Security Enhancement for Two-Gene-Relation Password Authentication Protocol (2GR)

Chun-Li Lin Ching-Po Hung
Dept. of Computer Science and Information Engineering, Shu-Te University
118760@mail.csc.com.tw

Abstract

In 2004, Tsuji and Shimizu proposed a one-time password authentication protocol, named 2GR. By the 2GR protocol, an attacker who has stolen the verifiers from the server cannot impersonate a valid user. However, Lin and Hung described that it is vulnerable to an impersonation attack, in which any attacker can, without stealing the verifiers, masquerade as a legitimate user. In this paper, we shall propose an improved version of the 2GR protocol to overcome this impersonation attack.

Keywords: network security, user authentication, one-time password.

摘要

於西元 2004 年，Tsuji 與 Shimizu 提出一種簡稱為 2GR 的一次通行碼認證機制。根據 2GR 機制，已從伺服器竊得驗證碼的攻擊者，她/他無法偽冒使用者。然而，林與洪描述：2GR 機制仍然無法防禦一種偽裝攻擊，此類攻擊是攻擊者不必偷竊驗證碼即可偽裝成為一合法的使用者。本論文將提出一種 2GR 機制的改善版來克服這種偽裝攻擊。

關鍵詞：網路安全，使用者認證，一次通行碼。

1. Introduction

Password authentication schemes allow a valid user to login a remote server and to access the services provided by the remote server over an insecure channel. Due to the convenience and low cost of password authentication techniques, they are widely used in many network applications. However, they often suffer from eavesdropping, replaying, guessing or stealing attacks. One of the solutions to this problem is one-time password authentication, in which every password is used only once. Once a one-time password is used, it will be no longer valid even if it is eavesdropped, replayed, guessed or stolen. Usually, a one-time password authentication method employs a user-side token for storing the high-entropy seed and generating dynamic passwords.

Since Lamport [7] brought up the first one-time password authentication scheme, there have been many subsequent improvements, including CINON [1], S/KEY [9], PERM [2], SAS [8], OSPA [4], SAS-2 [12], ROSI [6]. Among these schemes, S/KEY does not provide protection against active attacks [10]. Both CINON and PERM is vulnerable to a kind of 'Man in the Middle' attack [8]. SAS, OSPA and SAS-2 suffer from the stolen-verifier attacks [5], [11]. ROSI suffers from the theft attack [11]. In 2004, Tsuji and Shimizu proposed a one-time password authentication protocol, named 2GR. By the 2GR protocol, an attacker who has stolen the verifiers from the server cannot impersonate a valid user. However, in 2006, Lin and Hung described that it is vulnerable to an impersonation attack [3], in which any attacker can, without stealing the verifiers, masquerade as a legitimate user. In this paper, we shall propose an improved version of the 2GR protocol to overcome this impersonation attack.

2. Review of the 2GR Protocol

The 2GR scheme is divided into two phases: the registration phase and the authentication phase. We first list notations used throughout this paper as follows:

U	the user
S	the authentication server
A	the attacker
ID	the user's identity
P	the user's password
$h(\cdot)$	a secure one-way hash function
N_i	a large-enough random number for the ith authentication session
R	a large-enough random number
L	the maximum allowable number of log-in attempts

2.1 Registration Phase

Figure 1 shows the registration phase of the 2GR scheme.

(1) U enters ID and P. Then U generates N_0, N_1, N_2 and stores N_1, N_2. Next, U calculates G_0, G_1, G_2 and D_1, D_2, where

