Cryptanalysis of a key agreement protocol based on chaotic Hash

Debiao He

School of Mathematics and Statistics, Wuhan University, Wuhan, People's Republic of China

Email:hedebiao@163.com

Abstract*:* With the rapid development of theory and application of chaos, more and more researchers are focusing on chaos based cryptosystems. Recently, Guo et al.'s [X. Guo, J. Zhang, Secure group key agreement protocol based on chaotic Hash, *Information Science*s 180 (2010) 4069 – 4074] proposed a secure key agreement protocol based on chaotic Hash. They claimed that their scheme could withstand various attacks. Unfortunately, by giving concrete attacks, we indicate that Guo et al.'s scheme is vulnerable to the off-line password guessing attack. The analysis shows Guo et al.'s scheme is not secure for practical application.

Key words*: Chaos; Hash function; Key agreement; Chebyshev; Password guessing attack*

1. Introduction

The key agreement scheme plays an important role in secure communications. A key agreement scheme allows two or more parties to agree upon a secret common session key over a public network, which will be used in the future communications. Over the past decades, key agreement scheme based on chaos theory has been studied widely.

In 2005, Xiao et al. [9] proposed a chaos-based key agreement protocol, which utilized efficient chaotic public-key cryptosystem[5] to reduce computation costs. However, Alvarez [1] has demonstrated that the Xiao et al.'s scheme is vulnerable to a man-in-the-middle attack. To enhance the security , Xiao et al. [6] proposed an improved key agreement by assuming that all participants have a shared long-term secret key. However, Han [2] points out that the improved scheme can not resist replaying attacks. Han et al. [3], Xiao et al. [7] used time-stamps or nonces to enhance the security of scheme [6], respectively. Unfortunately, Guo et al.'s[4] pointed out that none of [3,6,7] can satisfy the contributory nature of key agreement, that is, the malicious server can predetermine the shared secret key. Guo et al.'s [4] also proposed an improved

key agreement based on the chaotic Hash function[8] and claimed their scheme could withstand various attacks. However, in this paper, by giving concrete attacks, we indicate that Guo et al. scheme is not secure against the password guessing attack.

2. Preliminaries

In this section, we will introduce some knowledge about Chebyshev chaotic map.

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

$$T_n(x)=\cos(n\arccos(x)) \qquad (1)$$

With Definition 1, the recurrence relation of Chebyshev polynomial is defined as:

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

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

Chebyshev polynomial exhibits the following two important properties:

(1) The semi-group property:

$$\begin{aligned} &T_r(T_s(x))=\cos(r\cos^{-1}(s\cos^{-1}(x))) \\ &=\cos(rs\cos^{-1}(x))=T_{sr}(x)=T_s(T_r(x)) \end{aligned} \qquad (3)$$

(2) The chaotic property:

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

It is commonly believed that there is no polynomial-time algorithm to solve the two following problems with non-negligible probability.

Definition 2. The discrete logarithm problem (DLP) is explained by the following: Given an element y, find the integer r, such that $T_r(x)=y$.

Definition 3. The Diffie-Hellman problem (DHP) is explained by the following: Given an elements $T_r(x)$ and $T_s(x)$, find $T_{rs}(x)$.

3. Review of Guo et al.'s scheme

In this section, we briefly review Guo et al.'s key agreement scheme based on chaotic Hash[4]. Assume that A and B (A is the user, B is the server) share the Hash value $h_{pw} = H(ID_A, PW_A)$ of A's random password PW_A and identification ID_A, where $H(\cdot)$ denotes the chaotic Hash function of paper [8]. There are two phases in Arshad et al.'s scheme: authentication phase and key agreement phase.

3.1. Authentication phase

(1) A generates a random number $r_a \in [-1,1]$, and sends the authenticated message $AU_A = \{ID_A, r_a, H(h_{pw}, r_a)\}$ to B.

(2) Upon receiving the authentication request message AU_A, B computes $H'(h_{pw}, r_a)$ and verifies whether $H'(h_{pw}, r_a) = H(h_{pw}, r_a)$. If not B stops here; otherwise, A is authenticated and B returns a message $AU_B = \{r_b, H(h_{pw}, r_a, r_b)\}$ to A, where rb is a random integer selected by B.

