A Hash-based Strong Password Authentication Protocol with User Anonymity

Kumar Mangipudi and Rajendra Katti
(Corresponding author: Kumar Mangipudi)

Department of Electrical and Computer Engineering, 1411 Centennial Blvd,
North Dakota State University, Fargo, ND 58105, USA (Email: {kumar.mangipudi, rajendra.katti}@ndsu.edu)

(Received Aug. 8, 2005; revised and accepted Sep. 21, 2005)

Abstract

Password authentication protocols range from complex public-key cryptosystems to simple hash-based password authentication schemes. One common feature of these protocols is that the user's identity is transmitted in plain during the authentication process, which allows an attacker to monitor the user's activities. In many cases, the user's anonymity is a desirable security feature. In this paper, we propose a hash-based Strong Password Authentication Protocol with user Anonymity (SPAPA). We also analyze the security of our proposed scheme against known attacks.

Keywords: Anonymity, authentication, hash-based, strong password

1 Introduction

User authentication is an important security mechanism. Sometimes a key exchange mechanism is also included in the authentication scheme. More often it is not uncommon to verify the identities of the communicating party before allowing him/her to access a remote server or the system's resources. Until now, a variety of authentication protocols ranging from complex public-key cryptosystems to simple password based authentication schemes have been proposed. These password based authentication schemes can be categorized into two types, one may use a weak password and the other requires strong passwords. A strong password has high entropy and thus cannot be guessed easily. On the other hand a weak password can easily be guessed because of its low entropy and therefore weak-password authentication schemes employ public-key techniques, imposing a heavy computational load on the system. In contrast, the strong password authentication schemes consists only simple operations, i.e., cryptographic hash function and XOR (exclusive-or) operations. Hence, the strong password authentication schemes are more suitable for constrained environments (smart card applications) because of simple design, light computational overhead and easy implementation.

One common feature of the hash-based authentication protocols (referred in the next section) is that the user's identity is transmitted in plain during the authentication process, which allows an attacker to monitor the user activities. In many cases, it is of utmost importance to provide anonymity so that the adversary cannot trace user activity. In this paper, we propose a hash-based Strong Password Authentication Protocol with user Anonymity (SPAPA). The strong password, which has high entropy, can also be referred to as a "key". The rest of the paper is organized as follows. We review the related work in Section 2. Our method (described in Section 4) is an improvement of Lin et al.'s [8] protocol, which is described in Section 3. We analyze the security of our proposed protocol in Section 5 followed by conclusions in Section 6.

2 Related Work

Lamport's protocol is the first hash-based password authentication protocol that was proposed almost two decades ago [6]. In this scheme, it is not possible to impersonate the user either by eavesdropping on the authentication message or by reading the database in the server. But, the application of the Lamport's scheme is questionable because of its high hash overhead and the necessity to reset the password in addition to being vulnerable to the reply attack. Later Haller [3] proposed, the S/KEY, which is a deployable version of the Lamport's scheme, but it is also vulnerable to the reply attack as shown by Mitchell [9]. Gong [2] proposed an optimal authentication protocols that is resistant to password guessing attacks. CINON [11] and PERM [12] are the one time password schemes proposed by Shimizu and Shimizu et al., respectively. While the former is very inconvenient to the user as he has to memorize two random numbers, the later succumbs to the man-in-the-middle attack and the impersonation attack. Again, Sandirigama et al. [10] proposed a simple strong-password authentication protocol, SAS, which was designed to be superior

Registration Phase:			
Step	User		Server
R1	$A, h^2(P \oplus N)$	$\rightarrow$	
R2		$\leftarrow$	$K = h^2(P \oplus N) \oplus h(x\|\|A)$

