A Secure Hash-Based Strong-Password Authentication Scheme

Shuyao Yu[12], Youkun Zhang[3], Runguo Ye[12], Chuck Song[1]

[1]Computer Network Information Center, Chinese Academy of Sciences, No.4 South 4th Zhongguancun Street,China

[2]Institute of Computing Technology, Chinese Academy of Sciences, China
[3]Zhejiang Greatwall Reducer Cooperation, China

Abstract. Password authentication remains to be the most common form of user authentication. So far, many strong-password authentication schemes based on hash functions have been proposed, however, none is sufficiently secure and efficient. Based on the analysis of attacks against OSPA(Optimal Strong Password Authentication) protocol, we present a hash-based Strong-Password mutual Authentication Scheme (SPAS), which is resistant to DoS attacks, replay attacks, impersonation attacks, and stolen-verifier attacks.

1 Introduction

Despite the existence of more secure means for user authentication, including smart cards and biometrics, password-only user authentication continues to be the most common means in use for its simplicity, convenience, non-hardware-dependence. The last two decades have seen a new family of password-based authentication protocols which can withstand offline dictionary attacks and thus support weak password, like DH-EKE, SPEKE, SRP, AMP, however, these protocols employ public key cryptography which is formidable for computationally limited user devices, like PDA and mobile phone. With the increasingly widespread of mobile applications, there is an increasing call for a secure password-only computationally-light user authentication protocol.

Halevi and Krawczyk[2] proved that public key techniques are unavoidable for password-based authentication protocols to resist off-line guessing attacks. To avoid offline dictionary attacks, using strong password is the effective way for users to avoid expensive public key cryptographic computations without using smart cards or other alternative devices. A strong password is a password with high entropy, thus cannot be guessed easily. In this paper, we deal with strong-password authentication which only employs computationally light functions such as hash and bit-wise exclusive OR operations.

The rest of the paper is organized as follows: Section 2 introduces related work, we review OSPA protocol as a typical example of hash-based authentication protocols and analyzes its flaws in section 3; the proposed SPAS scheme is introduced in

Yu S., Zhang Y., Ye R. and Song C. (2005).
A Secure Hash-Based Strong-Password Authentication Scheme.
In *Proceedings of the 3rd International Workshop on Security in Information Systems*, pages 13-20
DOI: 10.5220/0002541700130020

section 4 and Section 5 analyzes its security features, Conclusions are drawn in section 6.

2 Related work

Since Lamport [9] first proposed the hash-based authentication protocol which has the property of dynamically changing password verifier, many similar schemes were proposed, e.g., S/Key [5][6][7]、CINON [11][12]、PERM [13], etc, but all of these protocols were later proven to have flaws [8][[10] [13] [14]. In 2000, Sandirigama et al. [1] proposed a simple strong-password authentication scheme called SAS. The authors claimed that SAS could resist man-in-the-middle attack. However, the SAS protocol suffers from vulnerabilities to replay attacks and DoS attacks which were pointed out by Lin et al. [3], who proposed a new scheme called OSPA (Optimal Strong-Password Authentication) protocol. However, Chen and Ku [4] showed OSPA protocol cannot withstand stolen-verifier attack.

After all these proposals were found to have flaws, researchers found it was too hard to propose a secure hash-based password-only authentication protocol to avoid all the stolen-verifier attacks, man-in-the-middle attacks, offline guessing attacks, denial-of-service attacks, so that several schemes based on the use of smart card were proposed [15][16][18][19][20][21]. Since using a smart card obviates the advantage of convenience of only using password to authenticate, herein we would not discuss about schemes based on the use of smart card.

In 2004, Ku [17] proposed a scheme without using smart card, although their scheme can withstand known attacks to OSPA, their scheme employs too many parameters and is unnecessarily complicated thus is inefficient.