(3) After receiving AU_B, A computes $H'(h_{pw}, r_a, r_b)$ and checks whether $H'(h_{pw}, r_a, r_b) = H(h_{pw}, r_a, r_b)$. If yes, the mutual authentication is done. A computes $ACK = H(r_b \oplus h_{pw})$ as the acknowledgement message and sends it to B, where $\oplus$ is the bitwise XOR operator.

(4) B calculates $ACK' = H(r_b \oplus h_{pw})$. If the verification of $ACK' = ACK$ is successful, then B confirms that ACK is sent by A.

3.2. Key agreement phase

(5) A sends $M_1 = H(h_{pw}) \oplus H(T_r(r_a))$ to B, where r is a random integer.

(6) B computes $X = M_1 \oplus H(h_{pw}) = H(T_r(r_a))$ and sends $M_2 = \{(H(h_{pw}) \oplus H(X)) \oplus T_s(r_a), H(T_s(r_a))\}$ to A, where s is a random integer kept by B.

(7) While receiving M_2, A takes out his own copies of h_{pw} and $T_r(r_a)$, computes $T_s'(r_a) = H(H(h_{pw}) \oplus H(T_r(r_a))) \oplus H(H(h_{pw}) \oplus H(X)) \oplus T_s(r_a)$ and

validates whether $H(T_s'(r_a)) = H(T_s(r_a))$ or not. If it holds true, A believes that M_2 is sent by B and $T_s(r_a)$ is valid, at the same time, A computes $M_3 = H(h_{pw} \oplus T_s(r_a)) \oplus T_r(r_a)$ and sends M_3 to B.

(8) Now, B computes $T_r'(r_a) = M_3 \oplus H(h_{pw} \oplus T_s(r_a))$ and verifies whether $H(T_r(r_a)) = H(T_r'(r_a))$. If yes, B believes $T_r(r_a) = T_r'(r_a)$ and keeps $T_r(r_a)$ as a secret.

(9) A and B compute the shared secret key $k = T_r(T_s(r_a)) = T_s(T_r(r_a)) = T_{rs}(r_a)$, respectively.

5. Cryptanalysis of Guo et al.'s scheme

In password key agreement scheme that the user is allowed to choose his password, the user tends to choose a password that can be easily remembered for his convenience. However, these easy-to-remember passwords are potentially vulnerable to password guessing attack, in which an adversary can try to guess the user's password and then verify his guess. In general, the password guessing attack can be classified into online password guessing attack and offline password guessing attack. The adversary tries to use guessed passwords iteratively to pass the verification of the server in an online manner in online password guessing attack. While in offline password attack, the adversary intercepts some password-related messages exchanged between the user and the server, and then iteratively guesses the user's password and verifies whether his guess is correct or not in an offline manner. Online password guessing attacks can be easily thwarted by limiting the number of continuous login attempts within a short period. In an offline password guessing attack, since there is no need for the server to participate in the verification, the server cannot easily notice the attack. In [4], Guo et al. claimed that their protocol can resist the off-line password guessing attack. However, in this section, we will show that the off-line password guessing attack, not as they claimed, is still effective in Guo et al.'s protocol. In Guo et al.'s scheme, since all transcripts are transmitted over an open network, a benign (passive) adversary, can easily obtain the valid message transcript of $AU_A = \{ID_A, r_a, H(h_{pw}, r_a)\}$, $AU_B = \{r_b, H(h_{pw}, r_a, r_b)\}$, $ACK = H(r_b \oplus h_{pw})$, $M_1 = H(h_{pw}) \oplus H(T_r(r_a))$, $M_2 = \{(H(h_{pw}) \oplus H(X)) \oplus T_s(r_a), H(T_s(r_a))\}$ and

$M_3 = H(h_{pw} \oplus T_s(r_a)) \oplus T_r(r_a)$. Then the adversary can carry out the off-line password guessing attack as follows.

5.1. The first off-line password guessing attack

In the authentication phase of Guo et al.'s scheme, the information of the password is include in $AU_A = \{ID_A, r_a, H(h_{pw}, r_a)\}$ and $AU_B = \{r_b, H(h_{pw}, r_a, r_b)\}$. The adversary C can verify the correctness the password he guesses. The detail is described as follows.