$$G_0 = h(ID,P,N_0),$$

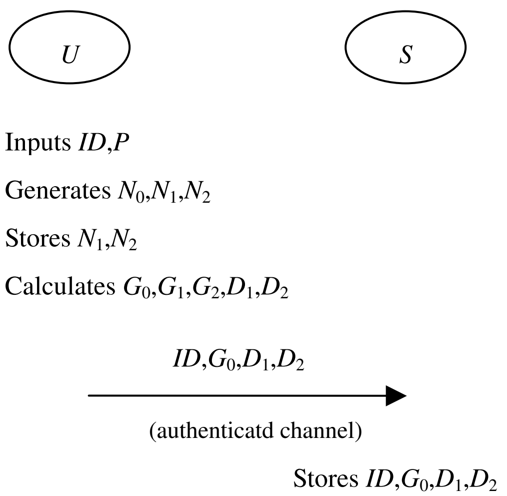

Fig. 1 Registration phase of 2GR

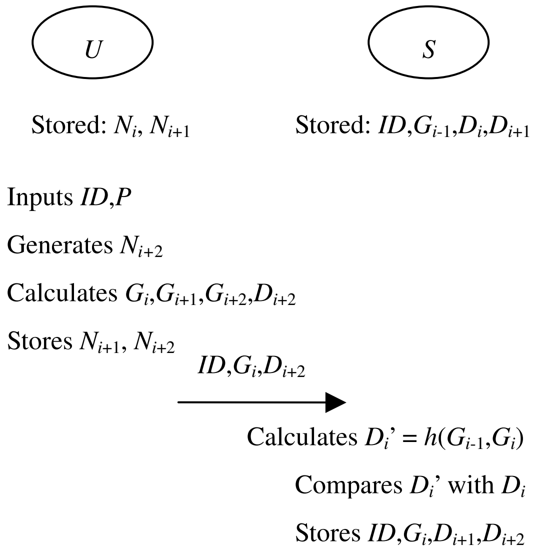

Fig. 2 The *i*th authentication phase of 2GR

$$G_1 = h(ID,P,N_1),$$
$$G_2 = h(ID,P,N_2),$$
$$D_1 = h(G_0,G_1),$$
$$D_2 = h(G_1,G_2).$$

(**2**) U sends ID, G_0, D_1, D_2 to S through an authenticated channel.

(**3**) S stores the received data (ID,G_0,D_1,D_2) for subsequent authentication.

2.2 Authentication Phase

When U wants to login the system, U executes the ith authentication session of the 2GR protocol. Before the beginning of the ith authentication session, U is with (N_i, N_{i+1}) and S is with (ID, G_{i-1}, D_i, D_{i+1}). Figure 2 shows the ith authentication phase of the 2GR scheme.

(**1**) U enters ID and P. Then U generates N_{i+2} and calculates G_i, G_{i+1}, G_{i+2} and D_{i+2}, where

$$G_i = h(ID,P,N_i),$$
$$G_{i+1} = h(ID,P,N_{i+1}),$$
$$G_{i+2} = h(ID,P,N_{i+2}),$$
$$D_{i+2} = h(G_{i+1},G_{i+2}).$$

Next, U stores (N_{i+1}, N_{i+2}) instead of (N_i, N_{i+1}).

(**2**) U sends ID, G_i and D_{i+2} to S.

(**3**) After receiving the data from the user, S calculates $D_i' = h(G_{i-1},G_i)$ using the stored G_{i-1} and the received G_i. Then S compares D_i' with the stored D_i. If they are equal, U is authenticated and S stores (ID, G_i, D_{i+1}, D_{i+2}) in place of (ID, G_{i-1}, D_i, D_{i+1}).

3. Review of the Impersonation Attack on the 2GR Protocol

The impersonation attack consists of three phases: (i) *server spoofing phase*, (ii) *verifier modifying phase*, and (iii) *impersonating phase*. We describe these three phases as follows. Figure 3 shows the flow of the impersonation attack.

3.1 Server Spoofing Phase

In the ith authentication session, U sends ID, G_i, and D_{i+2} to S. The attacker can receive the transmitted data, and accept this logon connection. Later, the attacker can break this accepted connection and put the user under an illusion of network or system errors. In the user's point of view, the ith authentication is accomplished but the service is interrupted. Now, the user is with (N_{i+1}, N_{i+2}) while the server is still with (ID, G_{i-1}, D_i, D_{i+1}).

3.2 Verifier Modifying Phase

In the (i+1)th authentication session, when U sends ID, G_{i+1}, D_{i+3}, to S, the attack intercepts the transmitted data. The attacker records G_{i+1} and then forwards the server ID, G_i, and D_{i+2}', in which the attacker chooses a random number G_{i+2}' and calculates $D_{i+2}' = h(G_{i+1},G_{i+2}')$. After receiving the data from the attacker, S calculates $D_i' = h(G_{i-1},G_i)$ using the stored