3 OSPA and its vulnerabilities

In this section, we will introduce OSPA protocol as a typical example of hash-based strong-password authentication protocols, and analyze its vulnerabilities to DoS attacks, stolen-verifier attacks and replay attacks. We use notations as in Table 1.

Table 1. Notations

Notation	Description
A	the user
S	the server
P	user's password
N_i	the *i*th random nonce
h()	a cryptographic hash function
$h^i()$	apply h() function i times
	bit-wise XOR operation
||	string concatenation operation
k	server's secret key
$U_1 \rightarrow U_2$: mesg	U_1 sends mesg to U_2 through a public channel

3.1 Review of OSPA

The OSPA protocol is composed of two phases, the registration phase and the authentication phase. During the registration phase, user A wants to securely register with the server S for accessing services. A securely sends S registration message (ID_A, $h^2(P \oplus 1)$), and the server stores the registration message (ID_A, $h^2(P \oplus T)$, T=1) in its password verifier database.

During authentication phase, suppose that user A wants to login the *i*th time. The authentication phase includes the following steps:

1. A→S: ID_A, service request.
2. S→A: T(=i).
3. A→S: {ID_A, c_l, c_2, c_3}, where

$c_l = h(P \oplus i) \oplus h^2(P \oplus i)$,

$c_2 = h^2(P \oplus (i+1)) \oplus h(P \oplus i)$,

$c_3 = h^3(P \oplus (i+1))$.

4. Upon receiving the data in step 3, the server first checks if c1≠ c2 holds, if this holds, then computes:

$Y_1 = c_1 \oplus h^2(P \oplus i)$, where $h^2(P \oplus i)$ is retrieved from its password database,

$Y_2 = Y_1 \oplus c_2$,

$Y_3 = h(Y_2)$,

if $h(Y_1)$ equals the stored $h^2(P \oplus i)$, then the user is verified as a legitimate user, the server continues to check whether $Y_3 == c_3$, if this equation holds, which means c_2 and c_3 are integrity protected, so that the server can update its verifier file by replacing{ID_A, $h^2(P \oplus T)$, T=i} with {ID_A, $h^2(P \oplus T)$, T=i+1} for the (i+1)th authentication, where the value of server-side verifier $h^2(P \oplus (i+1))$ will take the value of Y_2.

One salient feature of OSPA is that although user's password remains unchanged, the server's verifier should be changed after each successful authentication. This feature of dynamically changing verifier will bring another advantage: even if the server is compromised and the verifier is stolen, the intruder cannot use this verifier to impersonate legitimate user since the intruder cannot derive $h(P \oplus T)$ from $h^2(P \oplus T)$. However, OSPA actually cannot withstand this stolen-verifier attack, which will be discussed in the next section.

3.2 Stolen-verifier attack on OSPA

Stolen-verifier attack means when an adversary compromises a server and steals its verifier database, the adversary may take advantage of the knowledge of the stolen verifier to launch DoS attack, impersonation attack and replay attack. In OSPA, suppose an attacker has stolen the server's stored verifier $h^2(P \oplus i)$ after A's (i-1)th login and intercepts the login message {ID_A, c_l, c_2, c_3} at the user's *i*th login. Then the at-

tacker can derive $h(P\oplus i)$ from $h^2(P\oplus i)\oplus c_1$. To control the following authentication verifier, the attacker may choose a new password P' and compute $c_2' = h^2(P'\oplus (i+1)) \oplus h(P\oplus i)$ and $c_3' = h^3(P'\oplus (i+1))$,the attacker then sends (ID_A,c_1,c_2',c_3') to the server to impersonate A to login. The server checks c_1 and concludes this is a legitimate user and finds the integrity protection of c_2' and c_3' holds, so that the server replaces $h^2(P\oplus i)$ with $h^2(P'\oplus (i+1))$ and set T=i+1. From now on, the legal user A will be rejected while the attacker gains the new password to impersonate the legitimate user. So the adversary succeeds in both stolen-verifier attack and DoS attack.

