Two Improved Multi-server Authentication Protocols Based on Hash Function and Smart Card

Qi Xie  
Zhejiang University / College of Computer Science, Hangzhou, P R China  
Hangzhou Normal University / School of Information Science and Engineering, Hangzhou, P R China  
Email: qixie68@yahoo.com.cn

Deren Chen  
Zhejiang University / College of Computer Science, Hangzhou, P R China  
Email: drc@zju.edu.cn

Abstract—To use the network services provided by multiple servers in mobile wireless network, recently, Tsai proposed a hash function and smart card based multi-server authentication protocol. Chen et al. showed that Tsai’s scheme cannot resist the server spoofing attack, and proposed a novel one. In this paper, we show that Chen et al.’s protocol cannot resist off-line password guessing attacks, and propose two improvements of Tsai and Chen et al.’s protocols, respectively. Two improved protocols have many advantages over existing solutions, as it relies on neither of encryption, signature, verification tables, timestamp, and public keys directory.

Index Terms—authentication protocol, multi-server, hash function, smart card, off-line password guessing attack

I. INTRODUCTION

With the rapid development of the network services, it is important to authenticate the user’s identity before allowing access to the servers’ resource. Most of the existing authentication schemes are based on encryption strategies, so the communication and computation costs are relatively high, and the public keys directory should be maintained and protected. To use the network services provided by servers in mobile wireless network, password and hash function based authentication using smart card is one of the simplest and the most widely used strategies [1]. However, there are two weaknesses in hash-function-based schemes. The one is that the server should store verification tables to verify the validity of the users, but it will suffer from the stolen-verifier attack [2]. The other is, timestamps are used to avoid replay attacks, but it needs time synchronization [3]. Therefore, it is important to design hash function and random nonce based authentication protocols without verification tables.

Since Chang and Wu [4] introduced password authentication scheme with smart card, many smart card based password authentication schemes have been proposed. However, many of them are vulnerable to off-line password guessing attack when attacker has ever obtained the information stored in smart card [5]. The reason is that user’s password is a short string with low entropy, and smart card always stores secret information. Therefore, one of the major challenges of designing smart card based password authentication protocols is how to resist the off-line password guessing attacks.

On the other hand, if remote users want to use network services provided by multiple servers, they must register their identities and passwords at these servers and become the legal users. Obviously, it is not efficient. To meet this demand, Li et al. [6], Lin et al. [7], Tsaur et al. [8][9] and Juang[10] proposed multi-server authentication schemes, respectively. In their schemes, any remote user only registers with the registration center once, and can obtain services from multiple servers without repeating registration to every single server. However, the communication and computation costs in these authentication schemes are relatively high, since these schemes used encryption strategies or neural networks. In 2006, Cao et al. [11] pointed out that Lin et al.’s scheme is insecure. In 2009, Liao et al. [12] proposed a dynamic ID scheme for multi-server environment. However, Hsiang et al. [13] found their scheme suffers from the insider attacks.

Recently, Tsai [14] proposed a hash-function-based multi-server authentication scheme using smart card and random nonce without verification tables, it is more efficient then before. However, Chen et al. [15] showed that Tsai’s scheme cannot resist the server spoofing attack, and proposed a novel one. They claimed that the new protocol is not only the most secure but also the most efficient in a multi-server environment.

In this paper, we point out that Chen et al.’s protocol cannot resist off-line password guessing attacks. Moreover, we propose two improvements of Tsai’s and Chen et al.’s protocols, respectively. Our protocols have
many advantages over existing solutions, as it relies on neither of encryption, signature, verification tables, timestamp, and public keys directory. Compared to the first improved protocol, the second improvement has the merit that the user can access the server’s resource without the help of the authentication center from the second time authentication.

The remainder of this article is organized as follows. In Section II, we propose an improvement of Tsai’s protocol. Then, we briefly review and analyze Chen et al’s protocol in Section III. After that, the proposed improvement of Chen et al’s protocol is presented and analyzed in Sections IV. Finally, the paper is concluded in Section V.

II. IMPROVEMENT OF TSAI’S PROTOCOL

In this section, we propose the improvement of Tsai’s protocol, and present the security analysis. The following notations are used throughout this paper:

- \( p \): a large prime number.
- \( g \): a primitive element for \( GF(p) \).
- \( AC \): a trusted authentication center.
- \( RC \): a trusted registration center.
- \( U_u \): one of the users.
- \( S_j \): one of the servers.
- \( ID_u \), \( SID_j \): \( U_u \)’s and \( S_j \’s \) identities, respectively.
- \( SC \): smart card.
- \( PW_u \): \( U_u \)’s password.
- \( SK \): a session key between \( U_u \) and \( S_j \).
- \( h() \): secure one-way hash function.

