathrm{AU_B}$.

B decrypts the message with the key shared with Trend, and verifies the correctness of $\mathrm{ID_B}$. If it does not hold, B stops here. Otherwise, B selects a random large number $r_{\rm B}$ and calculates $T_{r_{\rm B}}(\alpha)$, shared session key $\mathrm{SK_B} = T_{r_{\rm B}}(T_{r_{\rm A}}(\alpha))$, and authentication information $\mathrm{AU_B} = H(T_{r_{\rm A}}(\alpha), T_{\rm A}, \mathrm{SK_B})$. B sends $T_{r_{\rm B}}(\alpha)$ and $\mathrm{AU_B}$ to A.

(4) A $\rightarrow$ B : $\mathrm{AU_A}$.

A computes session key $\mathrm{SK_A} = T_{r_{\rm A}}(T_{r_{\rm B}}(\alpha))$ and $\mathrm{AU'_B} = H(T_{r_{\rm A}}(\alpha), T_{\rm A}, \mathrm{SK_A})$ after receiving message from B, then checks whether $\mathrm{AU'_B} = \mathrm{AU_B}$. If not, A stops here. Otherwise, B's identity is authenticated. A computes $\mathrm{AU_A} = H(T_{r_{\rm A}}(\alpha), T_{r_{\rm B}}(\alpha), \mathrm{SK_A})$ and sends it to B.

(5) B computes $\mathrm{AU'_A} = H(T_{r_{\rm A}}(\alpha), T_{r_{\rm B}}(\alpha), \mathrm{SK_B})$ and verifies $\mathrm{AU'_A} = \mathrm{AU_A}$. If not, B stops. Otherwise, A's identity is authenticated. Then, the session key is $\mathrm{SK_B}(\mathrm{SK_A})$.

6. Security analysis

We present security analyses of the proposed key agreement protocol here. Theoretical analysis indicates that the proposed protocol overcomes security problems in Gong *et al.*'s protocol and could effectively resist common attacks. Besides, it is efficient and practical.

(1) Ability to resist Bergamo *et al.*'s attack The basic premises are needed to successfully execute Bergamo *et al.*'s attack. First, the attacker could obtain the related elements $\alpha, T_{r_{\rm A}}(\alpha)$, and $T_{r_{\rm B}}(\alpha)$. The second condition is that Chebyshev polynomials pass the same point with different degree n due to the periodicity of cosine function. Obviously, the attacker can only obtain $\alpha, N, T_{r_{\rm B}}(\alpha)$, and $T_{r_{\rm A}}(\alpha)$ is transmitted in a encrypted form all the way. Only the sender and receiver could know $T_{r_{\rm A}}(\alpha)$. Furthermore, in our protocol, we choose enhanced Chebyshev polynomials, with the interval of x extended to $(-\infty, +\infty)$, to avoid the periodicity of cosine function. The second condition is not met. Thus, Bergamo *et al.*'s attack is ineffective.

(2) Mutual authentication Mutual authentication means that each communicating entity's identity is authenticated by the other entity in the same protocol. In our protocol, Trend is a trusted third party. Trend does not know the session key but only knows $T_{r_{\rm A}}(\alpha)$. Even if Trend is broken, the attack still cannot obtain the session key because he/she faces with DHP to compute $\mathrm{SK_A}(\mathrm{SK_B})$. Therefore, the attacker cannot compute $\mathrm{AU_B}$ or $\mathrm{AU_A}$. In Step (4), A checks $\mathrm{AU_B}$ to validate B's identity. In Step (5), B validates A by checking $\mathrm{AU_A}$. As $\mathrm{SK_A}(\mathrm{SK_B})$ can only be computed by A or B, our protocol provides mutual authentication.

(3) Resistance to stolen-verifier attack The stolen-verifier attack means that an adversary who steals the password verifier from the server can use it directly to masquerade as a legitimate user in an authentication run.[21] Since there are no any secret data shared between user and server, there is nothing to be stolen to help to attack the protocol or the

communication. Thus, our protocol meets resistance to stolen-verifier attack.

(4) Key management problem By bringing in a trusted third party, the key management becomes simpler. Trend shared just one key with each entity. There is no need to share a key between every two computers. Thus, in the whole network with n computers, only n keys will be sufficient.

(5) Clock synchronization In our protocol performing process, timestamp T_A is adopted and T_A has nothing to do with the clock of B. Hence, even if the clock is not synchronized, our proposed protocol still ensures the communication because A just needs to check his own timestamp.

(6) Resistance to replay attack Replaying attack means that an adversary first intercepts some communication data in the current key agreement protocol run, and then the adversary replays the intercepted data with the receiver in a future key agreement protocol run. Trend, A, and B can find the attack by verifying the correctness of timestamp T_A and AU_A. Therefore, the message intercepted in the past communications can be found, and our protocol is against the replay attack.