3.3 Replay attacks on OSPA

The attacker listens on the (i-1)th and *i*th authentication, then during the (i+1)th authentication, the attacker does the following:

- keeps $c_1^{(i+1)} = h(P\oplus (i+1))\oplus h^2(P\oplus (i+1))$ unchanged
- replaces $c_2^{(i+1)} = h^2(P\oplus (i+2))\oplus h(P\oplus (i+1))$ with $c_2'^{(i+1)} = c_1^{(i)}\oplus c_2^{(i)}\oplus c_1^{(i+1)} = h^2(P\oplus i)\oplus \mathbf{h(}P\oplus (i+1))$
- replaces $c_3^{(i+1)} = h^3(P\oplus (i+2))$ with $c_3'^{(i+1)} = h^3(P\oplus i)$

Upon receiving the modified authentication material, since $c_1^{(i+1)} \neq c_2'^{(i+1)}$, S calculates $y_1 = c_1^{(i+1)}\oplus h^2(P\oplus (i+1))$ and $y_2' = c_2'^{(i+1)}\oplus y_1$, since $h(y_1) = h^2(P\oplus (i+1))$ and $h(y_2') = c_3'^{(i+1)}$, S replaces $h^2(P\oplus (i+1))$ with $h^2(P\oplus i)$ and increases T as i+2. From now on, the attack can repeatedly send $\{h(P\oplus i)\oplus h^2(P\oplus i)$、$h^2(P\oplus (i+1))\oplus h(P\oplus i)$、$h^3(P\oplus (i+1))\}$and $\{h(P\oplus (i+1))\oplus h^2(P\oplus (i+1))$、$h^2(P\oplus i)\oplus h(P\oplus (i+1))$、$h^3(P\oplus i)\}$to impersonate legitimate user, which can also succeed in DoS attack to prevent legitimate user to login.

3.4 Predictable T attacks on OSPA

The parameter T will be increased by 1 after each successful authentication thus it is predictable. After listening and recording all the authentication messages before the *i*th authentication, the attacker can impersonate the server to send the user T>i, so that the adversary can get user's future authentic authentication messages and impersonate legitimate user. Furthermore, since in OSPA, user does not authenticate server, any adversary sending arbitrary T to user can successfully impersonate server.

4 The Proposed Scheme-SPAS

In this section, we will present the Strong-Password Authentication Scheme (SPAS), which can overcome the security vulnerabilities of OSPA protocol. SPAS is composed of two phases: registration phase and authentication phase.

During registration phase, user A registers to server S via a secure channel, A computes $V_1=h^2(S \| P \| N_1)$, where N_1 is the first nonce generated by S, S is the name of the server. The server stores ID_A, N_1 and $SV_1 = V_1 \oplus K_A$, where $K_A = H(ID_A \| k)$, where k is S's secret key for all its clients, which should be stored under strict protection.

step1: A→S: ID_A, N_A, service request
step2: S→A: N_i, $N_{i+1} \oplus h(h^2(S \| P \| N_i) \| N_A)$
step3: A→S: d_1, d_2, d_3
step4: S→A: $h(N_A \| N_{i+1} \| h^2(S \| P \| N_i))$

Fig. 1. SPAS authentication protocol

During the *i*th authentication phase, A generates a random nonce N_A and sends it along with his/her ID_A and service request.

Upon receiving service request from user A in step1, the server S retrieves SV_i corresponding to ID_A from its verifier file, and derives $h^2(S||P||N_i)$ by computing $SV_i \oplus h(ID_A||k)$, then generates a random nonce N_{i+1} for the next authentication phase and sends step2 messages.

