Security Enhancement for Optimal Strong-Password Authentication Protocol

Chih-Wei Lin[‡] Jau-Ji Shen[‡] Min-Shiang Hwang[†]

Department of Information Management,[‡]
Graduate Institute of Networking and Communication Engineering,[†]
Chaoyang University of Technology
168, Gifeng E. Rd., Wufeng, Taichung County, Taiwan 413, R.O.C.
Email: mshwang@mail.cyut.edu.tw

Abstract

In 2001, Lin et al. proposed an optimal strong-password authentication protocol called the OSAP protocol. However, Chen and Ku pointed out that it is vulnerable to the stolen-verifier attack. In this paper, we shall propose an improved version of the OSAP protocol to enhance the security.

Keywords: Authentication, cryptography, hash function, password, security, smart card.

I Introduction

In public network systems, user authentication is the most important part of the security. The identities of the communication parties must be verified before they start a new connection so that no harm is done. A variety of password-based authentication schemes have been proposed so far [1, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14]. Existing password authentication schemes can enable a legitimate user to login a remote server and access the resources.

In 2000, Sandirigama et al. [13] proposed a simple strong-password authentication scheme called SAS. The authors claimed that their can resist the "Man in the middle" attack and that it has the lowest storage, processing, and transmission overhead. However, the SAS protocol suffers from vulnerability to the replay attack and the denial of service attack by Lin et al. [11], who proposed a new scheme called the OSPA (Optimal Strong-Password Authentication) protocol. This protocol can protect the system against the stolen-verifier attack, replay attack and denial of service attack. Nevertheless, Chen and Ku [2] showed the OASP protocol, as well as the SAS protocol, cannot withstand the stolen-verifier attack.

In this article, we shall propose an improved scheme that makes the OSPA protocol able to withstand the stolen-verifier attack. This article is organized as follows: In Section 2, we shall review the attack by Chen and Ku on the OSPA protocol. Then, we shall propose an improved scheme to enhance the security of OSPA in Section 3. In Section 4, we shall analyze the security of our improved version. Finally, our brief conclusion will be in Section 5.

II Review of Stolen-verifier Attack on OSPA Protocol

The OSPA protocol is composed of two phases, which are the registration phase and the authentication phase. The detailed procedures are described as follows.

Registration phase: Suppose a new user i wants to securely register with the server for accessing services.

1. User i sends a message $\{ID_i, h^2(P \oplus 1)\}$ to the server for registering through a secure channel, where P is a password chosen by user i, N is a nonce and $h(\cdot)$ is a strong one-way hash function.
2. The server stores the registration message $\{ID_i, h^2(P \oplus 1)\}$ and sets $T = 1$ into database after identifying the identity of the user.

Authentication phase: Suppose that user i wants to login the nth time. The authentication phase includes the following steps.

1. User i sends a service request to a server.
2. The server gives a sequential number n to i.
3. User i calculates $\{c_1, c_2, c_3\}$ as a login request and sends it along with ID_i to the server for login, where

$$\begin{aligned} c_1 &= h(P \oplus n) \oplus h^2(P \oplus n), \\ c_2 &= h^2(P \oplus (n+1)) \oplus h(P \oplus n), \\ c_3 &= h^3(P \oplus (n+1)). \end{aligned}$$

4. Upon receiving the login request $\{ID_i, c_1, c_2, c_3\}$ from user i, the server checks if $c_1 \neq c_2$ holds. If it does, the server computes $y_1 = c_1 \oplus h^2(P \oplus n)$ and $y_2 = c_2 \oplus y_1$ by the stored $h^2(P \oplus n)$.
Then, the server replaces $h^2(P \oplus n)$ with y_2 and sets $T = n+1$ for the next login when the equations $h(y_1) = h^2(P \oplus n)$ and $h(y_2) = c_3$ hold.

However, Chen and Ku showed that an attacker could impersonate a legal user to login and obtain the resources by launching the stolen-verifier attack. Suppose an attacker A has stolen the verifier $h^2(P \oplus n)$ after i's $(n-1)$th login and intercepted the login message $\{ID_i, c_1, c_2, c_3\}$ at the nth login. Hence, he/she can derive $h(P \oplus n)$ from $h^2(P \oplus n) \oplus c_1$. Choosing a new password P_A and computing $c_{A_2} = h^2(P_A \oplus (n+1)) \oplus h(P \oplus n)$ and $c_{A_3} = h^3(P_A \oplus (n+1))$ with (ID_i, c_1), the attacker sends it to the server for impersonates i to login. The server executes the fourth step from the authentication phase to identify the login user.

$$\begin{aligned} y_1 &= c_1 \oplus h^2(P \oplus n), \\ y_{A_2} &= c_{A_2} \oplus y_1, \\ c_{A_3} &= h(y_{A_2}) \end{aligned}$$

The attacker will pass the authentication phase, and his/her verifier $h^2(P_A \oplus (n+1))$ will be stored in the server's database. That is, the legal user i will be rejected in the next login process because her/his verifier has been replaced with a new verifier $h^2(P_A \oplus (n+1))$ by the attacker. Therefore, the attacker can use P_A to login and pretend to be user i.

