Comments on Three Multi-Server Authentication Protocols

Yalin Chen [1], *Jue-Sam Chou[2], Wen-Yi Tsai [3]

[1] Institute of information systems and applications, National Tsing Hua University
d949702@oz.nthu.edu.tw
[2,3] Department of Information Management, Nanhua University Chiayi 622 Taiwan, R.O.C
*: corresponding author
jschou@mail.nhu.edu.tw
Tel: 886+ (0)5+272-1001 ext.56226

Abstract

Recently, Tsai et al., Liao *et al.* and Li *et al.* each proposed a multi-server authentication protocol. They claimed their protocols are secure and can withstand various attacks. However, we found some security loopholes in each of their schemes, for example, both Tsai et al.'s and Liao *et al.*'s schemes suffers from server spoofing attack by an insider server. Li *et al.*s' suffers from the lost smart card password-guessing attack. In addition, Liao *et al.*'s scheme also has the off-line password-guessing attack. In this paper, we will first review then show the attacks on each of the schemes. Then, based on Li *et al.'s* scheme, we proposed a novel one and examined its security in several security features. After security analysis, we concluded that our protocol outperformed Li *et al.'s* scheme in the security feature of lost smart card password-guessing attack.

Keywords: *multi-server, password authentication protocol, server spoofing attack, password-guessing attack, insider attack*

1. Introduction

A two-party password authentication protocol for client-server architecture is often not sufficient as a network getting larger nowadays. Consequently, several multi-server protocols were proposed [1-16].

In 2003, Li *et al*. [5] proposed a multi-server protocol based on ElGamal digital signature and geometric transformations on an Euclidean plane. Unfortunately, their protocol is vulnerable and has been broken by Cao and Zhong [8]. In 2004 and 2005, Tsaur *et al*. [3, 4] proposed two multi-server schemes. However, both of their schemes are based on Lagrange interpolating polynomial which is computationally intensive, and were broken by Chou *et al.* [17]. In 2006 and 2007, Cao *et al*. [9] and Hu *et al*. [7] each proposed an authentication scheme for multi-server environment. Both schemes assume that all servers are trustworthy. Nevertheless, this assumption is somewhat impractical as stated in [1]. In 2008, Lee *et al*. [6] proposed an authenticated key agreement scheme for multi-server using mobile equipment. However, their scheme can not add a server freely. Because when a server is added, all users who want to login to this new server have to re-register at the registration center for getting a new smart card. This increases the registration center's card-issue cost. Also, in 2008, Tsai [1] proposed an efficient multi-server authentication scheme. He claims that his protocol can withstand seven known attacks. Yet, after our analysis, we found that it is vulnerable to the server spoofing attack**.** Recently, in 2009, Liao and Wang [2] proposed a secure dynamic ID scheme for multi-server environment. They claim that their protocol is secure. However, we found their scheme suffers from both the server spoofing attack. Most recently in 2013, Li et al. [16] also propose a novel multi-server scheme and claim that their scheme is secure. However, we found it has the smart card lost password-guessing attack. In this paper, we will first show the attacks on [1] and [2], respectively. Then, we show the attack on [16], meanwhile we also propose a novel one. After security analysis, we concluded that our scheme not only avoid the lost password-guessing attack but also more efficient than [16] in the protocol's number of passes.

The remainder of this paper is organized as follows: In Section 2 and 3, we review and show the attack on Tsai's protocol and Liao-Wang's protocol, respectively. Section 4 first demonstrates and attacks on Li et al.'s protocol, then propose a novel one and examine its security. Finally, a conclusion is given in Section 5.

2. Review of Tsai's protocol

In this section, we review Tsai's protocol in Section 2.1 and examine its security in Section 2.2. Before that, the notations used throughout this paper are first defined as follows.

RC	: the registration center,	U_u	: a legal user u
S_j	: a legal server j,	SID_j	: the identity of S_j
E(P)	: an attacker E who masquerades as a peer P,		
ID_u	: the identity of U_u,	PW_u	: the password of U_u
x,y	: RC's two secret keys,	p	:a large prime number
g	: the primitive element in a Galois field GF(p),	$\oplus$	: a bitwise Xor operator
H( )	: a collision-resistant one-way hash function,	=>	: a secure channel
(a,b)	: a string denotes that string a is concatenated with string b		
Δ T	: a tolerant time delay for messages transmission over network		
→	: a common channel		

2.1 The protocol

Tsai's protocol contains four phases. They are: (1)user registration phase, (2)login phase, (3)authentication of server and RC phase, and (4)authentication of server and user phase. We describe it as follows and also depict phases (1), (2) in Figure 1, phase (3) in Figure 2, and phase (4) in Figure 3.

Assume that there are s servers in the system. At beginning, RC computes and sends $H(SID_j,y)$ to S_j, for j = 1 to s, with S_j keeping it secret, via a secure channel.

