LETTER *Special Section on Information and Communication System Security*

Cryptanalysis of Group Key Agreement Protocol Based on Chaotic Hash Function

Eun-Jun YOON[†a)] ***and*** **Kee-Young YOO**[††b)]**,** ***Members***

SUMMARY In 2010, Guo and Zhang proposed a group key agreement protocol based on the chaotic hash function. This letter points out that Guo-Zhang's protocol is still vulnerable to off-line password guessing attacks, stolen-verifier attacks and reflection attacks.

key words: *cryptanalysis, key agreement, chaotic maps, hash function, Chebyshev map*

1. Introduction

Chaos [1] is a universal, random-like and robust phenomenon in nonlinear systems. Over the past decades, cryptography based on chaos theory has been studied widely to exploit the idea of using chaotic dynamical systems in cryptography because of the random-like, unpredictable dynamics of chaotic systems. For instance, chaotic systems have been widely used to design secure communication protocols [2]–[14], cryptographic hash functions [15], [16], and secure digital image encryption schemes [17].

In 2007, Xiao et al. [8] proposed a novel key agreement protocol based on chaotic maps. However, Han et al. [10] pointed out that Xiao et al.'s protocol is insecure against the replay attacks. In 2010, Guo and Zhang [11] pointed out that none of [8], [10], [12] can satisfy the contributory nature of key agreement, that is, the malicious server can predetermine the shared secret key. They also proposed a secure group key agreement protocol based on the chaotic hash function [7], [13]–[16] to surmount the aforementioned flaws. However, this letter points out that Guo-Zhang's protocol is still vulnerable to off-line password guessing attacks and stolen-verifier attacks [18]–[20].

Guo-Zhang's protocol uses user's password to prove the security. So, the password based key agreement protocols can be vulnerable to off-line password guessing attacks since users usually choose easy-to-remember passwords. Actually, Guo-Zhang's protocol is vulnerable to off-line password guessing attacks in which an adversary can easily obtain the secret password of the legal user from the intercepted values. In addition, Guo-Zhang's protocol is susceptible to stolen-verifier attacks in which obtaining secret data, which are stored in a server, can allow an illegitimate user to login to a server as a legitimate user.

Manuscript received February 1, 2011.
Manuscript revised April 20, 2011.

[†]The author is with the School of Computer Engineering, Kyungil University, 33 Buho-Ri, Hayang-Ub, Kyungsan-Si, Kyungsangpuk-Do 712–701, South Korea.
[††]The author is with the School of Computer Science and Engineering, College of IT Engineering, Kyungpook National University, 1370 Sankyuk-Dong, Buk-Gu, Daegu 702–701, South Korea.
a) E-mail: ejyoon@kiu.ac.kr
b) E-mail: yook@knu.ac.kr (Corresponding Author)
DOI: 10.1587/transinf.E94.D.2167

2. Chaotic Map Based Hash Function

Conventional hash functions such as MD4, MD5 and SHA-1 are realized by a complicated method based on logical operations or multi-round iteration of some available ciphers. However, since the conventional hash functions are successfully attacked by Wang et al. [21], the construction of a new and more secure hash function has been studied by lots of researches in recent years. Chaos has been widely utilized to construct cryptographic hash function in the past decade for its interesting characteristics, such as sensitivity to tiny changes in initial conditions and parameters, mixing property, ergodicity, etc. Compared with conventional hash functions, the chaotic hash function based on high-dimensional cat map [7], [13]–[16] not only is simple and efficient, but also has strong diffusion and confusion capability, good collision resistance, extreme sensitivity to message and secret key (please refer to [7], [13]–[16] for more detailed information).