A. The Improved Protocol

The improved protocol has \( s \) servers, \( n \) users and an authentication center (\( AC \) for short). At the beginning, \( AC \) randomly chooses two secret numbers \( x \) and \( y \), computes and sends \( h(SID_j \| y) \) to each \( S_j \) via a secure channel, where \( SID_j \) is server \( S_j \’s \) identity. The improved protocol consists of four phases: user registration, login, authentication center authenticates the server and the user, mutual authentication and session key generation.

1. User registration

The user \( U_u \) sends his identity \( ID_u \) and password \( PW_u \) to \( AC \) via a secure channel. \( AC \) computes

\[ R_c = h(ID_u \| x) \quad \text{and} \quad C_0 = R_c \oplus h(PW_u) , \]

And stores one-way hash function \( h() \) and \( C_0 \) into a smart card (\( SC \) for short) and issues this smart card to \( U_u \) via secure channel.

2. Login

When the user \( U_u \) wants to get the session key with the server \( S_j \), he inserts smart card \( SC \) into a terminal device, keys password \( PW_u \). \( SC \) computes

\[ R_c = C_0 \oplus h(PW_u) \quad \text{and} \quad C_1 = h(SID_j \| R_c) \oplus N_c , \]

where \( N_c = g^c \text{ mod } p \) and \( c \) is a random nonce generated by the smart card, and \( SID_j \) is the server \( S_j \’s \) identity. Then \( SC \) sends \( ID_u \), \( SID_j \) and \( C_1 \) to \( AC \), and sends \( ID_u \), \( C_1 \) to the server \( S_j \).

3. Authentication center authenticates the server and the user

\( RSU1: S_j \rightarrow AC: ID_u, SID_j, C_1, C_2 \)

When the server \( S_j \) receives \( ID_u \) and \( C_1 \), it generates a random nonce \( s \), computes \( N_s = g^s \text{ mod } p \),

\[ C_2 = h(ID_u \| h(SID_j \| y)) \oplus N_s , \]

and sends \( ID_u \), \( SID_j \), and \( C_2 \) to \( AC \).

\( RSU2: AC \rightarrow U_u: C_3, C_4, AC \rightarrow S_j: C_5, C_6 \)

Upon receiving messages from \( U_u \) and \( S_j \), \( AC \) retrieves

\[ N_c = h(SID_j \| h(ID_u \| x)) \oplus C_1 , \]

\[ N_s = h(ID_u \| h(SID_j \| y)) \oplus C_2 , \]

and generates two random nonce \( r \) and \( Rc \), computes

\[ N_{rc1} = g^r \text{ mod } p , \]

\[ N_{rc2} = g^R \text{ mod } p , \]

\[ C_3 = N_{rc1} \oplus h(SID_j \| R_c) , \]

\[ C_4 = h(N_{rc1}^c \| ID_u) , \]

\[ C_5 = N_{rc2} \oplus h(ID_u \| h(SID_j \| y)) , \]

\[ C_6 = h(N_{rc1}^c \| SID_j) . \]

At last, \( AC \) sends \( C_3, C_4 \) to \( U_u \), and sends \( C_5, C_6 \) to the server \( S_j \).

\( RSU3: U_u \rightarrow AC: C_7; S_j \rightarrow AC: C_8 \)

\( U_u \) retrieves \( N_{rc1} = C_3 \oplus h(SID_j \| R_c) \), computes \( h(N_{rc1}^c \| ID_u) \) and verifies whether it equals to \( C_4 \). If correctness, then computes \( C_7 = h(N_{rc1}^c \oplus ID_u) \) and sends \( C_7 \) to \( AC \). Or else, the communication is rejected.

Similarly, the server \( S_j \) retrieves \( N_{rc2} \) and verifies the correctness of \( C_6 \), computes \( C_8 = h(N_{rc2}^c \oplus SID_j) \) and sends it to \( AC \).