III Our Scheme

In this section, we shall propose a new scheme to withstand the stolen verifier problem. Our scheme is composed of two phases: the registration phase and the authentication phase.

Registration phase:

1. User i calculates $h^2(P \oplus N)$ and sends it along with ID_i to the server through a secure channel, where N is a nonce.
2. The server stores the verifier $h^2(P \oplus N)$ into the database when the identity of the user is identified. Then, it stores $K = h^2(P \oplus N) \oplus h(x||ID_i)$ and N into user i's smart card and releases it to the user i, where x is the secret key of the server. The secret key is maintained by the server.

Our modified registration phase is illustrated in Figure 1.

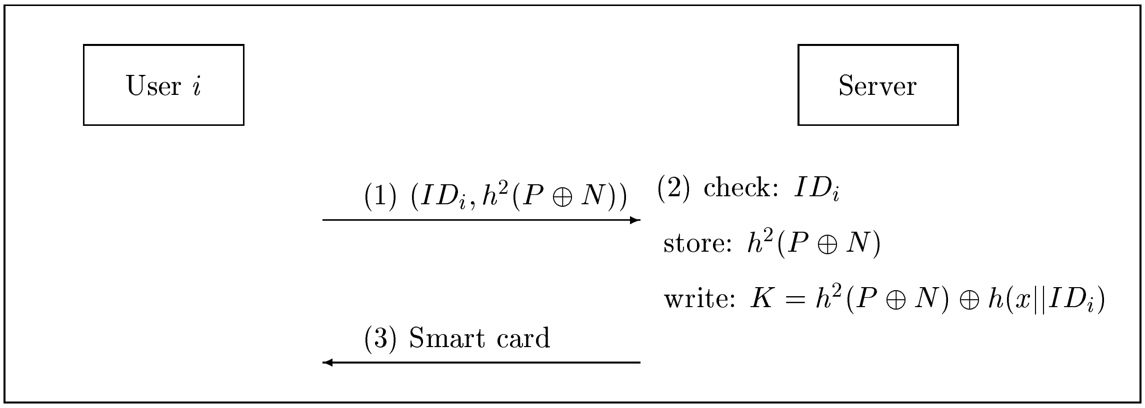

Figure 1: The registration phase of the proposed scheme

Authentication phase: User i inserts his/her smart card into a login device and keys in the password (ID_i, P). The smart card will perform the following operations:

1. Calculate $c_1 = K \oplus h^2(P \oplus N) = h(x||ID_i)$. Here K is stored in the smart card.
2. Calculate $c_2 = c_1 \oplus h(P \oplus N)$.
3. Calculate $c_3 = h(c_1) \oplus h^2(P \oplus N')$, where N' is a new random nonce and $h^2(P \oplus N')$ is a new verifier for the next login.
4. Send $\{ID_i, c_2, c_3\}$ to the server as a login request.

Upon receiving the login request $\{ID_i, c_2, c_3\}$ from the user i, the server performs the following operations to identify the login user:

1. Check the format of ID_i. If it does not pass, then disconnect this connection.
2. Check the identity of the login user by c_2. The server computes $h(x||ID_i)$ and c_2', where $c_2' = h(x||ID_i) \oplus c_2 = h(P \oplus N)$ and checks $h(c_2')$ with the stored verifier $h^2(P \oplus N)$. If it passes, then the server accepts the login request and permits the accessing power.
3. Calculate $h^2(P \oplus N')$ and update $h^2(P \oplus N)$ with $h^2(P \oplus N')$ for the next login.

$$\begin{aligned} h^2(P \oplus N') &= h^2(x||ID_i) \oplus c_3 \\ &= h^2(x||ID_i) \oplus h^2(x||ID_i) \oplus h^2(P \oplus N') \\ &= h^2(P \oplus N') \end{aligned}$$

The modified authentication phase is illustrated in Figure 2.

User i		Server
(1) $c_1 = K \oplus h^2(P \oplus N)$	(2) $\{ID_i, c_2, c_3\}$ $\longrightarrow$	(3) check: ID_i
$c_2 = c_1 \oplus h(P \oplus N)$		compute: $c_2' = c_2 \oplus h(x\|\|ID_i)$
$c_3 = h(P \oplus N) \oplus h^2(P \oplus N')$		check: $h^2(P \oplus N) =?h(c_2')$
		compute: $h^2(P \oplus N') = c_3 \oplus c_2'$
		(4) store: $h^2(P \oplus N')$ into database

Figure 2: The authentication phase of the proposed scheme

IV Analysis

In this section, we shall show that the proposed scheme can withstand some possible attacks as follows.