3. Review of Guo-Zhang's Protocol

Figure 1 illustrates Guo-Zhang's protocol. Before performing the key agreement protocol, assume that user A and server B secretly share the password 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 [7], [13]–[16], and ID_A and PW_A are juxtaposed as the pending message from left to right. The details of Guo-Zhang's protocol are as follows:

3.1 Authentication Phase

(1) $A \rightarrow B$: AU_A

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

(2) $B \rightarrow A$: AU_B

After receiving AU_A, B takes out his/her own copies of h_{pw}, computes $H'(h_{pw}, ra)$ and verifies whether $H(h_{pw}, ra) \stackrel{?}{=} H'(h_{pw}, ra)$. If not B stops here; otherwise, A is authenticated and B returns a message $AU_B =$

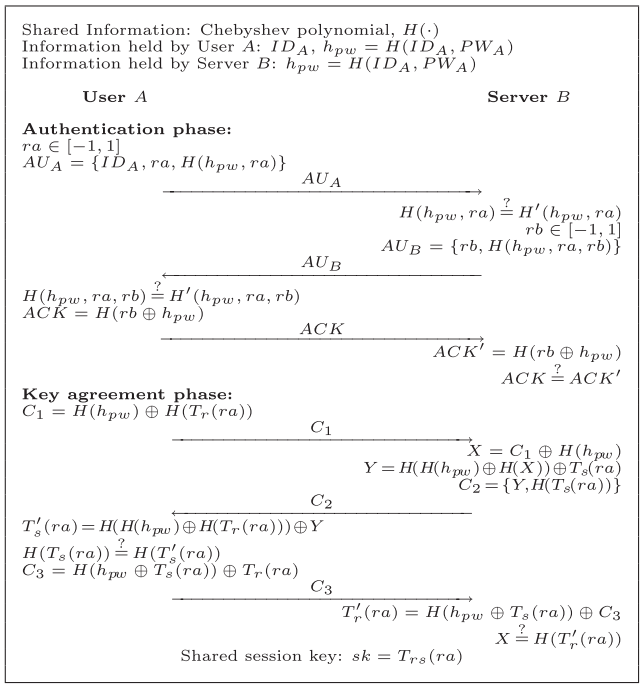

Fig. 1 Guo-Zhang's protocol.

$\{rb, H(h_{pw}, ra, rb)\}$ to A, where rb is a random integer selected by B.

(3) $A \rightarrow B$: ACK

After receiving AU_A, A takes out his/her own copies of ra and h_{pw}, computes $H'(h_{pw}, ra, rb)$ and checks whether $H(h_{pw}, ra, rb) \stackrel{?}{=} H'(h_{pw}, ra, rb)$. If yes, the mutual authentication is done. A computes $ACK = H(rb \oplus h_{pw})$ as the acknowledgement message and sends it to B, where $\oplus$ is the bitwise XOR operator.

(4) B takes out his/her own copies of h_{pw}, rb and calculates $ACK' = H(rb \oplus h_{pw})$. If the verification of $ACK = ACK'$ is successful, then B confirms that ACK is sent by A.

3.2 Key Agreement Phase

(1) $A \rightarrow B$: C_1

A sends $C_1 = H(h_{pw}) \oplus H(T_r(ra))$ to B, where r is a random integer, and $T_r(ra)$ denotes the Chebyshev polynomial of degree r.

(2) $B \rightarrow A$: C_2

B computes $X = C_1 \oplus H(h_{pw}) = H(T_r(ra))$ and $Y = H(H(h_{pw}) \oplus H(X)) \oplus T_s(ra)$, and then sends $C_2 = \{Y, H(T_s(ra))\}$ to A, where s is a random integer kept by B.

(3) $A \rightarrow B$: C_3

While receiving C_2, A takes out his/her own copies of h_{pw} and $T_r(ra)$, computes $T'_s(ra) = H(H(h_{pw}) \oplus H(T_r(ra))) \oplus Y$ and validates whether $H(T_s(ra)) \stackrel{?}{=} H(T'_s(ra))$ or not. If it holds true, A believes that C_2 is sent by B and $T_s(ra)$ is valid, at the same time, A computes $C_3 = H(h_{pw} \oplus T_s(ra)) \oplus T_r(ra)$ and sends C_3 to B.

(4) B computes $T'_r(ra) = H(h_{pw} \oplus T_s(ra)) \oplus C_3$ and verifies whether $X \stackrel{?}{=} H(T'_r(ra))$ or not. If it holds true, B believes $T_r(ra) = T'_r(ra)$ and keeps $T_r(ra)$ as a secret.