\( RSU4: AC \rightarrow U_u: C_9, AC \rightarrow S_j: C_9 \)

When \( AC \) receives \( C_7 \) and \( C_8 \), it computes

\[ h(N_{rc1}^c \| ID_u) \quad \text{and} \quad h(N_{rc2}^c \| SID_j) , \]

checks whether they are equal to \( C_7 \) and \( C_8 \), respectively. If they are equal, \( AC \) computes

\[ C_9 = h(ID_u \| h(SID_j \| y) \| N_{rc2}^c + 1) \]

\[ \oplus h(SID_j \| h(ID_u \| x) \| N_{rc1}^c + 1) , \]

and sends it to \( U_u \) and \( S_j \).

4. Mutual authentication and session key generation

\( MA1: S_j \rightarrow U_u: C_{10}; U_u \rightarrow S_j: C_{11} \)

After receiving \( C_{10} \) from \( AC \), the server \( S_j \) retrieves

\[ h(SID_j \| h(ID_u \| x) \| N_{rc1}^c + 1) \]

© 2010 ACADEMY PUBLISHER
A. Brief Review of Chen et al.’s Protocol

Chen et al. claimed that their protocol can resist all attacks including off-line password guessing attack and smart-card-lost attack. However, we will show that their protocol is vulnerable to off-line password guessing attack.

III. ATTACKS ON CHEN ET AL.’S PROTOCOL

Chen et al.’s protocol consists of three phases: preparation phase, registration phase, authentication phase. The authentication phase can be divided into three scenarios: (1) the first time authentication and session key agreement phase; (2) not the first time authentication and session key agreement phase; (3) authentication and password change phase.

1. Preparation phase

The trusted registration center (RC) chooses his secret numbers \(x\) and \(y\), and computes \(RS_j = H(SID_j, y)\) for
each server $S_j$ with identity $SID_j$. Then, $RC$ sends $RS_j$ to $S_j$ via a secure channel.

2. Registration phase
User $U_u$ chooses his identity $ID_u$ and password $PW_u$, computes $H(ID_u, PW_u)$, and sends $(ID_u, H(ID_u, PW_u))$ to $RC$ via a secure channel.

$RC$ computes $B = H(ID_u, x) \oplus H(ID_u, PW_u)$, and stores $H()$, $ID_u$ and $B$ into smart card. Then $RC$ sends smart card to $U_u$ via a secure channel.

3. Authentication phase
(1) The first time authentication and session key agreement phase
When $U_u$ wants to access any resource of the server $S_j$, he inserts smart card into a terminal device, and performs the following steps:

Step 1: $U_u$ keys his identity $ID_u$ and password $PW_u$, the smart card computes $B_u = B \oplus H(ID_u, PW_u)$, generates a random nonce $c$ and computes $N_c = g^c$, $C_i = H(B_u, SID_u, N_c)$.

Step 2: $U_u$ sends $(ID_u, SID_u, C_i, N_c, Flag)$ to $S_j$, where $Flag$ is set to ‘the first time login’.

Step 3: When $S_j$ receives $(ID_u, SID_u, C_i, N_c, Flag)$ from $U_u$, he knows $U_u$ is the first time to login according to $Flag$, and generates a random nonce $s$, computes $N_u = g^s$, $V_u = H(RS_u, ID_u, N_u)$, then sends $(ID_u, SID_u, C_i, N_c, T_u, V_u, N_u)$ to $RC$.

Step 4: After receiving $(ID_u, SID_u, C_i, N_c, T_u, V_u, N_u)$, $RC$ computes

$$C'_i = H(H(ID_u, x), SID_u, N_c),$$
$$V'_u = H(H(SID_u, y), ID_u, N_u),$$

and checks if $C'_i = C_i$ and $V'_u = V_u$. If so, $U_u$ and $S_j$ are authenticated. Then, $RC$ computes

$$C_i = H(SID_u, H(ID_u, x), N_c),$$
$$V_i = H(ID_u, H(SID_u, y), N_u),$$

and sends $(C_i, V_i)$ to $S_j$.

Step 5: After receiving $(C_i, V_i)$, $S_j$ computes

$$V'_i = H(ID_u, RS_u, N_u)$$

and checks if $V'_i = V_i$. If so, $RC$ is authenticated. Then, $S_j$ computes the session key $SK = (N_u)^yi$ to be shared with $U_u$ and computes

$$C_3 = H(C_2 \oplus SK),$$
$$B_j = H(ID_u, RS_u),$$
$$B_u = B \oplus SK.$$

At last, $S_j$ sends $(C_3, N_u, B_u)$ to $U_u$.