Guess attack: Assume that an attacker intercepts a login request $\{ID_i, c_2, c_3\}$ over a public network. The attacker cannot derive the password P of the login user from c_2 and c_3. The reason is because the attacker does not know N, N' and x, where x is the secret key of the server.

Replay attack: The user updates the verifier to resist the replay attack in each login session. The next verifier is hidden in the previous session such that c_3 is an implicit verifier $h^2(P \oplus N')$ for the next session. That is, an attacker cannot login the remote server by replaying the previous login request.

Impersonation attack: An attacker A may impersonate ID_i by forging a login request $\{ID_i, c_{A_2}, c_{A_3}\}$ and sending it to the server. The server will execute the authentication phase for identifying the login user as follows:

$$c'_{A_2} = c_{A_2} \oplus h(x||ID_i).$$

However, the forged c_{A_2} cannot pass the test because the verifying equation does not hold $(h^2(P \oplus N) \neq h(c_{A_2}))$. Therefore, an attacker has no chance to login by launching the impersonation attack.

Stolen-verifier attack: Assume that an attacker has stolen a verifier $h^2(P \oplus N)$ and intercepted user i's $(N-1)$th login request $\{ID_i, c_2, c_3\}$ from a server and over the public network. Because he/she cannot derive $h(P \oplus N)$ and $h^2(P \oplus N')$ from c_2 and c_3 by way of the verifier $h^2(P \oplus N)$, respectively, the proposed scheme can resist the stolen-verifier attack since $h(\cdot)$ is a strong one-way hash function.

V Conclusion

In this article, we have proposed an improved version of the OSPA protocol to resist the stolen-verifier attack [2]. We also has shown that the proposed scheme does work.

ACKNOWLEDGEMENTS

Part of this research was supported by the National Science Council, Taiwan, R.O.C., under contract no. NSC91-2213-E-324-003.

References

[1] Chi-Kwong Chan and L. M. Cheng. Cryptanlysis of timestamp-based password authentication scheme. *Computers & Security*, 21(1):74–76, 2002.

[2] Chien-Ming Chen and Wei-Chi Ku. Stolen-verifier attack on two new strong-password authentication protocols. *IEICE Transactions on Communications*, E85-B(11):2519–2521, November 2002.

[3] Hung-Yu Chien, Jinn-Ke Jan, and Yuh-Min Tseng. A modified remote login authentication scheme based on geometric approach. *Journal of Systems and Software*, 55:287–290, 2001.

[4] Min-Shiang Hwang. Cryptanalysis of remote login authentication scheme. *Computer Communications*, 22(8):742–744, 1999.

[5] Min-Shiang Hwang. A remote password authentication scheme based on the digital signature method. *International Journal of Computer Mathematics*, 70:657–666, 1999.

[6] Min-Shiang Hwang, Cheng-Chi Lee, and Yuan-Liang Tang. An improvement of SPLICE/AS in WIDE against guessing attack. *International Journal of Informatica*, 12(2):297–302, 2001.

[7] Min-Shiang Hwang and Li-Hua Li. A new remote user authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, 46(1):28–30, 2000.

[8] Cheng-Chi Lee, Min-Shiang Hwang, and Wei-Pang Yang. A flexible remote user authentication scheme using smart cards. *ACM Operating Systems Review*, 36(3):46–52, 2002.

[9] Cheng-Chi Lee, Li-Hua Li, and Min-Shiang Hwang. A remote user authentication scheme using hash functions. *ACM Operating Systems Review*, 36(4):23–29, 2002.

[10] Li-Hua Li, Iuon-Chung Lin, and Min-Shiang Hwang. A remote password authentication scheme for multi-server architecture using neural networks. *IEEE Transactions on Neural Networks*, 12(6):1498–1504, 2001.

[11] C. L. Lin, H. M. Sun, and T. Hwang. Attacks and solutions on strong-password authentication. *IEICE Transactions on Communications*, E84-B(9):2622–2627, September 2001.

[12] Yuan-Liang Tang Min-Shiang Hwang, Cheng-Chi Lee. A simple remote user authentication scheme. *Mathematical and Computer Modelling*, 36:103–107, 2002.

[13] M. Sandirigama, A. Shimizu, and M. T. Noda. Simple and secure password authentication protocol (sas). *IEICE Transactions on Communications*, E83-B(6):1363–1365, June 2000.

[14] W. H. Yang and S. P. Shieh. Password authentication schemes with smart cards. *Computers & Security*, 18(8):727–733, 1999.