When A receives step2 messages, A computes $h^2(S||P||N_i)$ by using the entered password, then derives N_{i+1} , if $N_i \neq N_{i+1}$ holds, then A calculates and sends d_1, d_2, d_3 as follows:

$$d_1 = h^2(S \| P \| N_i) \oplus h(S \| P \| N_i)$$

$$d_2 = h(S \| P \| N_i) \oplus h^2(S \| P \| N_{i+1})$$

$$d_3 = h(h^2(S \| P \| N_{i+1}) \| N_i \| N_{i+1})$$

Upon receiving step3 message, the server computes $y_1=d_1 \oplus h^2(S||P||N_i)$, and verifies whether $h(y_1)$ equals $h^2(S||P||N_i)$, if this holds, the user is believed to be legitimate, and the server calculates $y_2=d_2 \oplus y_1$, $y_3=h(y_2||N_i||N_{i+1})$, then verifies whether $y_3=d_3$ holds, if this equation holds, the server replaces i with i+1, and stores $SV_{i+1}=h(ID_A||k) \oplus y_2$, then sends $h(N_A||N_{i+1}||h^2(S||P||N_i))$ to the user to complete the authentication.

Finally, upon receiving $h(N_A|| N_{i+1}|| h^2(S||P||N_i))$, the user verifies this value and assure there is no server impersonation attack on the other side if this verification result is right.

5 Security Analysis

Since we require users to choose strong password with high entropy, SPAS is resistant to offline dictionary attacks given that the adversaries' power is limited compared to the power required to break a large dictionary; also, SPAS can mitigate online password guessing by counting the number of failed authentications at the server side and taking measures when suspected attack occurs; since we do not employ an increasing T to randomize each authentication messages, SPAS is resistant to predictable T attacks; We claim that SPAS can also thwart the following attacks: replay attacks, DoS attacks, both user and server impersonation attacks, more resistant to stolen-verifier attacks compared with OSPA, and can achieve mutual authentication. We further analyze how SPAS resists these attacks as follows.

5.1 Replay attacks

Suppose the adversary sniffs all the communication information before the *i*th authentication, and tries to replay the old messages for the *i*th authentication. Because the computation of d_1, d_2, d_3 all includes nonce which is chosen randomly by the server during every new authentication time, there is no chance for adversaries to succeed by replaying the old d_1, d_2, d_3. Furthermore. Also there is no chance for adversary to replace part of d_1, d_2, d_3 to launch replay attack since these three values are interrelated and any one of these three changes will cause the authentication failed.

5.2 Stolen-verifier attacks

In SPAS, we store SV_i as a verifier into the server's verifier file. To derive verifier $h^2(P \oplus i)$, we need the server's secret key k to compute $h^2(P \oplus i)=SV_i \oplus h(ID_A \| k)$, therefore SPAS can resist stolen-verifier attacks as long as server's secret key k is kept secure. To the worst end, if k and the verifier file are both compromised, the protocol would be vulnerable to stolen-verifier attack, just like the same attack on OSPA, which requires the adversary to first compromise the server and get the server's verifier, then intercept a legitimate user's authentication messages and modify the old password to his/her chosen password.

5.3 DoS attacks

Suppose the adversary tries to change d_2 to launch DoS attacks, this attempt can not succeed since d_3 protects the integrity of d_2, any modification made on legal d_1, d_2 or d_3 will be detected by S. The reason that SPAS can avoid this attack while OSPA cannot is that SPAS involves different random nonces in calculating d_3, while OSPA's message c_2 can be simply replaced by some calculation of former messages which can be verified by the former c_3.

5.4 Impersonation attacks

User impersonation attacks are prevented since the previous messages sent on the communication channel cannot help an adversary in calculating authentication messages needed in his/her current authentication. We employ a changing verifier to avoid replay attacks and thus prevent user impersonation. Server impersonation attack is prevented by requiring server to send $h(N_A \| N_{i+1} \| h^2(S\|P\|N_i))$ in step4 to prove that the server has the password verifier, and an adversary can not simply replay old messages to impersonate server because N_A is random and unique for every authentication protocol run so that a server impersonator can not generate a valid step4 message to complete server authentication.