Step 6: When $U_u$ receives $(C_3, N_u, B_u)$ from $S_j$, he computes

$$SK' = (N_u)^y,$$
$$C'_3 = H(H(SID_u, B_j, N_u, N_c) \oplus SK'),$$

and checks if $C_3 = C'_3$. If it holds, $S_j$ is authenticated by $U_u$. Then, $U_u$ computes

$$B_j' = B_j \oplus SK',$$
$$B_u = B_u \oplus H(ID_u, PW_u),$$

stores $B_u$ into smart card. Thus, without the help of the $RC$, $U_u$ can login to $S_j$ later. $SK' = SK = (g^y)^i$ is the common session key between $U_u$ and $S_j$.

(2) Not the first time authentication and session key agreement phase
When $U_u$ wants to access any resource of the server $S_j$ again, he inserts smart card into a terminal device, and performs the following steps:

Step 1: $U_u$ keys his identity $ID_u$ and password $PW_u$, the smart card computes $B_u = B \oplus H(ID_u, PW_u)$, and generates a random nonce $u$, computes $N_u = g^u$, $C = H(B_u', ID_u, SID_u, N_u)$.

Then, sends $(ID_u, SID_u, C, N_u, Flag)$ to $S_j$, where $Flag$ is set to ‘not the first time login’.

Step 2: After receiving $(ID_u, SID_u, C, N_u, Flag)$ from $U_u$, $S_j$ knows $U_u$ is not the first time to login according to $Flag$, he generates a random nonce $j$, computes $N_j = g^j$, $B_j = H(ID_u, RS_u)$,
$$C' = H(B_j, ID_u, SID_u, N_u),$$
and checks if $C' = C$. If so, $U_u$ is authenticated by $S_j$. $S_j$ computes the session key $K = (N_u)^y$ to be used with $U_u$, computes

$$U = H(N_u, K \oplus B_j),$$

and sends $(N_j, U)$ to $U_u$.

Step 3: After receiving $(N_j, U)$ from $S_j$, $U_u$ computes

$$K' = (N_u)^y,$$
$$U' = H(N_u, K \oplus B'_j),$$
and checks if $U' = U$. If so, $S_j$ is authenticated by $U_u$. $K' = K = g^u$ is the common session key between $U_u$ and $S_j$.

(3) Authentication and password change phase
When $U_u$ wants to change his password, he performs the following steps:

Step 1: $U_u$ keys his identity $ID_u$ and password $PW_u$, and his new password $PW'_u$, the smart card computes $B_u = B \oplus H(ID_u, PW'_u)$, and his new password $PW'_u$ is authenticated by $S_j$. Then, $S_j$ generates a random nonce $c$ and computes $N_c = g^c$, $C_i = H(B_u, SID_u, N_c)$.

Step 2: $U_u$ sends $(ID_u, SID_u, C_i, N_c, Flag)$ to $S_j$, where $Flag$ is set to ‘the first time login’.

Step 3: When $S_j$ receives $(ID_u, SID_u, C_i, N_c, Flag)$ from $U_u$, he knows $U_u$ is the first time to login according to $Flag$, and generates a random nonce $s$, computes $N_u = g^s$, $V_u = H(RS_u, ID_u, N_u)$, then sends $(ID_u, SID_u, C_i, N_c, T_u, V_u, N_u)$ to $RC$.

Step 4: After receiving $(ID_u, SID_u, C_i, N_c, T_u, V_u, N_u)$, $RC$ computes

$$C'_i = H(H(ID_u, x), SID_u, N_c),$$
$$V'_u = H(H(SID_u, y), ID_u, N_u),$$

and checks if $C'_i = C_i$ and $V'_u = V_u$. If so, $U_u$ and $S_j$ are authenticated. Then, $RC$ computes

$$C_i = H(SID_u, H(ID_u, x), N_c),$$
$$V_i = H(ID_u, H(SID_u, y), N_u),$$

and sends $(C_i, V_i)$ to $S_j$.

Step 5: After receiving $(C_i, V_i)$, $S_j$ computes

$$V'_i = H(ID_u, RS_u, N_u)$$

and checks if $V'_i = V_i$. If so, $RC$ is authenticated. Then, $S_j$ computes the session key $SK = (N_u)^yi$ to be shared with $U_u$ and computes

$$C_3 = H(C_2 \oplus SK),$$
$$B_j = H(ID_u, RS_u),$$
$$B_u = B \oplus SK.$$

At last, $S_j$ sends $(C_3, N_u, B_u)$ to $U_u$.