Figure 1: Regisration phase of LSH-OSPA

to other schemes, in storage utilization, processing time, and transmission overhead. Later, Lin et al. [7] found SAS to be vulnerable to the replay attack and the denial-of-service attack, as such proposed the OSPA (Optimal Strong Password Authentication) protocol. The OSPA is a refined scheme that was claimed to be secure against the stolen-verifier attack, the replay attack and the denial-of-service attack. Unfortunately, Chen and Ku [1] presented the stolen-verifier attack on OSPA. Further to this, Tsuji and Shimizu [13] showed a man-in-the-middle attack on OSPA. Again the OSPA protocol was refined in [8] so that it resists the guessing attack, the replay attack, the impersonation attack, and the stolen-verifier attack. This was later crypt analyzed by Ku et al. [4], only to show the methods to mount two well known attacks, the denial-of-service and the replay attack. Recently, Ku [5] proposed a hash-based strong password authentication protocol without using smart cards. Unfortunately, the author did not consider incorporating the user anonymity.

3 Lin-Shen-Hwang's Protocol

In this section, we describe the Lin-Shen-Hwang's Protocol OSPA [8] (LSH-OSPA) and the attacks presented by Ku et al. on this protocol. For the rest of our discussion we shall use the following nomenclature. U, S and E denote the user, server and adversary, respectively. A is the user's identity and P is a high entropy strong password or the user's key. x is the server's secret key. N and N' represent two different one time generated random numbers (nonce). $h(m)$ and $h_{\text{key}}(m)$ are a one way hash and keyed hash functions on message m, respectively. The keyed hash function can also be referred to as a Message Authentication Code (MAC). $||$ and $\oplus$ are the concatenation and bitwise exclusive XOR operations, respectively.

3.1 Lin-Shen-Hwang's Protocol

LSH-OSPA protocol comprises of two phases: the registration phase and the authentication phase. The registration phase is invoked once for registering the user using a secure channel while the authentication phase is executed every time the user logs-in the system or the authentication server via a common (insecure) channel. Figures 1 and 2 depict the registration phase and the authentication phase of LSH-OSPA, respectively, whose details are given below:

In Step R1 of the Registration phase (Figure 1), U generates N; calculates $h^2(P \oplus N)$ and sends it along with his/her identity to the server S. Then, S stores the verifier $h^2(P \oplus N)$ and the user's identity in its database. In Step R2, S issues a smart card storing $K = h^2(P \oplus N) \oplus h(x||A)$ to U. Finally, the user's smart card stores K and N. A user can re-register with the server as and when it is required.

In Step A1 of the authentication phase (Figure 2), U inserts his/her smart card into a login device and keys in the password P, and then the smart card computes $c_1 = K \oplus h^2(P \oplus N) = h(x||A), c_2 = c_1 \oplus h(P \oplus N), c_3 = h(c_1) \oplus h^2(P \oplus N')$, where N' is a new nonce generated by U. Next, U sends A, c_2, c_3 to S in Step A2. After receiving U's login request, S first computes $h(x||A)$ and then uses $h(x||A)$ and the received c_2 to compute v i.e., $v = h(x||A) \oplus c_2 = h(P \oplus N)$. If $h(v)$ equals the stored verifier $h^2(P \oplus N)$, S approves U's login request and computes $h^2(P \oplus N') = h^2(x||A) \oplus c_3$. Finally, S updates the verifier $h^2(P \oplus N)$ with $h^2(P \oplus N')$ for U's next login.

3.2 Weakness of Lin-Shen-Hwang's Protocol

Ku et al. [4] showed the weakness of LSH-OSPA protocol by presenting a Denial-of-Service (DoS) attack and a replay attack. We show only the DoS attack. For complete details refer to Ku et al. [4]. During the Step A2 of LSH-OSPA, an adversary E replaces the transmission of c_3 with the transmission of an equal-sized random number, denoted by R while the transmission of A and c_2 are left unchanged. Upon receiving the modified message, S computes $v = h(x||A) \oplus c_2 = h(P \oplus N)$. Since $h(v)$ equals the stored verifier $h^2(P \oplus N)$, S approves A's login request and computes $h^2(x||A) \oplus R$. Then, S updates the verifier $h^2(P \oplus N)$ with $h^2(x||A) \oplus R$ for U's next login. Although S can successfully verify the authenticity of the user U in this session, the user's subsequent login requests will be denied unless he re-registers with S again. Thus, a DoS attack as E can easily lock the account of any user.