6 Conclusion

In summary, it is not an easy task to design a secure and efficient password authentication protocol without requiring the use of storage devices like smart cards or the use of public key cryptographic techniques. Based on the analysis of security flaws of a typical hash-based authentication protocol-OSPA, we present a hash-based Strong-Password Authentication Scheme (SPAS), which can achieve mutual authentication and is resistant to DoS attacks, replay attacks, impersonation attacks, and stolen-verifier attacks.

References

1. M. Sandirigama, A. Shimizu and M. T. Noda: Simple and secure password authentication protocol (SAS). IEICE Transactions on Communications, vol. E83-B, no. 6 (2000)1363-1365
2. Halevi, S. and Krawczyk, H: Public-key cryptography and password protocols. In: Proceedings of 5th ACM Conference On Computer and Communications Security, San Francisco, CA (1998) 122–131
3. C. L. Lin, H. M. Sun, and T. Hwang: Attacks and solutions on strong-password authentication. IEICE Transactions on Communications, vol. E84-B, no. 9, (2001) 2622--2627
4. C.M. Chen and W.C. Ku: Stolen-verifier attack on two new strong-password authentication protocols. IEICE Transactions on Communications, vol. E58-B, no. 11, (2002)2519-2521
5. N.M. Haller: On internet authentication. RFC 1704, Oct. 1994
6. N.M. Haller: The S/KEY one-time password system. In: Proceedings of Internet Society Symposium on Network and Distributed System Security (1994)151–158
7. N.M. Haller: A one-time password system. RFC 1938, May 1996
8. C. Kaufman, R. Perlman, and M. Speciner. Network Security-Private communication in a public world. Prentice Hall (2002)
9. Leslie Lamport: Password authentication with insecure communication. Communications of the ACM, v.24 n.11 (1981)770-772
10. C.J. Mitchell and L. Chen. Comments on the S/KEY user authentication scheme. ACM Operating Systems Review, vol.30, no.4 (1996)12–16
11. A. Shimizu: A dynamic password authentication method by one-way function. IEICE Transactions, vol.J73-D-I, no.7 (1990) 630–636

12. A. Shimizu: A dynamic password authentication method by one-way function. System and Computers in Japan, vol.22, no.7(1991)
13. A. Shimizu, T. Horioka, and H. Inagaki: A password authentication method for contents communication on the Internet. IEICE Transactions of Communications., vol.E81-B, no.8 (1998)1666–1673
14. S.M. Yen and K.H. Liao: Shared authentication token secure against replay and weak key attacks. Information Processing Letters, vol.62 (1997) 77–80
15. Chih-Wei Lin, Jau-Ji Shen, Min-Shiang Hwang: Security enhancement for Optimal Strong-Password Authentication protocol. Operating Systems Review 37 (2003) 7-12
16. Wei-Chi Ku, Hao-Chuan Tsai, Shuai-Min Chen: Two simple attacks on Lin-Shen-Hwang's strong-password authentication protocol. ACM operating Systems Review, vol.37, no. 4 (2003) 26-31
17. Wei-Chi Ku: A Hash-Based Strong-Password Authentication Scheme without Using Smart Cards. ACM Operating Systems Review, vol.38, no.1 (2004) 29-34
18. Hung-Yu Chien, Jinn-ke Jan: Robust and Simple Authentication Protocol. Computer Journal 46(2) (2003)193-201
19. M.S.Hwang and L.H.Li: A new remote user authentication scheme using smart cards. IEEE Transactions on Consumer Electronics, vol.46, no.1 (2000) 28-30
20. C.C.Lee, M.S.Hwang, and W.P.Yang: A flexible remote user authentication scheme using smart cards. ACM operating Systems Review, vol.36, no.3,(2002) 46-52
21. Ya-Fen Chang, Chin-Chen Chang: A secure and efficient strong-password authentication protocol. ACM SIGOPS Operating Systems Review archive, Volume 38, Issue 3 (2004) 79 – 90