(5) A and B compute the shared session key $sk = T_r(T_s(ra)) = T_s(T_r(ra)) = T_{rs}(ra)$, respectively.

4. Cryptanalysis of Guo-Zhang's Protocol

This section shows that Guo-Zhang's protocol is vulnerable to off-line password guessing attacks and stolen-verifier attacks [18]–[20].

4.1 Off-Line Password Guessing Attacks

A guessing attack involves an attacker – randomly or systematically – trying long-term private keys (e.g., user passwords or server secret keys) one at a time, in a hope of finding the correct private key. Ensuring that long-term private keys are chosen from a sufficiently large space can reduce exhaustive searches. Most users, however, select passwords from a small subset of the full password space. Unlike typical private keys, a password has low entropy, and is constrained by the memory of the user. Such weak passwords with low entropy are easily guessed by using the so-called dictionary attack. Roughly speaking, the entropy of a user generated password is about 2 bits per character [19]. For example, one alphanumerical character has 6 bits of entropy, and thus the goal of the attacker, which is to obtain a legitimate communication party's password, can be achieved within a reasonable time. Therefore, the password guessing attacks [20] on Guo-Zhang's password-based key agreement protocol should be considered a real possibility.

4.1.1 Off-Line Password Guessing Attack 1

Let Adv be an active adversary who interposes the communication between A and B. Suppose that Adv has eavesdropped valid messages $AU_A = \{ID_A, ra, H(h_{pw}, ra)\}$ from an open network. It is easy to obtain the information since the messages are all exposed over the open network. Then the off-line password guessing attack 1 proceeds as follows:

(1*) To obtain the password PW_A of user A, the adversary Adv makes a guess at the secret password PW^*_A from dictionary D and computes $h^*_{pw} = H(ID_A, PW^*_A)$.

(2*) Adv computes $H(h^*_{pw}, ra)$ and checks if $H(h_{pw}, ra) \stackrel{?}{=} H(h^*_{pw}, ra)$. If it holds true, Adv then can guess the user A's password PW_A. Otherwise, Adv repeatedly

performs the Steps (1*) and (2*) until $H(h_{pw}, ra) = H(h^*_{pw}, ra)$.

The following illustrates the algorithm of an off-line password guessing attack 1.

Off-linePasswordGuessingAttack1$(ID_A, ra, H(h_{pw}, ra), D)$
{
  for $i := 0$ **to** $|D|$
  {
    $PW^*_A \leftarrow D$;
    $h^*_{pw} \leftarrow H(ID_A, PW^*_A)$;
    if $H(h_{pw}, ra) \stackrel{?}{=} H(h^*_{pw}, ra)$ **then return** PW^*_A
  }
}

4.1.2 Off-Line Password Guessing Attack 2

Suppose that Adv has eavesdropped valid messages $AU_A = \{ID_A, ra, H(h_{pw}, ra)\}$ and $AU_B = \{rb, H(h_{pw}, ra, rb)\}$ from an open network. Then the off-line password guessing attack 2 proceeds as follows:

(1*) To obtain the password PW_A of user A, the adversary Adv makes a guess at the secret password PW^*_A from dictionary D and computes $h^*_{pw} = H(ID_A, PW^*_A)$.

(2*) Adv computes $H(h^*_{pw}, ra, rb)$ and checks if $H(h_{pw}, ra, rb) \stackrel{?}{=} H(h^*_{pw}, ra, rb)$. If it holds true, Adv then can guess the user A's password PW_A. Otherwise, Adv repeatedly performs the Steps (1*) and (2*) until $H(h_{pw}, ra, rb) = H(h^*_{pw}, ra, rb)$.

The following illustrates the algorithm of an off-line password guessing attack 2.