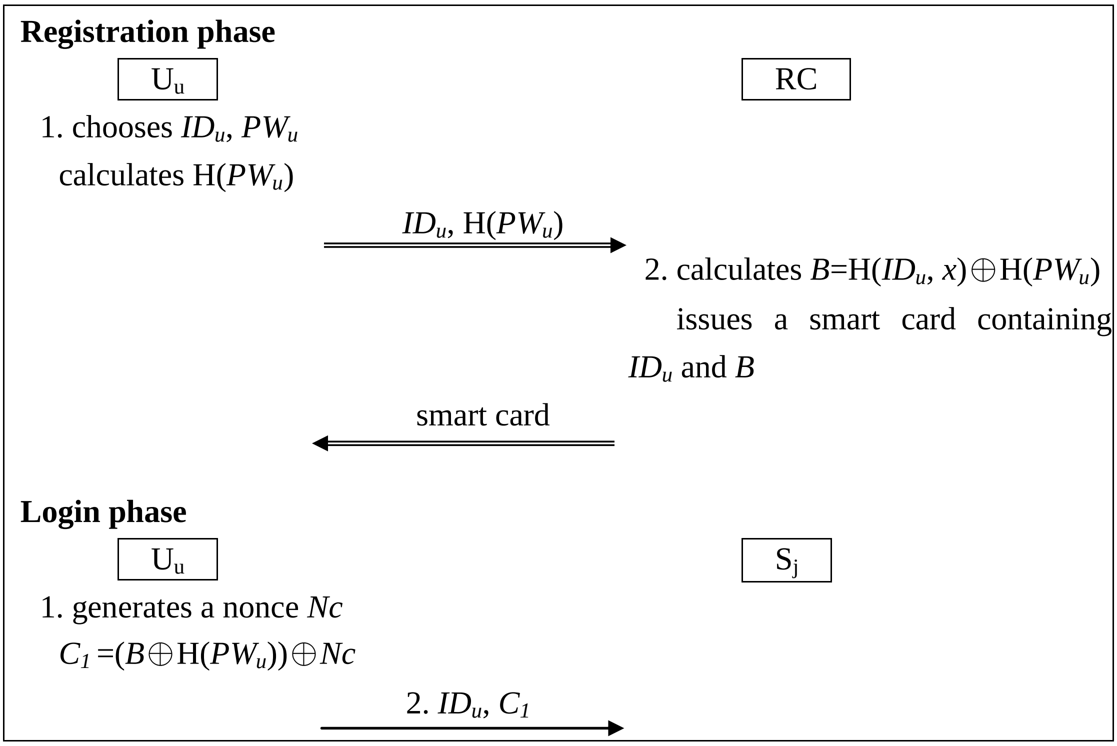


Fig. 1. Registration phase and login phase of Tsai's protocol

(1) Registration phase

In this phase, U_u performs the following steps for obtaining a smart card from RC.

1. U_u freely chooses his ID_u and PW_u and calculates $H(PW_u)$. He then sends $\{ID_u, H(PW_u)\}$ to RC through a secure channel.
2. RC calculates $B=H(ID_u, x)\oplus H(PW_u)$ and issues to U_u a smart card containing ID_u and B through a secure channel.

(2) Login phase

When U_u wants to login to S_j, he inserts his smart card and performs the following steps.

1. U_u keys his ID_u and PW_u and generates a random nonce Nc. He then computes $C_1 = (B \oplus H(PW_u)) \oplus Nc = H(ID_u, x) \oplus Nc$.
2. U_u sends $\{ID_u, C_1\}$ to S_j.

(3) Authentication of server and RC phase

In this phase, when receiving message $\{ID_u, C_1\}$ from U_u, S_j will run the following steps to let himself be authenticated by RC, verify U_u's legitimacy, and negotiate a session key with U_u. The secret key shared between S_j and RC is $H(H(SID_j, y), Ns+1, N_{RC}+2)$, where Ns and N_{RC} are S_j's and RC's randomly chosen nonces respectively. To reduce the computational cost, this phase is divided into two scenarios: (a) the secret key is not generated, and (b) the secret key has been generated. We describe them below.

Authentication of server and RC phase

(a) the secret key is not generated

S_j RC

1. generates a nonce Ns
   computes $C_2 = H(SID_j, y) \oplus Ns$.

2. ID_u, SID_j, C_1, C_2 →

3. derives $Ns' = H(SID_j, y) \oplus C_2$
   generates a nonce N_{RC}
   computes $C_3 = N_{RC} \oplus H(SID_j, y)$

← 4. C_3

5. retrieves
   $N'_{RC} = C_3 \oplus H(SID_j, y)$
   calculates
   $C_4 = H(H(SID_j, y), Ns) \oplus N'_{RC}$

6. C_4 →

7. computes
   $C'_4 = H(H(SID_j, y),$

Ns')$\oplus N_{RC}$
checks C'_4 =? C_4
retrieves $N'c = \mathrm{H}(ID_u, x)\oplus C_1$
computes
$C_5 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns', N_{RC})$,
$C_6 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns'+1, N_{RC}+2)\oplus \mathrm{H}(\mathrm{H}(ID_u, x), N'c)$

8. C_5, C_6 ←

9. calculates
$C'_5 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns, N'_{RC})$
compares C'_5 =? C_5

(b) the secret key has been generated

S_j RC

1. ID_u, SID_j, C_1 →

2. derives $N'c = \mathrm{H}(ID_u, x)\oplus C_1$
computes
$C_6 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns'+1, N_{RC}+2) \oplus \mathrm{H}(\mathrm{H}(ID_u, x), N'c)$

3. C_6 ←

Fig. 2. Authentication of server and RC phase of Tsai's protocol

(a) The secret key is not generated.