Step 6: When $U_u$ receives $(C_3, N_u, B_u)$ from $S_j$, he computes

$$SK' = (N_u)^y,$$
$$C'_3 = H(H(SID_u, B_j, N_u, N_c) \oplus SK'),$$

and checks if $C_3 = C'_3$. If it holds, $S_j$ is authenticated by $U_u$. Then, $U_u$ computes

$$B_j' = B_j \oplus SK',$$
generates a random nonce \( c \), computes
\[
N_r = g^r,
\]
\[
C_i = H(B_u, H(ID_u, PW_u^{\text{new}}), N_r),
\]
\[
CP_i = H(B_u, N_r) \oplus H(ID_u, PW_u^{\text{new}}),
\]
and checks if \( CP_i = C_i \). If so, \( U_u \) is authenticated. \( RC \) generates a random nonce \( r \), computes
\[
N_r = g^r,
\]
\[
K = g^{r \cdot r},
\]
\[
CP'_r = H(ID_r, K),
\]
\[
CP'_i = H(Flag, B_u, N_r, N_r),
\]
and checks if \( CP'_i = CP'_r \). If so, \( RC \) is authenticated.

Then, \( U_u \) checks if \( CP'_i = CP'_r \). If it holds, \( U_u \) knows his password change request has been accepted, and smart card replaces \( B \) with \( B_u \oplus H(ID_u, PW_u^{\text{new}}) \).

### B. Attacks on Chen et al.’s Protocol

In this section, we will show that Chen et al.’s protocol cannot resist the off-line password guessing attack. The details of attacking on his three authentication scenarios are as follows.

If an attacker gets the information \( B \) from the user \( U_u \) ‘s smart card, and obtains the interactive message \( \{C_i, N_r\} \) between \( U_u \) and \( S_j \). Then, attacker computes
\[
B_j^* = B \oplus H(ID_u, PW_u),
\]
\[
C_j = H(B_u, SID_j, N_r),
\]
where \( PW_u \) is the guessed password by attacker. If \( C_j = C_j^* \), the guessed password is correct. Else, he tries again. Since the password is always easy to remember, it may be not difficult to get the right password.

If an attacker gets the information \( B \) from the user \( U_u \) ‘s smart card, and obtains the interactive message \( \{ID_u, SID_j, C, N, Flag\} \) between \( U_u \) and \( S_j \). Then, attacker computes
\[
B_j = B \oplus H(ID_u, PW_u),
\]
where \( PW_u \) is the guessed password by attacker. Then attacker checks if \( C = H(B_j^*, ID_u, SID_j, N_r) \). If so, the guessed password is correct. Else, he tries again. As password is supposed to be short and memorizable, the password space is small and off-line attack is feasible.

If an attacker gets user \( U_u \) ‘s password change request \( \{ID_u, SID_j, C_i, CP_i, N_r, Flag\} \) and interactive message \( \{CP_r, CP_r, N_r\} \). Then, attacker does off-line compute
\[
B_u = B \oplus H(ID_u, PW_u),
\]
\[
CP' = H(Flag, B_u, N_r, N_r),
\]
till \( CP'' = CP'_r \), where \( PW''_u \) is the guessed password. Then, attacker does off-line compute
\[
C_i = H(B_u, H(ID_u, PW_u^{\text{new}}), N_r)
\]
till \( C_i = C_i^* \), where \( PW''_u \) is the guessed password. Therefore, Chen et al.’s protocol is insecure.

### IV. IMPROVEMENT OF CHEN ET AL.’S PROTOCOL

To overcome the weaknesses of Chen et al.’s protocol, in this section, we will propose the improved protocol.

#### A. Improvement of Chen et al.’s Protocol

The improved protocol consists of three phases: preparation phase, registration phase, authentication phase. The preparation phase and registration phase are the same as those of Chen et al.’s protocol, authentication phase can also be divided into three scenarios: (1) the first time authentication and session key agreement phase; (2) not the first time authentication and session key agreement phase; (3) authentication and password change phase. The details are as follows.