Off-linePasswordGuessingAttack2$(ID_A, ra, rb, H(h_{pw}, ra, rb), D)$
{
  for $i := 0$ **to** $|D|$
  {
    $PW^*_A \leftarrow D$;
    $h^*_{pw} \leftarrow H(ID_A, PW^*_A)$;
    if $H(h_{pw}, ra, rb) \stackrel{?}{=} H(h^*_{pw}, ra, rb)$ **then return** PW^*_A
  }
}

4.1.3 Off-Line Password Guessing Attack 3

Suppose that Adv has eavesdropped valid messages $AU_A = \{ID_A, ra, H(h_{pw}, ra)\}$, $AU_B = \{rb, H(h_{pw}, ra, rb)\}$, and $ACK = H(rb \oplus h_{pw})$ from an open network. Then the off-line password guessing attack 3 proceeds as follows:

(1*) To obtain the password PW_A of user A, the adversary Adv makes a guess at the secret password PW^*_A from dictionary D and computes $h^*_{pw} = H(ID_A, PW^*_A)$.

(2*) Adv computes $ACK^* = H(rb \oplus h^*_{pw})$ and checks if $ACK \stackrel{?}{=} ACK^*$. If it holds true, Adv then can guess the user A's password PW_A. Otherwise, Adv repeatedly performs the Steps (1*) and (2*) until $ACK = ACK^*$.

The following illustrates the algorithm of an off-line password guessing attack 3.

Off-linePasswordGuessingAttack3(ID_A, rb, ACK, D)
{
  for $i := 0$ **to** $|D|$
  {
    $PW^*_A \leftarrow D$;
    $h^*_{pw} \leftarrow H(ID_A, PW^*_A)$;
    $ACK^* = H(rb \oplus h^*_{pw})$;
    if $ACK \stackrel{?}{=} ACK^*$ **then return** PW^*_A
  }
}

As a result, Guo-Zhang's protocol is susceptible to the off-line password guessing attacks. In the modern life which the Internet has strong influence to people, passwords are the most common means of user authentication on the Internet. For practical applications, password-based authentication schemes are required when making use of Internet network services like E-learning, on-line polls, on-line ticket-order systems, roll call systems, on-line games, etc. In real applications, users offer the same password as above to access several application servers for their convenience. Thus, the attacker may try to use the password PW^*_A to impersonate the user to login to other systems that the user has registered with outside this authentication server. If the targeted outside server adopts the normal authentication protocol, it is possible that the attacker can successfully impersonate the user to login to it by using the guessed password PW^*_A.

4.2 Stolen-Verifier Attacks

In most existing password-based authentication and key agreement protocols, the server stores the user's verifier (e.g. hashed passwords), rather than the user's bare password, in order to reduce the security of the breach once the server is compromised. Therefore, servers are always the targets of attacker, because numerous customers' secrets are stored in their databases. The stolen-verifier attacks [18], [19] mean that an adversary who steals a password-verifier from the server can use it *directly* to impersonate a legitimate user in a user authentication execution. Note that the main purpose of an authentication and key agreement protocol against the stolen-verifier attacks is to reduce the immediate danger to the authenticate user. In fact, an adversary who has a password-verifier may further mount a guessing attack.

In Guo-Zhang's protocol, the hash value $h_{pw} = H(ID_A, PW_A)$ of the user A's password PW_A, which is stored in the server, can be eavesdropped and then used to masquerade as the original user. Guo-Zhang did not explain the stolen-verifier attacks, with regard to obtaining the secret data $h_{PW} = H(ID_A, PW_A)$, which is stored in a server. This information can allow an illegitimate user to login to the server as a legitimate user. Suppose an adversary Adv has stolen the hash value $h_{PW} = H(ID_A, PW_A)$ from the server B in Guo-Zhang's protocol. Then, he/she can perform the user A impersonating attack based on the above authentication phase and key agreement phase in the Sects. 3.1 and 3.2. After finishing the phases, Adv and B will compute the same shared secret session key: $sk^* = T_{r^*s}(ra^*)$, where r^* and ra^* are random numbers chosen by Adv. As a result, Adv can continually impersonate the original user A by using sk^* in their subsequent communication.

By using the stolen $h_{pw} = H(ID_A, PW_A)$ and the above stolen-verifier attacks, the adversary Adv can also impersonate the server B in order to cheat the legal user. Therefore, Guo-Zhang's protocol is not secure against stolen-verifier attacks.