4 Strong Password Authentication Protocol with User Anonymity (SPAPA)

In this section, we propose SPAPA that circumvents the above presented attacks. Our protocol achieves user anonymity by making the user transmit a temporary identity $h(A||N||P)$ instead of his/her true identity, A during authentication phase. Similar to LSH-OSPA, the SPAPA has two phases, details of which are shown in Figure 3 (the registration phase, which is executed through a secure channel) and Figure 4 (the authentication phase that is executed in real time in a hostile environment).

In Step R1 of the registration phase (Figure 3), U generates N and sends his/her identity A and the computed values of $h^2(P \oplus N)$, $h(A||N||P)$ to S. Upon receiving the message from U, in Step R2, S calculates

Authentication phase:			
Step	User		Server
A1	Computes c_1, c_2, c_3		
A2	Sends $\{A, c_2, c_3\}$	$\rightarrow$	Computes v and verifies $h(v)$

Figure 2: Authentication phase of LSH-OSPA

Registration Phase in (SPAPA):			
Step	User		Server
R1	$A, h^2(P \oplus N), h(A\|\|N\|\|P)$	$\rightarrow$	
R2		$\leftarrow$	$K = h^2(P \oplus N) \oplus h(x\|\|A)$

Figure 3: Registration phase of SPAPA via a secure channel

$K = h^2(P \oplus N) \oplus h(x||A)$; sends K; and stores A, the temporary identity (TID) $h(A||N||P)$ and the verifier $h^2(P \oplus N)$. Finally, U's smart card stores K and N.

In Step A1 of the authentication phase (Figure 4), U keys in the password P. Then his/her smart card generates a new nonce N'; calculates c_1 and the current (TID) $h(A||N||P)$ using the stored values of K and N; calculates the next (TID') $= h(A||N'||P), c_2, c_3, c_4, c_5$; c_1 through c_5 are computed as:

$$\begin{aligned}
c_1 &= K \oplus h^2(P \oplus N) = h(x||A), \\
c_2 &= h(c_1) \oplus h(P \oplus N), \\
c_3 &= h(c_1) \oplus h(A||N'||P), \\
c_4 &= h(c_1) \oplus h_2(P \oplus N'), \\
c_5 &= h_{\text{KEY}}(h^2(P \oplus N')||h(A||N'||P)||h(P \oplus N))
\end{aligned}$$

and stores N' instead of N. Here, the $h(x||A)$, a shared secret between the user and server serves as the key i.e., KEY=$h(x||A)$ for computing c_5.

Next, U sends $\{h(A||N||P), c_2, c_3, c_4, c_5\}$ to S in Step A2. Note that the transmitted or received information is enclosed in curly braces. Upon receiving the message from the user, S calculates $h(x||A)$ based on $h(A||N||P)$ from its database; computes $v = h^2(x||A) \oplus c_2 = h(P \oplus N)$; checks if $h(v) = h^2(P \oplus N)$ (stored verifier); then it performs a bit-wise XOR using $h^2(x||A)$ on c_3 and c_4 to retrieve $h(A||N'||P)$ and $h^2(P \oplus N')$, respectively. The server now checks the integrity of the received $h(A||N'||P)$, $h^2(P \oplus N')$, and $h(P \oplus N)$ using the keyed hash function before granting access to the user and finally stores $h(A||N'||P)$ and $h^2(P \oplus N')$ for the user's next login.