(1) C guesses a candidate passwords PW_A^*, and computes $h_{pw}^* = H(ID_A, PW_A^*)$.

(2) C checks whether $H(h_{pw}^*, r_a)$ and $H(h_{pw}, r_a)$ are equal. If they are equal, the adversary can conclude that A's password $PW_A = PW_A^*$. Otherwise, adversary repeat steps (1) and (2) until the correct password is found.

The adversary also could verify the correctness of PW_A^* by checking whether $H(h_{pw}^*, r_a, r_b)$ and $H(h_{pw}, r_a, r_b)$ are equal.

5.2. The first off-line password guessing attack

In the key agreement phase of Guo et al.'s scheme, the information of the password is include in $M_1 = H(h_{pw}) \oplus H(T_r(r_a))$, $M_2 = \{(H(h_{pw}) \oplus H(X)) \oplus T_s(r_a), H(T_s(r_a))\}$ and $M_3 = H(h_{pw} \oplus T_s(r_a)) \oplus T_r(r_a)$, where $X = M_1 \oplus H(h_{pw}) = H(T_r(r_a))$. The adversary C can verify the correctness the password he guesses. The detail is described as follows.

(1) C guesses a candidate passwords PW_A^*, and computes $h_{pw}^* = H(ID_A, PW_A^*)$.

(2) C computes $H(T_r(r_a))^* = M_1 \oplus H(h_{pw}^*)$ and $T_s(r_a)^* = (H(h_{pw}) \oplus H(X)) \oplus T_s(r_a) \oplus (H(h_{pw}^*) \oplus H(T_r(r_a))^*)$.

(3) C checks whether $H(T_s(r_a)^*)$ and $H(T_s(r_a))$ are equal. If they are equal, the adversary can conclude that A's password $PW_A = PW_A^*$. Otherwise, adversary repeat steps (1), (2) and (3) until the correct password is found.

Please note, the above attacks are a brute-force method in essence, i.e. the attacker tries offline all possible passwords in a given small set of values. Even though such attacks are not very effective in the case of high entropy keys, they can be very damaging when the secret key is a password since the attacker has a non-negligible chance of winning. Then Guo et al.'s scheme is vulnerable to the off-line password guessing attack.

5. Conclusions

In this paper, we have shown that Guo et al.'s key agreement scheme based on chaotic Hash is vulnerable to the off-line password guessing attack. The analysis shows Arshad et al.'s authentication scheme is not for practical application.

Reference

[1]. G. Alvarez, Security problems with a chaos-based deniable authentication scheme, Chaos, Solitons & Fractals 26 (2005) 7 – 11.

[2]. S. Han, Security of a key agreement protocol based on chaotic maps, Chaos, Solitons & Fractals 38 (2008) 764 – 768.

[3]. S. Han, E. Chang, Chaotic map based key agreement with/out clock synchronization, Chaos, Solitons & Fractals 39 (2009) 1283 – 1289.

[4]. X. Guo, J. Zhang, Secure group key agreement protocol based on chaotic Hash, Information Sciences 180 (2010) 4069 – 4074.

[5]. L. Kocarev, Z. Tasev, Public key encryption based on Chebyshev maps, in: Proceedings of the 2003 IEEE Symposium on Circuits and Systems, vol. 3, Bangkok, TH, pp. 28 – 31.

[6]. D. Xiao, X. Liao, S. Deng, A novel key agreement protocol based on chaotic maps, Information Sciences 177 (2007) 136 – 1142.

[7]. D. Xiao, X. Liao, S. Deng, Using time-stamp to improve the security of a chaotic maps-based key agreement protocol, Information Sciences 178 (2008) 1598 – 11602.

[8]. D. Xiao, X. Liao, S. Deng, One-way Hash function construction based on the chaotic map with changeable-parameter, Chaos, Solitons & Fractals 24 (2005) 65 – 71.

[9]. D. Xiao, X. Liao, K. Wong, An efficient entire chaos-based scheme for deniable authentication, Chaos, Solitons & Fractals 23 (2005) 1327 – 1331.