5. Conclusions

This letter pointed out that Guo-Zhang's group key agreement protocol based on the chaotic hash function is still vulnerable to off-line password guessing attacks and stolen-verifier attacks. As a result, there is no quick tweak that can be applied to make Guo-Zhang's protocol can withstand both off-line password guessing attacks and stolen-verifier attacks since the user's password PW_A and the secret data h_{PW} are easily obtained from open transmission channel and target server, respectively.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments in improving our manuscript. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2010-0010106).

References

[1] F. Dachselt and W. Schwarz, "Chaos and cryptography," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol.48, no.12, pp.1498–1509, 2001.
[2] G. Alvarez, "Security problems with a chaos-based deniable authentication scheme," Chaos, Solitions and Fractals, vol.26, no.1, pp.7–11, 2005.
[3] P. Bergamo, P. D'Arco, A. Santis, and L. Kocarev, "Security of public key cryptosystems based on Chebyshev polynomials," IEEE Trans. Circuits Syst. I, vol.52, no.7, pp.1382–1393, 2005.
[4] L. Kocarev and Z. Tasev, "Public key encryption based on Chebyshev maps," Proc. 2003 IEEE Symposium on Circuits and Systems, vol.3, pp.28–31, 2003.
[5] Y. Niu and X. Wang, "An anonymous key agreement protocol based on chaotic maps," Commun. Nonlinear Sci. Numer. Simul., vol.16, no.4, pp.1986–1992, 2011.
[6] X. Wang and J. Zhao, "An improved key agreement protocol based on chaos," Commun. Nonlinear Sci. Numer. Simul., vol.15, no.12, pp.4052–4057, 2010.
[7] D. Xiao, X. Liao, and S. Deng, "One-way hash function construction based on the chaotic map with changeable-parameter," Chaos, Solitions and Fractals, vol.24, no.1, pp.65–71, 2005.
[8] D. Xiao, X. Liao, and S. Deng, "A novel key agreement protocol based on chaotic maps," Inf. Sci., vol.177, no.4, pp.1136–1142, 2007.
[9] S. Han, "Security of a key agreement protocol based on chaotic maps," Chaos, Solitions and Fractals, vol.38, no.3, pp.764–768, 2008.
[10] S. Han and E. Chang, "Chaotic map based key agreement with/out clock synchronization," Chaos, Solitons and Fractals, vol.39, pp.1283–1289, 2009.
[11] X. Guo and J. Zhang, "Secure group key agreement protocol based on chaotic hash," Inf. Sci., vol.180, no.20, pp.4069–4074, 2010.
[12] D. Xiao, X. Liao, and S. Deng, "Using time-stamp to improve the security of a chaotic maps based key agreement protocol," Inf. Sci., vol.178, no.6, pp.1598–1602, 2008.
[13] E. Yoon and K. You, "An entire chaos-based biometric remote user authentication scheme on tokens without using password," Informatica, vol.21, no.4, pp.627–637, 2010.
[14] E. Yoon and I. Jeon, "An efficient and secure Diffie-Hellman key agreement protocol based on Chebyshev chaotic map," Commun. Nonlinear Sci. Numer. Simul., vol.16, no.6, pp.2383–2389, 2011.
[15] S. Deng, Y. Li, and D. Xiao, "Analysis and improvement of a chaos-based hash function construction," Commun. Nonlinear Sci. Numer. Simul., vol.15, no.5, pp.1338–1347, 2010.
[16] D. Xiao, X. Liao, and Y. Wang, "Improving the security of a parallel keyed hash function based on chaotic maps," Phys. Lett. A, vol.373, no.47, pp.4346–4353, 2009.
[17] H. Liu and X. Wang, "Color image encryption based on one-time keys and robust chaotic maps," Comput. Math. with Appl., vol.59, no.10, pp.3320–3327, 2010.
[18] C. Lin and T. Hwang, "A password authentication scheme with secure password updating," Comput. Secur., vol.22, no.1, pp.68–72, 2003.
[19] A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of applied cryptography, CRC Press, Boca Raton, 1997.
[20] B. Schneier, Applied cryptography: Protocols, algorithms, and source code in C, John Wiley & Sons, 1995.
[21] X. Wang and H. Yu, "How to break MD5 and other hash functions," Proc. Eurocrypt 2005, pp.19–35, 2005.