1. The first time authentication and session key agreement phase

When \( U_u \) wants to access any resource of the server \( S_j \), he inserts smart card into a terminal device, and performs the following steps:

**Step 1:** \( U_u \rightarrow RC: ID_u, SID_j, C_i, Flag \)

**Step 2:** \( U_u \rightarrow S_j: ID_u, C_i, Flag \)

\( U_u \) keys his identity \( ID_u \) and password \( PW_u \), smart card computes
\[
R_s = B \oplus H(ID_u, PW_u),
\]
\[
C_i = h(SID_j \parallel R_s) \oplus N_c,
\]
where \( N_c = g^c \) mod \( p \) and \( c \) is a random nonce generated by the smart card, and \( SID_j \) is the server \( S_j \)’s identity. Then \( SC \) sends \( ID_u, SID_j, C_i \) and \( Flag \) to \( RC \), sends \( ID_u, C_i \) and \( Flag \) to the server \( S_j \), where \( Flag \) is set to ‘the first time login’.

**Step 2—Step 5:** The same as steps RSU1—RSU4 of improvement of Tsai’s protocol in section II.

**Step 3:** The same as step MA1 of improvement of Tsai’s protocol in section II.

**Step 4:**

**Step 5:**

**Step 6:** The same as step MA1 of improvement of Tsai’s protocol in section II.

**Step 7:**

**Step 8:**
After receiving \( C_{10} \) and \( C_{11} \), \( S_j \) retrieves \( N_{c2} \), computes
\[
 h(N_{c2}^S \mod p \| ID_u \| SID_j) \quad \text{and} \quad B_j = H(ID_u, RS_j) \oplus h(N_{c2}^S \mod p),
\]
and then sends them to \( U_u \).

\( U_u \) retrieves \( N_{c2} \), computes
\[
 h(N_{c2}^S \mod p \| ID_u \| SID_j),
\]
and \( U_u \) sends it to \( S_j \).

Step 8: Mutual authentication and session key generation
\( U_u \) and \( S_j \) compute and verify whether
\[
 h(N_{c2}^S \mod p \| ID_u \| SID_j),
\]
are correct, respectively. If they are correct, \( S_j \) and \( U_u \) will define a session key
\[
 SK = h(C_0 \| N_{c2}^S + 1) = h(C_0 \| N_{c2}^S + 1).
\]
Then, \( U_u \) generates the new password \( PW_u \) to login \( S_j \) later, and computes
\[
 H(ID_u, RS_j) = B_j \oplus h(N_{c2}^S \mod p),
\]
\[
 B_u = H(ID_u, RS_j) \oplus H(ID_u, PW_u^c),
\]
and stores \( B_u \) into smart card. Thus, without the help of the \( RC \), \( U_u \) can login to \( S_j \) later.

(2) Not the first time authentication and session key agreement phase

When \( U_u \) wants to access any resource of the server \( S_j \) again, he inserts smart card into a terminal device, and performs the following steps:

Step 1: \( U_u \rightarrow S_j : ID_u, SID_j, C_1, Flag \)

\( U_u \) keys his identity \( ID_u \) and password \( PW_u \), the smart card computes \( B'_u = B_u \oplus H(ID_u, PW_u^c) \), and generates a random nonce \( u \), computes \( N_u = g^u \mod p \),
\[
 C_1 = H(B'_u, ID_u, SID_j) \oplus N_u ,
\]
then sends \( \{ID_u, SID_j, C_1, Flag\} \) to \( S_j \), where Flag is set to ‘not the first time login’.

Step 2: \( S_j \rightarrow U_u : C_1, V_2 \)

After receiving \( \{ID_u, SID_j, C_1, Flag\} \) from \( U_u \), \( S_j \) computes
\[
 N_u = H(ID_u, RS_j), ID_u, SID_j) \oplus C_1 ,
\]
and generates a random nonce \( s \), computes
\[
 N_s = g^s \mod p ,
\]
\[
 C_2 = H(ID_u, RS_j), ID_u, SID_j) \oplus N_s ,
\]
\[
 SK = H(N_u^s, ID_u, SID_j) ,
\]
\[
 V_2 = H(SK, N_u^s) ,
\]
and sends \( \{C_2, V_2\} \) to \( U_u \).

Step 3: \( U_u \rightarrow S_j : V_1 \)

After receiving \( \{C_2, V_2\} \), \( U_u \) computes
\[
 N_s = H(B'_u, ID_u) \oplus C_2 ,
\]
\[
 SK = H(N_u^s, ID_u, SID_j) ,
\]
\[
 V_1 = H(B'_u, N_u^s) ,
\]
\[
 H(SK, N_u^s) ,
\]
and checks if \( H(SK, N_u^s) = V_2 \), If so, \( S_j \) is authenticated. Then, \( U_u \) sends \( V_1 \) to \( S_j \).