5 Security Analysis

In this section, we analyze the security of the SPAPA protocol. Assuming that the hash function, $h(m)$ is a collision-free, Strong, one-way, Hash-Function (SHF), P is a strong password and the server's secret key, x is under strict protection, our proposed SPAPA is resistant to the following known attacks and solves the two unresolved problems mentioned by Chen and Ku [1]. (i) The SHF does not allow an adversary to impersonate the user login even though he has stolen the user's verifier. (ii) $c_5 = h_{\text{KEY}}(h^2(P \oplus N')||h(A||N'||P)||h(P \oplus N))$ protects the integrity of the transmitted message. In practice, compromising x, results in compromising the entire system, which requires re-initialization of the system where in all the users re-register with the server.

5.1 User Anonymity

An adversary cannot identify the person who is trying to login, as his identity A, the random number N and his password P are hashed together to generate the TID, whose value varies for every login because of a different $N(N \neq N')$ i.e., $h(A||N||P) \neq h(A||N'||P)$.

5.2 Stolen Smart Card or Online Guessing Attack

Since the password P is a strong password and the server only allows a definite number of login attempts, an adversary E cannot impersonate the user by stealing his smart card that stores K and the previous nonce N.

5.3 Offline Guessing Attack

An adversary who intercepts a login request $\{h(A||N||P), c_2, c_3, c_4, c_5\}$ over a public network cannot derive the password P from the message because of SHF.

5.4 Stolen-Verifier Attack

Suppose an adversary has stolen the verifier $h^2(P \oplus N)$ and intercepted user i's N^{th} login request $\{h(A_i||N_i||P), c_2, c_3, c_4, c_5\}$ over the public network. He/she cannot identify the user and at the same time cannot derive P, $h(P \oplus N)$ and $h^2(P \oplus N')$ from c_2, c_4 and the verifier $h^2(P \oplus N)$, because h is a strong one way hash function.

Authentication phase in (SPAPA):			
Step	User		Server
A1	Computes $h(A\|\|N\|\|P), h(A\|\|N'\|\|P)$, c_1, c_2, c_3, c_4, c_5.		
A2	Sends $\{h(A\|\|N\|\|P), c_2, c_3, c_4, c_5\}$	$\rightarrow$	Computes v, verifies $h(v)$, and stores $h(A\|\|N'\|\|P)$ and $h^2(P \oplus N')$

Figure 4: Authentication phase of SPAPA

5.5 Replay Attack

A reply attack is not possible with SPAPA due the following two reasons. First, the TID does not provide an adversary enough information to verify that two successive log-in requests belong to the same user. Second, consider that the adversary was wise enough to eavesdrop on $(n-2)^{\text{th}}$ and $(n-1)^{\text{th}}$ login requests $\{h(A||N^{(\mathrm{n}-3)}||P), c_2^{(\mathrm{n}-2)}, c_3^{(\mathrm{n}-2)}, c_4^{(\mathrm{n}-2)}, c_5^{(\mathrm{n}-2)}\}$ and $\{h(A\ ||N^{(\mathrm{n}-2)}||P), c_2^{(\mathrm{n}-1)}, c_3^{(\mathrm{n}-1)}, c_4^{(\mathrm{n}-1)}, c_5^{(\mathrm{n}-1)}\}$, respectively before A's n$^{\text{th}}$ login. If he tries to launch a reply attack and hence an impersonation attack similar that was described in [4] by replacing whole or a part of A's n^{th} login request, $\{h(A||N^{(n-1)}||P), c_2^{(\mathrm{n})}, c_3^{(\mathrm{n})}, c_4^{(\mathrm{n})}, c_5^{(\mathrm{n}-2)}\}$ with the eavesdropped messages then the server denies the current login request. Two such cases are detailed below. Note bold letters indicate the replaced values.