(7) Resistance to man-in-the-middle attack A man-in-the-middle attack means that an attacker can intercept, relay, substitute, or modify the information communicated between the sender and the receiver, and try to derive some private information that is significant to the communication parties. In our protocol, the attacker could not know the shared secret session key $SK_A(SK_B)$, so he/she could not forge messages. In addition, in Steps (4) and (5), A and B both check the correctness of received messages AU_B and AU_A, any modification can be checked. Therefore, our protocol can resist the man-in-the-middle attack.

(8) Other analysis Concerning the password guessing attack, since there are no shared secret keys, this type of attack is infeasible. Besides, Trend just takes the responsibility of computation and validation, which improves the security and efficiency. Table 2 compares our protocols and some other key agreement protocols in terms of computational costs. From Table 2, we can see that our key agreement protocol has a better performance. Some notations shown in the table are defined as: T_C, the time of executing Chebyshev chaotic maps operation; T_S, the time of executing symmetric encryption/decryption; T_H, the time of executing a chaotic hash function.

Table 2. Computation cost comparisons with other protocols.

	Xiao *et al.*'s protocol	Tseng *et al.*'s protocol	Our protocol
A	$2T_S+2T_C+T_H$	$2T_S+2T_C+3T_H$	$T_S+2T_C+T_H$
B	$2T_S+2T_C+T_H$	$2T_S+2T_C+2T_H$	$T_S+2T_C+2T_H$

7. Conclusions

We have reviewed and analyzed Gong *et al.*'s key agreement protocol, and proposed a new key agreement protocol based on Chebyshev chaotic maps. Our protocol overcomes some problems, such as stolen-verifier attack, clock synchronization, and key management problems, in Gong *et al.*'s protocol. Through theoretical analysis, we prove that our key agreement can not only have the ability to resist common attacks, but also eliminate security risks in Gong *et al.*'s protocol. Therefore, the proposed protocol has a high level of security and efficiency, and is suitable for practical applications.

References

[1] Liu B and Peng J 2004 *Nonlinear Dynamics* (Beijing: High Education Press) p. 13

[2] Mao Y B, Chen G Y and Lian S G 2004 *Int. J. Bifurcat. Chaos* **14** 3613

[3] Xiang T, Liao X F, Tang G P, Chen Y and Wong K 2006 *Phys. Lett. A* **349** 109

[4] Wang X Y and Jin C Q 2011 *Opt. Commun.* **285** 412

[5] Wang Z, Huang X, Li N and Song X N 2012 *Chin. Phys. B* **21** 050506

[6] Chen G, Chen Y and Liao X F 2007 *Chaos Soliton. Fract.* **31** 571

[7] Xiao D, Liao X F and Deng S J 2005 *Chaos Soliton. Fract.* **24** 65

[8] Menezes A, van Oorschot P and Vanstone S 1997 *Handbook of Applied Cryptography* (Boca Raton: CRC Press) p. 1

[9] Kocarev L and Tasev Z 2003 *Proceedings of the* 2003 *International Symposium on Circuits and Systems* **3** 28

[10] Bergamo P, D'Arco P, Santis A and Kocarev L 2005 *IEEE Tran. Circuits -I* **52** 1382

[11] Bose R 2005 *Phys. Rev. Lett.* **95** 098702

[12] Wang K, Pei W J, Zou L H, Cheung Y and He Z Y 2006 *Phys. Lett. A* **360** 259

[13] Zhang L H 2008 *Chaos Soliton. Fract.* **37** 669

[14] Xiao D, Liao X F and Deng S J 2007 *Inform. Sci.* **177** 1136

[15] Han S 2008 *Chaos Soliton. Fract.* **38** 764

[16] Xiang T, Wong K W and Liao X F 2009 *Chaos Soliton. Fract.* **40** 672

[17] Chang E and Han S 2006 *CBS-IS-2006 Technical Report* (Curtin University of Technology) 1

[18] Han S and Chang E 2009 *Chaos Soliton. Fract.* **39** 1283

[19] Yoon E J and Yoo K Y 2008 *Replay Attacks on Han et al.'s Chaotic Map Based Key Agreement Protocol Using Nonce* (Berlin: Springer) p. 533

[20] Tseng H R, Jan R H and Yang W 2009 *IEEE International Conference on Communications* (Dresden:) p. 1

[21] Niu Y J and Wang X Y 2011 *Commun. Nonlinear Sci. Numer. Simul.* **16** 1986

[22] Xue K P and Hong P L 2012 *Commun. Nonlinear Sci. Numer. Simul.* **17** 2969

[23] Yoon E J 2012 *Commun. Nonlinear Sci. Numer. Simul.* **17** 2735

[24] Lee C C, Chen C L, Wu C Y and Huang S Y 2012 *Nonlinear Dyn.* **69** 79

[25] Gong P, Li P and Shi W B A 2012 *Nonlinear Dyn.* **70** 2401

[26] Schneier B 1996 *Applied Cryptography: Protocol, Algorithms and Source Code in C* (2nd edn.) (New York: John Wiley & Sons) p. 1