1. S_j generates a random nonce Ns and computes $C_2 = \mathrm{H}(SID_j, y)\oplus Ns$.
2. S_j sends $\{ID_u, SID_j, C_1, C_2\}$ to RC.
3. RC derives $Ns' = \mathrm{H}(SID_j, y)\oplus C_2$. He then generates a random nonce N_{RC} and computes $C_3 = N_{RC}\oplus \mathrm{H}(SID_j, y)$.
4. RC sends $\{C_3\}$ to S_j.
5. After receiving the message from RC, S_j retrieves $N'_{RC} = C_3\oplus \mathrm{H}(SID_j, y)$ and calculates $C_4 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns)\oplus N'_{RC}$.
6. S_j sends $\{C_4\}$ to RC.
7. RC computes $C'_4 = \mathrm{H}(\mathrm{H}(SID_j, y), Ns')\oplus N_{RC}$ and checks to see if C'_4 is equal to

the received C_4. If so, S_j is authentic. He then retrieves $N'_c = H(ID_u, x) \oplus C_1$ and computes $C_5 = H(H(SID_j, y), Ns', N_{RC})$, $C_6 = H(H(SID_j, y), Ns'+1, N_{RC}+2) \oplus H(H(ID_u, x), N'_c)$.

8. RC sends $\{C_5, C_6\}$ to S_j.
9. After receiving the message from RC, S_j calculates $C'_5 = H(H(SID_j, y), Ns, N'_{RC})$ and compares to see if C'_5 is equal to the received C_5. If so, RC is authentic. Both S_j and RC will store the common secret key $Auth_{S\text{-}RC} = H(H(SID_j, y), Ns+1, N'_{RC}+2)$ for the next time execution of this authentication, authentication of server and RC, to reduce the computational cost.

(b) the secret key has been generated.

1. S_j sends $\{ID_u, SID_j, C_1\}$ to RC.
2. RC derives $N'_c = H(ID_u, x) \oplus C_1$ and uses his $Auth_{S\text{-}RC}$ to compute $C_6 = Auth_{S\text{-}RC}$ ($= H(H(SID_j, y), Ns'+1, N_{RC}+2)$) $\oplus H(H(ID_u, x), N'_c)$.
3. RC sends $\{C_6\}$ to S_j.

(4) Authentication of server and user phase

After the authentication of server and RC phase, S_j and U_u perform the following steps for mutual authentication.

1. S_j generates a random nonce N_{SU} and uses his $Auth_{S\text{-}RC}$ to compute $C_7 = C_6 \oplus Auth_{S\text{-}RC}$ ($= H(H(SID_j, y), Ns+1, N'_{RC}+2)$) $= H(H(ID_u, x), N'_c)$. He then calculates $C_8 = C_1 \oplus C_7$, $V_2 = C_7 \oplus N_{SU}$, and $C_9 = H(C_7, N_{SU}) \oplus C_8$.
2. S_j sends $\{V_2, C_9\}$ to U_u.
3. After receiving the message, U_u computes $C'_7 = H(H(ID_u, x), Nc)$, retrieves $N'_{SU} = C'_7 \oplus V_2$, and calculates $C'_8 = C'_7 \oplus C_1$, $C'_9 = H(C'_7, N'_{SU}) \oplus C'_8$. He then checks to see if the computed C'_9 is equal to the received C_9. If so, S_j is authentic. U_u continues to calculate $C_{10} = H(C'_7, C'_8, N'_{SU})$.
4. U_u sends $\{C_{10}\}$ to S_j.
5. After receiving $\{C_{10}\}$, S_j computes $C'_{10} = H(C_7, C_8, N_{SU})$ and compares to see if C'_{10} is equal to the received C_{10}. If so, U_u is authentic. They then have the same session key SK$= H(C'_7+1, C'_8+2, N'_{SU}+3) = H(C_7+1, C_8+2, N_{SU}+3)$.

Authentication of server and user phase

U_u | S_j

1. generates a nonce N_{SU}

computes

$C_7 = C_6 \oplus H(H(SID_j, y), Ns+1, N'_{RC}+2) = H(H(ID_u, x), N'c)$

calculates

$C_8 = C_1 \oplus C_7$, $V_2 = C_7 \oplus N_{SU}$

$C_9 = H(C_7, N_{SU}) \oplus C_8$

← 2. V_2, C_9

3. computes

$C'_7 = H(H(ID_u, x), Nc)$

retrieves

$N'_{SU} = C'_7 \oplus V_2$

calculates

$C'_8 = C'_7 \oplus C_1$

$C'_9 = H(C'_7, N'_{SU}) \oplus C'_8$

checks $C'_9 =? C_9$,

calculates

$C_{10} = H(C'_7, C'_8, N'_{SU})$

4. C_{10} →

5. computes $C'_{10} = H(C_7, C_8, N_{SU})$

compares $C'_{10} =? C_{10}$

session key SK= $H(C_7+1, C_8+2, N_{SU}+3)$

5.session key

SK= $H(C'_7+1, C'_8+2, N'_{SU}+3)$

Fig. 3. Authentication of server and user phase of Tsai's protocol

2.2 Attack on Tsai's protocol

After analysis, we found Tsai's protocol suffers server spoofing attack in both scenarios. We will show the security loopholes in the following.