Step 4: \( S_j \) computes \( H(ID_u, RS_j, N_u^s) \), and checks if \( V_1 = H(ID_u, RS_j, N_u^s) \). If so, \( U_u \) is authenticated.
\[
 SK = H(N_u^s, ID_u, SID_j) \text{ (the common session key between } U_u \text{ and } S_j \text{)}
\]

(3) Authentication and password change phase

When \( U_u \) wants to change his password, he performs the following steps:

Step 1: \( U_u \) keys his identity \( ID_u \) and password \( PW_u \), and his new password \( PW_u^\text{new} \), the smart card computes \( B_u = B \oplus H(ID_u, PW_u^c) \), generates a random nonce \( c \), computes
\[
 N_c = g^c \mod p ,
\]
\[
 C_1 = H(B_u) \oplus N_c ,
\]
and sends \( \{ID_u, C_1, Flag\} \) to \( RC \), where Flag is set to ‘for password change’.

Step 2: \( RC \) computes
\[
 N_c = H(ID_u, x) \oplus C_1 ,
\]
Then, \( RC \) generates a random nonce \( r \), computes
\[
 N_{RC1} = g^r \mod p ,
\]
\[
 C_4 = N_{RC1} \oplus h(ID_u \| h(ID_u \| x)) ,
\]
\[
 C_4 = h(N_c^r \| ID_u) ,
\]
and sends \( C_3, C_4 \) to \( U_u \).

Step 3: \( U_u \) retrieves
\[
 N_{RC1} = C_3 \oplus h(ID_u \| B_u) ,
\]
computes \( h(N_c^{r} \| ID_u) \) and verifies whether it equals to \( C_4 \). If correctness, then computes
\[
 C_2 = h(N_c^{r} \| ID_u) ,
\]
\[
 CP_1 = H(B_u, N_{RC1} + 2) \oplus H(ID_u, PW_u^{new}) ,
\]
\[
 C_5 = H(B_u, H(ID_u, PW_u^{new}), N_{RC1}^c) ,
\]
and sends \( C_2, CP_1, C_5 \) to \( RC \). Or else, the communication is rejected.

Step 4: \( RC \) computes \( h(N_c^{r} \| ID_u) \) and checks if
\[
 C_2 = h(N_c^{r} \| ID_u) .
\]
If so, he computes
\[
 H(ID_u, PW_u^{new}) = CP_1 \oplus H(ID_u, x, N_c^{r} + 2) \]
and checks if
\[
 H(H(ID_u, x), H(ID_u, PW_u^{new}), N_c^{r}) = C_5 .
\]
If so, \( RC \) computes
$V = H(Flag, H(ID_u, x), N_c + 1, H(ID_u, PW_u^{new}))$, where Flag is set to ‘accept’.

Then, $R_C$ sends $\{V, Flag\}$ to $U_u$.

Step 5: Smart card computes

$V' = H(Flag, R_c, N_{RC1} + 1, H(ID_u, PW_u^{new}))$, and checks if $V' = V$. If so, $U_u$ knows his password change request has been accepted, and smart card replaces $B$ with $B_c \oplus H(ID_u, PW_u^{new})$.

B. Security Analysis

1. Password guessing attack and smart card lost attack
If an attacker obtains all information of user $U_u$’s smart card, and wants to guess his password $PW_u$, however, it is impossible. The reason is that the interactive messages $C_i = h(SID_j || R_c) \oplus N_c$ between $U_u$ and $S_j$ include the random nonce $c$, where $N_c = g^c \mod p$. Therefore, an attacker cannot get the correct password $PW_u$ if he does off-line password guessing attack by computing $R_c = B \oplus H(ID_u, PW_u^{new})$ and checks if $C_i = h(SID_j || R_c) \oplus N_c$, where $PW_u^{new}$ is guessed password. Moreover, all the interactive messages between $U_u$ and $S_j$, or between $U_u$, $S_j$ and $R_C$ are include fresh random nonce, so the proposed improvement can resist the off-line password guessing attacks.

On the other hand, if an attacker does on-line password guessing attack, it is still impossible. The reason is that the success probability of doing on-line dictionary attack is $O(U \mid D)$, where $D$ is password dictionary.