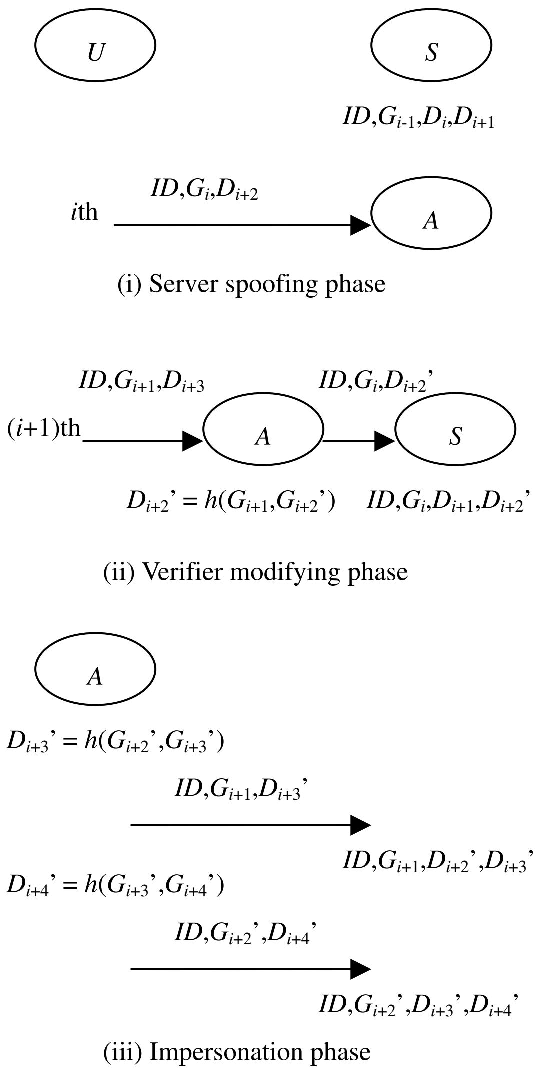

Fig. 3 Impersonation attack on 2GR

G_{i-1} and the received G_i. Then S compares D_i' with the stored D_i. The server will pass the authentication check and update user's verifier as (ID, G_i, D_{i+1}, D_{i+2}'). In the user's point of view, the (i+1)th authentication is accomplished and the service is supplied.

3.3 Impersonating Phase

The attacker will hereafter impersonate U to login without the user's participation until the user detects it. The attacker chooses a random number G_{i+3}' and calculates $D_{i+3}' = h(G_{i+2}',G_{i+3}')$. The attacker sends ID, G_{i+1}, and D_{i+3}' to S. The server will pass the authentication check and update user's verifier as (ID, G_{i+1},

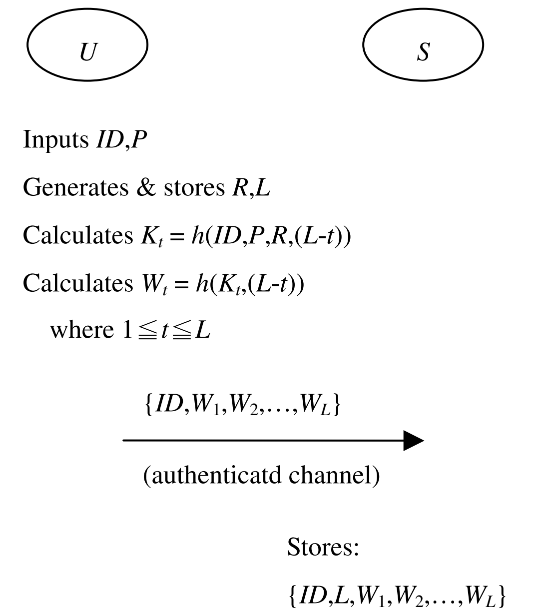

Fig. 4 Registration phase of the improved 2GR

D_{i+2}', D_{i+3}').

Then, the attacker chooses a random number G_{i+4}' and calculates $D_{i+4}' = h(G_{i+3}', G_{i+4}')$. *A* send *ID*, G_{i+2}', D_{i+4}' to the server. *S* will be cheated to update the user's verifier as (*ID*, G_{i+2}', D_{i+3}', D_{i+4}'). Note that in this session, *A* can hereafter impersonate the user without using any stealings.

4. The Improved 2GR Protocol

There are two phases in the improved 2GR protocol: registration and authentication.

4.1 Registration Phase

Figure 4 shows the registration phase of the improved 2GR protocol.