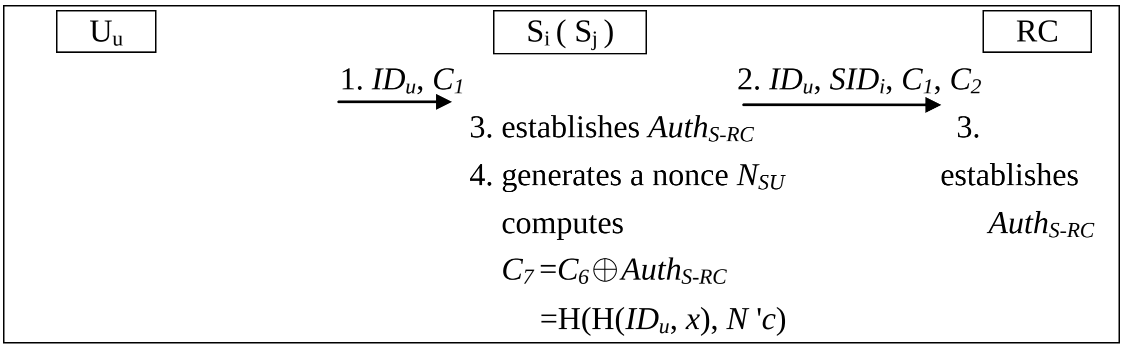
- **Server spoofing attack by an insider server on Tsai's protocol**

Assume that S_i is a legal server at RC. He also has his $H(SID_i, y)$ and keeps it secret. He can then masquerade as another legal server to cheat a remote user, because in the authentication of server and user phase, a user doesn't examine to see whether the message is really sent from the correct server. In the following, we present the server spoofing attacks on the two mentioned scenarios, (1) the secret key is not generated and (2) the secret key has been generated, and also depict them in Figure 4 and 5, respectively.

(1) The secret key is not generated.

1. When U_u wants to communicate with S_j, he starts the protocol and sends $\{ID_u, C_1\}$ to S_i who masquerades as S_j .
2. S_i generates a nonce Ns, computes $C_2 = H(SID_i, y) \oplus Ns$, and sends $\{ID_u, SID_i, C_1, C_2\}$ to RC. Because the subsequent messages C_3, C_4, C_5 and C_6, except C_6, sent between RC and S_i to authenticate each other are independent on U_u's secrecy $H(H(ID_u, x), Nc)$ as depicted in scenario (a) of Figure 2. RC and S_i will thus be able to achieve mutual authentication successfully.
3. RC and S_i then negotiate to establish the common secret key $Auth_{S\text{-}RC}=H(H(SID_i, y), Ns+1, N'_{RC}+2)=H(H(SID_i, y), Ns'+1, N_{RC}+2)$ in the server and RC authentication phase. After that, S_i and U_u will perform the following steps for the server and user authentication phase.
4. S_i generates a random nonce N_{SU} and uses his $Auth_{S\text{-}RC}$ to compute $C_7 = C_6 \oplus Auth_{S\text{-}RC} = H(H(ID_u, x), N'c)$. He then calculates $C_8 = C_1 \oplus C_7$, $V_2 = C_7 \oplus N_{SU}$, and $C_9 = H(C_7, N_{SU}) \oplus C_8$.
5. S_i sends $\{V_2, C_9\}$ to U_u.
6. After receiving the message, U_u computes $C'_7 = H(H(ID_u, x), Nc)$, retrieves $N'_{SU} = C'_7 \oplus V_2$, and calculates $C'_8 = C'_7 \oplus C_1$, $C'_9 = H(C'_7, N'_{SU}) \oplus C'_8$. He then checks to see if C'_9 is equal to the received C_9. If so, U_u confirms that the message is from the server who had received his C_1 in the login phase. S_i disguising as S_j is thus regarded as authentic. U_u continues to calculate $C_{10} = H(C'_7, C'_8, N'_{SU})$.
7. U_u sends $\{C_{10}\}$ to S_i.
8. S_i computes $C'_{10} = H(C_7, C_8, N_{SU})$ and compares to see if C'_{10} is equal to the received C_{10}. If so, U_u is authentic. They then compute the common session key $SK= H(C'_7+1, C'_8+2, N'_{SU}+3) = H(C_7+1, C_8+2, N_{SU}+3)$.

From the above-mentioned steps, we can see that a server spoofing attack can be successfully launched by the insider attacker S_i.



$C_8 = C_1 \oplus C_7$

$V_2 = C_7 \oplus N_{SU}$

$C_9 = \text{H}(C_7, N_{SU}) \oplus C_8$

5. V_2 , C_9 ←

6. computes

$C'_7 = \text{H}(\text{H}(ID_u, x), Nc)$

retrieves $N'_{SU} = C'_7 \oplus V_2$

calculates $C'_8 = C'_7 \oplus C_1$

$C'_9 = \text{H}(C'_7, N'_{SU}) \oplus C'_8$

checks $C'_9 =? C_9$

calculates

$C_{10} = \text{H}(C'_7, C'_8, N'_{SU})$

7. C_{10} →

8. computes

$C'_{10} = \text{H}(C_7, C_8, N_{SU})$

compares $C'_{10} =? C_{10}$

session key

$\text{SK} = \text{H}(C_7+1, C_8+2, N_{SU}+3)$

8. session key

$\text{SK} = \text{H}(C'_7+1, C'_8+2, N'_{SU}+3)$

Fig.4. Server spoofing attack by an insider server on Tsai's protocol:(a) the secret key is not generated.

(2) The secret key has been generated.

For this case, we describe the attack as follows and also illustrate it in Figure 5.