2. Replay attack
If attacker replay user $U_u$’s login message $C_i$, $R_C$ can retrieve $N_c$ but attacker cannot, which include the random nonce $c$, and $R_C$ can return the authentication message which includes the Diffie-Hellman value $g^c$, so attacker cannot verify whether the authentication message from $R_C$ right or not. On the other hand, the attacker cannot generate the authentication message that can make pass through the authentication of $R_C$.

Therefore, the proposed protocol can resist the replay attack.

3. Impersonation attack and forgery attack
If a legal user $U_u$ wants to impersonate $U_u$ to access the server $S_j$’s resource, however it is impossible. The reason is that $U_u$ cannot pass the authentication of $RC$ or $S_j$ without knowing the $U_u$’s password.

4. Forward security
In improved protocol, the session key is forward security. Since an attacker cannot obtain all the random nonce even if he knows the user’s password, all information of smart card, and the nonce’s discrete logarithm, so he cannot get the Diffie-Hellman values between the server and the user. Therefore, an attacker cannot compute $SK$.

5. Session key security
In improved protocol, the session key is not known by anyone but the server and the user, since the session key

$SK = h(C_0 || N_{c2} + 1) = h(C_0 || N_{c2} + 1)$,

$SK = H(N_c^{new}, ID_u, SID_j)$,

which include the Diffie-Hellman values between the server and the user, and the protocol is forward security, so attacker cannot get the random nonce and the Diffie-Hellman values.

Therefore, the session key is security.

6. Man-in-the-middle attack
In improved protocol, all the authentication messages of the server and the user include identities of two communication entities. The adversary cannot pass through the authentication since he doesn’t own the secret information $x$ and $y$ of $RC$, and the user and the server’s password. So, the improved protocol can resist man-in-the-middle attack.

V. CONCLUSIONS

Nowadays, Tsai’s and Chen et al.’s authentication protocols are the most efficient in a multi-server environment. However, their protocols are all insecure. In this paper, we presented off-line password guessing attacks on Chen et al.’s protocol, and proposed two improved protocols to overcome Tsai’s and Chen et al.’s weaknesses, respectively. In the future work, it is very interesting to study formal security model of this protocols.

REFERENCES


Qi Xie was born in Shangyu, Zhejiang, P R China. December 26, 1968. He received the B.S. in Department of mathematics from Hangzhou Normal University, Hangzhou, Zhejiang, in 1990; the M.S. in operational research from Shanghai University, Shanghai, in 1997; and a Ph.D. in applied mathematics from Zhejiang University, Hangzhou, Zhejiang, in 2005. Currently, he is a postdoctoral researcher in College of Computer Science at Zhejiang University, P R China; a visiting scholar at University of Birmingham in UK.

He has been working at Department of computer Science, Hangzhou Normal University, P R China, since 1997. Dr. Xie has published over 30 academic dissertations in all major refereed journals and in the proceedings of international conferences. His research interests are digital signatures, authentication and key exchange protocols.

Dr. Xie is a professor in the school of Information Science and Engineering, Hangzhou Normal University, P R China, a reviewer for ‘Mathematics reviews’ of America, and a senior member of Chinese Association for Cryptologic Research.

Deren Chen was born in Hangzhou, P R China, November 1951. He received the B.S. and M.S. in applied mathematics from Zhejiang University, Hangzhou, Zhejiang, P R China. He has been the visiting scientist to IBM Watson Research Center in USA and Visiting Professor to the School of Applied Science at Nanyang Technological University in Singapore.

Over recent years, he has been taking projects spreading aspects in services science, electronic commerce and online education, etc. He has also accomplished over 60 projects, Project of National Natural Science Foundation, Project for Science and Technology etc. Five of which have been granted State Science and Technology Awards and Teaching Achievement Awards. In addition, Prof. Chen has been the Chief Editor for over 15 books, and he has published over 160 academic dissertations. His research interests are services science, electronic commerce and online education, etc.

Prof. Chen is a Professor of Computer Science & Technology at ZJU, Director of Electronic-Service Research Center at ZJU, as well as one of the State Council Experts for Special Allowance, he is also entitled the Director of Electronic-Service Expert Council at China E-commerce Association, member of Online Education Exams of National Colleges, Ministry of Education, P.R.C., Editorial Board member at International Journal of Web Services Research (JWSR),US. Meanwhile, he has taken the responsibility as the co-chair of over 20 international academic conferences, Program Chair of International Conference.