(1) *U* inputs *ID* and *P*. Then *U* generates *R,L*. Next, *U* calculates K_t and W_t , where $1 \leqq t \leqq L$,

$$K_t = h(ID,P,R,(L-t)),$$
$$W_t = h(K_t, (L-t)).$$

(**2**) *U* sends {*ID*, W_1, W_2, …, W_L} to *S* through an

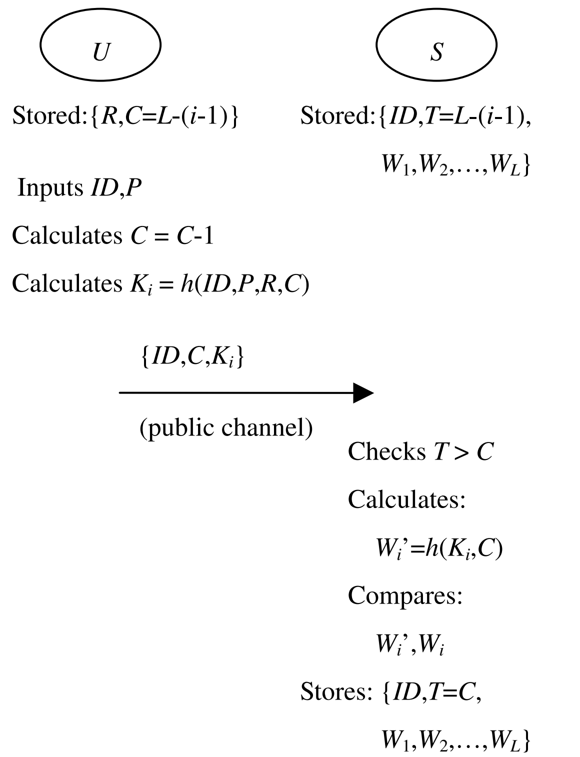

Fig. 5 The *i*th authentication phase of the improved 2GR

authenticated channel.

(**3**) *S* stores {*ID*, *L*, W_1, W_2, …, W_L } for subsequent authentication. Meanwhile, *U* stores {*R*, *L*}.

4.2 Authentication Phase

For the *i*th login, *U* must be with {*R*, *C*=*L*-(*i*-1)} and *S* may be with {*ID*, *T*=*L*-(*i*-1), W_1, W_2, …, W_L}. Figure 5 shows the *i*th authentication phase of the improved 2GR protocol.

(**1**) *U* inputs *ID* and *P*. Then *U* performs *C* = *C* -1 and then calculates $K_i = h(ID,P,R,C)$.

(**2**) *U* sends {*ID*, *C*, K_i} to *S* through a public channel.

(**3**) On receiving {*ID*, *C*, K_i}, *S* compares *C* with the stored *T*. If *T* > *C*, then *S* calculates $W_i' = h(K_i, C)$, compares the W_i' with the stored W_i. If the two values are equal, the authenticity of *U* is confirmed. *S* replaces *T* (=*L*-(*i*-1)) with *C* (=*L*-*i*).

5. Security and Performance Analysis

The improved 2GR scheme withstands some possible attacks and is the most efficient of the one-time password schemes.

5.1 Security Considerations

(1) Replaying attack
The improved 2GR uses the counter to resist the replay attack. An attacker cannot login the remote server by replacing the previous login message which has passed the authentication check.

(2) Guessing attack
Assume that an attacker intercepts a login request $\{ID, C, K_i\}$ over a public network. It is infeasible to guess P from $K_i = h(ID,P,R,C)$ without knowing the random number R.

(3) Denying attack
If a server is cheated by an attacker to update the false verifier for the subsequent authentication, the legal user will not login successfully anymore. To resist the attack, the improved 2GR protocol does not update the verifier (W_1, W_2, …, W_L) directly.

(4) Impersonating attack [3]
By the 2GR protocol, the server updates the new verifier without any integrity check, that is, D_{i+2} in the ith authentication session. This leads the attacker can forge the verifier for later authentication. Hence, Lin and Hung can illustrate a generalized impersonation attack which consists of three phases: (i) *server spoofing phase*, (ii) *verifier modifying phase*, and (iii) *impersonating phase*.

For the improved 2GR protocol does not update the verifier (W_1, W_2, …, W_L) directly and checks the integrity by comparing the calculated $h(K_i,C)$ with the stored W_i, the *verifier modifying phase* cannot work on the improved 2GR protocol. Hence, the improved 2GR protocol can withstand this impersonation attack.

(5) Stealing attack
If an attacker steals the verifier W_t $(=h(K_t,(L-t))$ and taps the tth login request ID, C, and K_t $(=h(ID,P,R,C)$, he still has no way to impersonate a legal user.