1. U_u starts the protocol and sends $\{ID_u, C_1\}$ to S_i who masquerades as S_j.
2. When S_i runs the authentication of server and RC phase, he simply sends $\{ID_u, SID_i, C_1\}$ to RC. RC deduces $N'c = \text{H}(ID_u, x) \oplus C_1$ and computes $C_6 = \text{H}(\text{H}(SID_i, y), Ns'+1, N_{RC}+2) \oplus \text{H}(\text{H}(ID_u, x), N'c)$.
3. RC sends $\{C_6\}$ to S_i as depicted in scenario (b) of Figure 2. S_i then continues the following steps with U_u for the server and user authentication phase.
4. S_i generates a random nonce N_{SU} and uses the generated common secret key $Auth_{S\text{-}RC}$ to compute $C_7 = C_6 \oplus Auth_{S\text{-}RC} = \text{H}(\text{H}(ID_u, x), N'c)$. He then calculates $C_8 = C_1 \oplus C_7$, $V_2 = C_7 \oplus N_{SU}$, and $C_9 = \text{H}(C_7, N_{SU}) \oplus C_8$.
5. S_i sends $\{V_2 , C_9\}$ to U_u.
6. After receiving the message, U_u computes $C'_7 = \text{H}(\text{H}(ID_u, x), Nc)$, retrieves $N'_{SU} = C'_7 \oplus V_2$, and calculates $C'_8 = C'_7 \oplus C_1$, $C'_9 = \text{H}(C'_7, N'_{SU}) \oplus C'_8$. He then checks

to see if C'_9 is equal to the received C_9. If so, U_u confirms that the message is sent from the right server who had received his C_1 in the login phase; and S_i disguising as S_j is therefore regarded as being authentic. U_u then proceeds to calculate $C_{10} = H(C'_7, C'_8, N'_{SU})$.

7. U_u sends $\{C_{10}\}$ to S_i.
8. After obtaining the message, S_i computes $C'_{10} = H(C_7, C_8, N_{SU})$ and compares to see if C'_{10} is equal to the received C_{10}. If so, U_u is authentic. They then can compute the common session key SK= $H(C'_7+1, C'_8+2, N'_{SU}+3) = H(C_7+1, C_8+2, N_{SU}+3)$.

From the above-mentioned steps, we can see that a server spoofing attack launched by insider attacker S_i has been successfully accomplished.

U_u | S_i (S_j) | RC

1. ID_u, C_1 →

2. ID_u, SID_i, C_1 →

3. C_6 ←

4. generates a nonce N_{SU}

computes

$C_7 = C_6 \oplus Auth_{S\text{-}RC}$

$=H(H(ID_u, x), N'c)$

calculates $C_8 = C_1 \oplus C_7$

$V_2 = C_7 \oplus N_{SU}$

$C_9 = H(C_7, N_{SU}) \oplus C_8$

5. V_2, C_9 ←

6. computes $C'_7 = H(H(ID_u, x), Nc)$

retrieves $N'_{SU} = C'_7 \oplus V_2$

calculates $C'_8 = C'_7 \oplus C_1$

$C'_9 = H(C'_7, N'_{SU}) \oplus C'_8$

checks $C'_9 =? C_9$

calculates $C_{10} = H(C'_7, C'_8, N'_{SU})$

7. C_{10} →

8. computes

$C'_{10} = H(C_7, C_8, N_{SU})$

compares $C'_{10} =? C_{10}$

8. session key

$SK = H(C'_7+1, C'_8+2, N'_{SU}+3)$

session key

$SK = H(C_7+1, C_8+2, N_{SU}+3)$

Fig.5. Server spoofing attack by an insider server on Tsai's protocol:(b) the secret key has been generated.

3 Review of Liao-Wang's protocol

In this section, we review Liao-Wang's protocol. Their protocol consists of four phases: (1) registration phase, (2) login phase, (3) mutual verification and session key agreement phase, and (4) password change phase. In their protocol, y is a secret number shared among RC and all servers. We describe their protocol as follows and also depict it in Figure 6.

Registration phase

U_u | RC

1. chooses ID_u, PW_u

ID_u, PW_u ⟹

2. computes

$B=H(ID_u, x)$, $B_1=B\oplus H(ID_u, PW_u)$

$B_2=H(PW_u)\oplus H(x)$, $B_3=H(B)$

smart card contains B_1, B_2, B_3, y

⟸ smart card

Login phase

U_u | S_j

1. keys ID_u, PW_u and SID_j

computes

$B'=B_1\oplus H(ID_u,PW_u)$, $B'_3=H(B')$

If $B_3=B'_3$, generates a nonce Nc.

calculates

CID_u= H(PW_u)⊕H(B', y, Nc)

C_1 =B'⊕H(y, Nc, SID_j)

C_2 =H(B_2, y, Nc)

2. CID_u, C_1, C_2, Nc →

Mutual verification and session key agreement phase

U_u | S_j

1. computes B^*=C_1⊕H(y, Nc, SID_j),

H_{PW} =CID_u⊕H(B^*, y, Nc),

B^*_2 =H_{PW}⊕H(x), H(B^*_2, y, Nc)

checks H(B^*_2, y, Nc) =?C_2, if so,

generates a nonce Ns

calculates C_3= H(B^*_2, Nc, y, SID_j)

2. C_3, Ns ←

3. computes H(B_2, Nc, y, SID_j)

compares

H(B_2, Nc, y, SID_j)=?C_3, if so,

calculates

C_4 = H(B_2, Ns, y, SID_j)

4. C_4 →

5. computes H(B^*_2, Ns, y, SID_j)

checks H(B^*_2, Ns, y, SID_j)=?C_4

6. session key

SK= H(B^*_2, Nc, Ns, y, SID_j)

6. session key

SK= H(B_2, Nc, Ns, y, SID_j)

Fig. 6. Liao-Wang's protocol

3.1 The protocol

(1) Registration phase

In this phase, U_u performs the following steps to register at RC for obtaining a smart card so that he can access the resources of all servers.