Case 1:
$\{h(A||N^{(n-1)}||P), c_2^{(\mathrm{n})}, \mathbf{c_3^{(n-2)}}, \mathbf{c_4^{(n-2)}}, \mathbf{c_5^{(n-2)}}\}$, where $\mathbf{c_5}^{\mathbf{(n-2)}} = h_{\mathrm{KEY}}(h^2(P \oplus N^{(\mathrm{n}-2)})||h(A||N(n-2)||P)||h\ (P \oplus N^{(\mathrm{n}-3)}))$. Then the server calculates $v = h(P \oplus N^{(\mathrm{n}-1)})$ from $c_2^{(\mathrm{n})}$; compares $h(v)$ with the stored verifier and calculates $c_5 = h_{\mathrm{KEY}}(h^2(P \oplus N^{(\mathrm{n}-2)})||h(A||N^{(\mathrm{n}-2)}||P)||h(P \oplus N^{(\mathrm{n}-1)}))$. Since the transmitted $\mathbf{c_5^{(n-2)}}$ is not equal to the calculated c_5, the server denies the current login request.

Case 2:
$\{h(A||N^{(n-1)}||P),\ \mathbf{c_2^{(n-2)}}, \mathbf{c_3^{(n-2)}}, \mathbf{c_4^{(n-2)}}, \mathbf{c_5^{(n-2)}}\}$, the server denies the request though the transmitted $\mathbf{c_5^{(n-2)}}$ and calculated c_5 are equal to $h_{\mathrm{KEY}}(h^2(P \oplus N^{(\mathrm{n}-2)})||h(A||N^{(\mathrm{n}-2)}||P)||h(P \oplus N^{(\mathrm{n}-3)}))$, but the transmitted verifier $\mathbf{v = h(P \oplus N^{(n-3)})}$ calculated form $\mathbf{c_2^{(n-2)}} \oplus h^2(x||A))$ is not equal to the stored verifier $h(v = h(P \oplus N^{(\mathrm{n}-1)}))$, which was updated during A's (n-1)$^{\text{th}}$ login request.

5.6 DoS Attack

In a DoS attack, the adversary uses some ways so that the server denies the subsequent access requests of the legitimate user (refer to the DOS attack in Section 3). Consider that the adversary corrupts c_3, c_4 which contain $h(A||N'||P)$ and $h^2(P \oplus N')$ by intercepting the transmitted message $\{h(A||N||P), c_2, c_3, c_4, c_5\}$ and replacing it with another set of numbers such as $\{h(A||N||P), c_2, c_{A3}, c_{A4}, c_5\}$. Then the server returns a failure on the integrity check as shown below. Let D$_1$, D$_2$, D$_{A1}$ and D$_{A2}$ are the results of the following computations, $\mathrm{D}_1 = h(A||N'||P) = c_3 \oplus h(c_1)$, $\mathrm{D}_2 = h^2(P \oplus N') = c_4 \oplus h(c_1)$, $\mathrm{D}_{A1} = c_{A3} \oplus h(c_1)$ and $\mathrm{D}_{A1} = c_{A4} \oplus h(c_1)$, respectively and $v = h^2(x||A) \oplus c_2 = h(P \oplus N)$. Then $c_5 = h_{\mathrm{KEY}}(\mathrm{D}_2||\mathrm{D}_1||v)h_{\mathrm{KEY}}(\mathrm{D}_{A2}||\mathrm{D}_{A1}||v)$ because D$_1$ and D$_2$ differ from D$_{A1}$ and D$_{A2}$, respectively and hence the server denies the adversary's attempt to the user account.

6 Conclusion

In this paper, we proposed a hash-based strong password authentication protocol with user anonymity (SPAPA). The user's anonymity is highly required in a hostile environment as it prevents observing the user's activity. In addition, the SPAPA protocol is very simple and contains only hash functions and XOR operations, which are suitable for power and computation constrained smart card applications.

References

[1] M. Chen, and W. Ku, "Stolen-verifier attack on two new strong-password authentication protocols," *IEICE Transactions on Communications*, vol. E85-B, no. 11, pp. 2519–2521, 2002.

[2] L. Gong, "Optimal authentication protocols resistant to password guessing attacks," in *Proceedings of the 8th IEEE Computer Security Foundation Workshop*, pp. 24–29, 1995.