(6) Man in the middle attack
If an attacker receives sets of data from two consecutive sessions, he is able to forge the login request to cheat the server successfully. From the next authentication session onwards the attacker can freely login impersonating the real user.

In the improved 2GR protocol, the ith login request $\{ID, C, K_i = h(ID,P,R,C)\}$ and the verifier $W_i = h(K_i,C)$ is assigned in advance. Although the attacker intercepts $\{ID, C=(L-i), K_i\}$ and $\{ID, C=L-(i+1), K_{i+1}\}$, he still cannot forge the subsequent login request to pass the authentication.

5.2 Performance Considerations

It is evaluated that 2GR is the most efficient of the one-time password methods [11]. In this section we only describe the performance of the improved 2GR is far superior to 2GR's. Table 1 summarizes performance ratings of 2GR and improved 2GR in the ith authentication phase.

The improved 2GR has to store two (R, C) and L+2 (ID, T, W_1, W_2, …, W_L) data in data storages which is more than 2GR has to do. However, this is not an issue because security, processing speed and load are more important than storage capacity. In the authentication phase the total of hashing, generating random number and comparing operations performed by the user and server, the improved 2GR is 4 while 2GR is 7. Especially, at user side, the improved 2GR performs 1 hashing operation is far superior to the 2GR does 4 operations.

Those operations for performing the counter *C* decreased by 1 in the authentication phase and calculating all the verifiers in the registration phase can be computed beforehnad. Hence, we ignored to count those operations.

Table 1 Performanec ratings

Item	2GR		improved 2GR	
	U	*S*	*U*	*S*
Data storge	2	4	2	*L*+2
Hashing	4	1	1	1
Generating random no.	1	0	0	0
Comparing	0	1	0	2

6. Conclusions

In this paper, we described the improved 2GR protocol to eliminate the impersonation attack. For applying the improved 2GR method, we will cope with the mutual authentication to prove the server.

References

[1] A. Shimizu, "A Dynamic Password Authentication Method by One-Way Function," System and Computers in Japan, vol.22, no.7, pp.32-40, July 1991.

[2] A. Shimizu, T. Horioka, and H. Inagaki, "A Password Authentication Method for Contents Communication on The Internet," IEICE Trans. Commun., vol.E81-B, no.8, pp.1666-1673, Aug. 1998.

[3] C.L. Lin and C.P. Hung, "Impersonation Attack on Two-Gene-Relation Password Authentication Protocol (2GR)," IEICE Trans. Commun., vol.E89-B, no.12, pp.3425-3427, Dec. 2006.

[4] C.L. Lin, H.M. Sun, and T. Hwang, "Attacks and Solutions on Strong-Password Authentication," IEICE Trans. Commun., volvol.E84-B, no.9, pp. 2622-2627, Sept. 2001.

[5] C.M. Chen and W.C. Ku, "Stolen-Verifier Attack on Two New Strong-Password Authentication Protocols," IEICE Trans. Commun., vol.E85-B, no.11, pp.2519-2521, Nov. 2002.

[6] H.-Y. Chien and J.-K. Jan, "Robust and Simple Authentication Protocol," Comput. J., vol.46, no.2, pp.193-201, Feb. 2003.

[7] L. Lamport, "Password Authentication with Insec-cure Communication," Commun. ACM, vol.24, no. 11, pp.770-772, Nov. 1981.

[8] M. Sandirigama, A. Shimizu, and M.T. Noda, "Simple and Secure Password Authentication Protocol (SAS)," IEICE Trans. Commun., vol.E83-B, no.6, pp.1363-1365, June 2000.

[9] N.M. Haller, "The S/KEY (TM) One-Time Password System," Proc. Internet Society Symposium on Network and Distributed System Security, pp. 151-158, Feb. 1994.

[10] N.M. Haller, "The S/KEY One-Time Password System," RFC 1760, Bellcore, Feb. 1995.

[11] T. Tsuji and A. Shimizu, "One-Time Password Authentication Protocol against Theft Attacks," IEICE Trans. Commun., vol.E87-B, no.3, pp. 523-529, March 2004.

[12] T. Tsuji, T. Kamioka, and A. Shimizu, "Simple and Secure Password Authentication Protocol, Ver.2 (SAS-2)," IEICE Technical Report, OIS2002-30, Sept. 2002.