1. Chooses his ID_u, PW_u and sends {ID_u, PW_u} to RC through a secure channel.
2. RC computes B=H(ID_u, x), $B_1=B\oplus$H(ID_u, PW_u), B_2=H(PW_u)$\oplus$H(x), and B_3=H(B). He then issues U_u a smart card containing B_1, B_2, B_3, and y through a secure channel.

(2) Login phase

1. U_u keys his ID_u, PW_u and SID_j to the smart card. The smart card computes $B'=B_1\oplus$H(ID_u, PW_u), B'_3=H(B'), and compares to see if the stored value B_3 is equal to B'_3. If so, smart card knows U_u is the real card holder. It then generates a random nonce Nc and calculates CID_u=H(PW_u)$\oplus$H(B', y, Nc), $C_1=B'\oplus$H(y, Nc, SID_j), and C_2 =H(B_2, y, Nc).
2. U_u sends {CID_u, C_1, C_2, Nc } to S_j.

(3) Mutual verification and session key agreement phase

After receiving the login message from U_u, S_j executes the following steps together with U_u to authenticate each other and compute a common session key.

1. S_j computes $B^*=C_1\oplus$H(y, Nc, SID_j), $H_{PW}=CID_u\oplus$H(B^*, y, Nc), and $B^*_2=H_{PW}\oplus$H(x). He then computes H(B^*_2, y, Nc) and checks to see if it is equal to the received C_2. If so, S_j then generates a random nonce Ns and calculates C_3= H(B^*_2, Nc, y, SID_j).
2. S_j sends {C_3, Ns} to U_u.
3. U_u computes H(B_2, Nc, y, SID_j) and compares to see if it is equal to the received C_3. If it is, S_j is authentic. U_u then calculates C_4 = H(B_2, Ns, y, SID_j).
4. U_u sends {C_4} to S_j.
5. After receiving the message from U_u, S_j computes H(B^*_2, Ns, y, SID_j) and checks

to see if it is equal to the received C_4. If so, U_u is authentic.

6. After finishing mutual authentication, U_u and S_j can compute the common session key SK= H(B_2, Nc, Ns, y, SID_j) which is equal to H(B^*_2 , Nc, Ns, y, SID_j).

(4) Password change phase

When U_u wants to change his password from PW_u to PW_u^{new}, he executes the following steps.

1. Keys his ID_u, PW_u to the smart card.
2. The smart card computes $B'=B_1\oplus H(ID_u, PW_u)$, $B'_3=H(B')$ and compares to see if B_3 in the smart card is equal to B'_3. If so, U_u is the real card holder.
3. The smart card allows U_u to submit a new password PW_u^{new}.
4. The smart card computes $B_1^{new}=B'\oplus H(ID_u, PW_u^{new})$, $B_2^{new}= B_2\oplus H(PW_u)\oplus H(PW_u^{new})$ and replaces B_1, B_2 with B_1^{new}, B_2^{new}, respectively.

3.2 Attack on Liao-Wang's protocol

In Liao-Wang's protocol, it can easily be seen that an insider peer (either a server or a user) can launch an off-line password-guessing attack by eavesdropping on the transmitted message {CID_u, C_1, C_2, Nc } and comparing C_2 with his computation of $H(H(PW') \oplus H(x), y, Nc)$, where value y stored in his smart card is shared with RC, PW' is his guessing password, and H(x) is shared by all servers which also can be derived by all legal users by computing $H(x) =B_2\oplus H(PW)$. Here, B_2 is the value stored in the smart card and PW is the user's password.

In addition, it also can be seen that anyone who has got U_u's smart card can launch a password-guessing attack by comparing B_3 with his computation result $B_1\oplus H(ID_u, PW')$. Here, B_3, B_1 are the values stored in U_u's smart card and PW' is his guessing password.

Besides, in this section, we will show two scenarios of server spoofing attack on Liao-Wang's protocol.

(1) Server spoofing attack by an insider server

Assume that S_i is a legal server who has registered at RC. He also has his secrets H(x), y to authenticate the users. We will show that S_i can masquerade as any server

(Here, without loss of generality, we assume S_i masquerades as S_j.) to cheat a remote user, because each server has the same secret data, H(x) and y, for faking messages to cheat users. We describe the server spoofing attack below and also depict it in Figure 7.

U_u		S_i (S_j)
	1. CID_u, C_1, C_2, Nc →	
		2. computes $B^* = C_1 \oplus H(y, Nc, SID_j)$ $H_{PW} = CID_u \oplus H(B^*, y, Nc)$ $B_2^* = H_{PW} \oplus H(x)$ generates a nonce Ns calculates $C_3 = H(B_2^*, Nc, y, SID_j)$
	← 3. C_3, Ns	
4. computes $H(B_2, Nc, y, SID_j)$ compares $H(B_2, Nc, y, SID_j) =? C_3$ calculates $C_4 = H(B_2, Ns, y, SID_j)$		
	5. C_4 →	
		6. computes $H(B_2^*, Ns, y, SID_j)$ checks $H(B_2^*, Ns, y, SID_j) =? C_4$
7. session key $SK = H(B_2, Nc, Ns, y, SID_j)$		7. session key $SK = H(B_2^*, Nc, Ns, y, SID_j)$

Fig. 7. Server spoofing attack by an insider server on Liao-Wang's protocol

1. U_u starts the protocol and sends $\{CID_u, C_1, C_2, Nc\}$ to S_i, where $C_1 = B' \oplus H(y, Nc, SID_j)$, as in the login phase of Figure 6.
2. After receiving the message $\{CID_u, C_1, C_2, Nc\}$ from U_u, S_i runs the mutual verification and session key agreement phase with U_u. He uses his secret data, H(x) and y, and the public parameter SID_j to compute $B^* = C_1 \oplus H(y, Nc, SID_j)$, $H_{PW} = CID_u \oplus H(B^*, y, Nc)$, and $B_2^* = H_{PW} \oplus H(x)$. He then generates a random nonce Ns and calculates $C_3 = H(B_2^*, Nc, y, SID_j)$.