[3] N. M. Hailer, "The S/KEY (TM) one-time password system," in *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, pp. 151–158, 1994.

[4] W. C. Ku, H. C. Tsai, and S. M. Chen, "Two simple attacks on Lin-Shen-Hwang's strong password authentication protocol," *ACM Operating Systems Review*, vol. 37, no. 4, pp. 26–31, 2003.

[5] W. C. Ku, "A hash-based strong-password authentication scheme without using smart cards," *ACM Operating Systems Review*, vol. 38, no. 1, pp. 29–34, 2004.

[6] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[7] L. Lin, H. M. Sun, and T. Hwang, "Attacks and solutions on strong-password authentication," *IEICE Transactions on Communications*, vol. E84-b, no. 9, pp. 2622–2627, 2001.

[8] W. Lin, J. J. Shen, and M. S. Hwang, "Security enhancement for optimal strong password authentication protocol," *ACM Operating Systems Review*, vol. 37, issue 2, pp. 7–12, 2003.

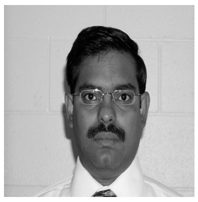
[9] C. J. Mitchell and L. Chen, "Comments on the S/KEY user authentication scheme," *ACM Operating Systems Review*, vol. 30, no. 4, pp. 12–16, 1996.

[10] M. Sandirigama, A. Shimizu, and M. T. Noda, "Simple and secure password authentication protocol (SAS)," *IEICE Transactions on Communications*, vol. E83-B, no. 6, pp. 1363–1365, 2000.

[11] A. Shimizu, "A dynamic password authentication method by one-way function," *IEICE Transactions on Fundamentals*, vol. J73-D-I, no. 7, pp. 630–636, 1990.

[12] A. Shimizu, T. Horioka, and H. Inagaki, "A password authentication methods for contents communication on the Internet," *IEICE Transactions on Communications*, vol. ESI-B, no. 8, pp. 1666–1673, 1998.

[13] T. Tsuji and A. Shimizu, "An impersonation attack on one-time password authentication protocol OSPA," *IEICE Transactions on Communications*, vol. E86-B, no. 7, pp. 2182–2185, 2003.



Kumar Mangipudi received his Diploma in Electrical and Electronics Engineering in 1994 from State Board of Technical Education, AP, India and his AMIE (B.S) in Electrical Engineering from the Institution of Engineers (India) in 2000. While pursuing his AMIE, he worked on as an Instrumentation Engineer executing the design, testing, and commissioning of instrumentation control systems for process industry at various places in India and Dubai, UAE. In 2002 he obtained his M.S in Electrical Engineering form South Dakota School of Mines & Technology. He is currently a PhD candidate in the department of Electrical and Computer Engineering at North Dakota State University. His interests are in Network Security, Cryptography, VLSI, and Embedded Systems.

Rajendra Katti received his B. Tech degree from the Indian Institute of Technology (Bombay), India in 1983. He received his M.S. in Mechanical Engineering from the University of Idaho in 1985, his M.S. in Electrical Engineering from Washington State University in 1987, where he also earned his PhD in Electrical Engineering in 1991. Dr. Katti teaches courses related to Digital Systems and Computer Architecture. Dr. Katti has received funding from the National Science Foundation in the area of Performance Modeling of Computer Architectures and Cryptography. His interests are in Cryptographic Hardware, Finite Field arithmetic, Fault Tolerant computing and Computer Architecture. He has published over 40 journal and conference papers on these topics. He was a senior design engineer at the Intel Corporation in 2000 and 2001 where he worked in the Design for Testability Group. He has also taught at the Wichita State University in Kansas. He has collaborated with the IBM Almaden Research Center for the development of unidirectional error correcting codes. He is currently an Associate Professor in the Department of Electrical and Computer Engineering at North Dakota State University.