3. S_i sends $\{C_3, Ns\}$ to U_u.
4. U_u computes $H(B_2, Nc, y, SID_j)$ and compares to see if it is equal to the received C_3. If so, U_u confirms that S_i is authentic. U_u then calculates $C_4 = H(B_2, Ns, y, SID_j)$.
5. U_u sends $\{C_4\}$ to S_i.
6. After obtaining the message, S_i computes $H(B_2^*, Ns, y, SID_j)$ and checks to see if it is equal to the received C_4. If so, U_u is authentic.
7. After finishing the mutual authentication, U_u and S_i can compute the common session key $SK= H(B_2, Nc, Ns, y, SID_j) = H(B_2^*, Nc, Ns, y, SID_j)$.

From the above-mentioned steps, we can see that the server spoofing attack has been successfully launched by S_i to masquerade as S_j.

(2) Server spoofing attack by an insider user

Assume that U_n is a legal user who has registered at RC. He also has a smart card to access the servers' resources. We will show that U_n can use both of the stored values B_2' and y to masquerade as any server to cheat a remote user. He can first uses B_2' and his password PW_n to compute $B_2' \oplus H(PW_n)$, obtaining $H(x)$, then uses $H(x)$ and y to fake desired messages to cheat the remote user. We describe this attack by using the following steps and also depict it in Figure 8.

U_u | $U_n(S_j)$

1. CID_u, C_1, C_2, Nc (→)

2. derives $H(x)=B_2' \oplus H(PW_n)$
computes $B^* = C_1 \oplus H(y, Nc, SID_j)$
$H_{PWu} = CID_u \oplus H(B^*, y, Nc)$
$B_2^* = H_{PWu} \oplus H(x)$
generates a nonce Ns
calculates $C_3 = H(B_2^*, Nc, y, SID_j)$

3. C_3, Ns (←)

4. computes $H(B_2, Nc, y, SID_j)$

compares H(B_2, Nc, y, SID_j)=?C_3

calculates C_4 = H(B_2, Ns, y, SID_j)

5. C_4 →

6. computes H(B_2^*, Ns, y, SID_j)

checks H(B_2^*, Ns, y, SID_j)=?C_4

7. session key

SK= H(B_2, Nc, Ns, y, SID_j)

7. session key

SK= H(B_2^*, Nc, Ns, y, SID_j)

Fig. 8. Server spoofing attack by an insider user on Liao-Wang's protocol

1. U_u starts the protocol and sends {CID_u, C_1, C_2, Nc} to U_n who impersonates S_j.
2. U_n uses his PW_n and B_2' in his smart card to derive the value of H(x) by computing $B_2' \oplus$H(PW_n). He then uses {CID_u, C_1, C_2, Nc}, H(x), y, and the public parameter SID_j to compute $B^* = C_1 \oplus$H(y, Nc, SID_j), $H_{PWu} = CID_u \oplus$H(B^*, y, Nc) and $B_2^* = H_{PWu} \oplus$H(x). In addition, he also generates a random nonce Ns and calculates C_3= H(B_2^*, Nc, y, SID_j).
3. U_n sends {C_3, Ns} to U_u.
4. After receiving the message, U_u uses his stored B_2 to compute H(B_2, Nc, y, SID_j) and compares to see if it is equal to the received C_3. If so, U_u authenticates U_n as S_j unconsciously. He then calculates C_4 = H(B_2, Ns, y, SID_j).
5. U_u sends {C_4} to U_n.
6. After obtaining the message, U_n computes H(B_2^*, Ns, y, SID_j) and checks to see if it is equal to the received C_4. If so, U_u is authentic.
7. After finishing the mutual authentication, U_u and U_n can compute the common session key SK= H(B_2, Nc, Ns, y, SID_j) = H(B_2^*, Nc, Ns, y, SID_j).

From the above-mentioned, we can see that the insider spoofing attack, launched by U_n to masquerade as S_j, has been accomplished successfully.

4. Review Li et al.'s protocol

In 2013, Li et al. [16] also proposed a multi-server protocol to enhance Lee et al.'s scheme [14] whose security weakness of suffering an insider server attack had been identified by *Chou et al.* [15]. They claimed that their protocol is secure. However after examining the protocol, we found it sufferers the smart card lost password-guessing attack if the lost the smart card is obtained by an insider user. We depict the original scheme in figure 10. In the following, and only demonstrate the attack. The details of the protocol can be referred to [16].

4.1 Attack on the protocol

This protocol suffers the smart card lost password-guessing attack launched by an insider, because from both the smart cards, his own and the lost, and from message 3, an insider user who has the value of *h(y)* can obtain the value N_i, and subsequently obtain E_i. Then, from the parameter D_i stored in the lost smart card, and CID_i in the

U_i (Secure channel) RC

Registration 1.ID_i ,$h(b \oplus PW_i)$ →

2.Computes T_i,V_i,B_i,and H_i

3.Smart card (V_i,B_i,H_i,h(.),h(y)) ←

4.Keys b into the smart card

U_i (Public channel) S_j

Login 1.Inserts smart card, and inputs ID_i, PW_i

Phase $T_i = V_i \oplus h(ID_i \| h(b \oplus PW_i))$, $H_i^* = h(T_i)$

Checks $H_i^* ?= H_i$

2.Computes $A_i = h(T_i \| h(y) \| N_i)$

$CID_i = h(b \oplus PW_i) \oplus h(T_i \| A_i \| N_i)$

$P_{ij} = T_i \oplus h(h(y) \| N_i \| SID_j)$, $Q_i = h(B_i \| A \| N_i)$

3.{ CID_i,P_{ij},Q_i,N_i } →

Verification 1.Computes

$T_i = P_{ij} \oplus h(h(y) \| N_i \| SID_j)$

Phase $A_i = h(T_i \| h(y) \| N_i)$

$H(b \oplus PW_i) = CID_i \oplus h(T_i \| A_i \| N_i)$

$B_i = h(h(b \oplus PW_i) \| h(x \| y))$

2.Check $h(B_i \| A_i \| N_i) ?= Q_i$

Generates a nonce N_j

$M'_{ij} = h(B_i \| N_i \| A_i \| SID_j)$

← $\{M'_{ij}, N_j\}$

3.Checks $h(B_i \| N_i \| A_i \| SID_j) ?= M'_{ij}$

$M'_{ij} = h(B_i \| Nj \| A_i \| SID_j)$

$\{M''_{ij}\}$ →

4.Checks $h(B_i \| N_j \| A_i \| SID_j)?=M''_{ij}$

$SK=h(B_i \| N_i \| N_j \| A_i \| SID_j)$ ←

Figure 10 Li et al.'s protocol

transferred message 3, he can obtain A_i. Then from value b stored in the lost the smart card and this A_i, he can guess the password as *psw* and check to see whether A_i is equal to $h(b \oplus psw)$. If so, he gets the right password. In addition, in their scheme the server cannot know the identity of the user which is somewhat impractical. Moreover, if a user collide with a server to get the values of $h(y)$ *and* $h(x \| y)$, their scheme is totally infeasible.

4.2 Improvement on the protocol

The key point of the smart card lost password-guessing attack is resulted in from the transferred M_2 in which N_i can be easily calculated by an insider user. To fix this problem, we must reconstruct some part of the original phases in the scheme. We first reconstruct the Registration phase. Then in the following two phases, Login phase and Verification phase, all the values *ys* in the original scheme are replaced with y_js. We only list the modifications needed to improvement the original scheme in these two phases, avoiding N_i be easily calculated when the user's smart card is los. The other part if not mentioned are kept unchanged. We describe them as follows and also depict it in Figure 11.

U_i (Secure channel) RC

Registration 3. chooses ID_i , PW_i 1. Chooses Y_j for each S_j

Phase 4. computes 2. computes

$A_i=h(b \oplus PW_i)$ $h(x \| y_j)$, $h(SID_j \| y_j)$

$C_1=h(ID_i \| h(x) \| A_i)$

5.Generate a nonce number b

6. ID_i , A_i → 7. Generate a nonce number r_j

computes

$hr_j=h(SID_j \| h(y_j) \oplus h(SID_j \| h(x \| y_j) \| r_j)$

$V_j = hr_j \oplus A_i$

Smart Card $= \{ b, h(.), C_1, r_m, V_m, \}$ ←

U_i (Public channel) S_j

Login Phase

1.Inserts smart card, and inputs ID_i, PW_i

2. Smart card generates a random number N_i

3.computes

$F = N_i \oplus A_i$,

$CID_i = h(N_i || A_i) \oplus ID_i$,

$M_1 = h(F || CID_i || V_j || N_i)$

4. r_j , V_j , F , CID_i , M_1 →

Verification Phase

5.Computes

$hr_j{}' = h(SID_j || h(y_j)) \oplus h(SID_j || h(x|| y_j)|| r_j)$

$A_i{}' = V_j \oplus hr_j{}'$

$N_i{}' = F \oplus A_i{}'$

$ID_i{}' = h(N_i{}' || A_i{}') \oplus CID_i$

$M_1{}' = h(F|| CID_i || V_j || N_i{}')$

6.Check $M_1 =? M_1{}'$

7.generate a nonce number N_s

8.computes

$R_j = h(SID_j || h(y_j)) \oplus h(SID_j || h(x|| y_j)|| N_i{}') \oplus N_s$

$M_2 = h(hr_j{}' || N_i{}' || SID_j)$

$M_3 = h(F|| A_i || M_2 || N_i{}' || SID_j) \oplus N_s$

← R_j , M_2 , M_3

session key SK= $h(A_i{}' || N_i{}' || N_s || SID_j)$

9.computes

$M_2{}' = h(hr_j || N_i || SID_j$

Check $M_2 = ? M_2$

Computes

$N_s{}' = M_3 \oplus h(F|| A_i || M_2 || N_i || SID_j)$

session key SK= $h(A_i|| N_i || N_s{}' || SID_j)$

/* for the next time login */

computes $r_j = N_i \oplus N_s{'}$, $hr_j = R_j \oplus N_s{'}$,
and $V_j = hr_j \oplus A_i$

Fig.11. the proposed improvement

(1) Registration phase

In this phase, RC chooses a secret number y_j for each server S_j. and computes $h(x|| y_j)$ and $h(SID_j|| y_j)$, where x is RC's master secret key. It then shares them with S_j via a secure channel. In each user's smart card, there are two little arrays V_m and r_m, where m is the number of servers, and $1 \leqq j \leqq m$. U_i freely chooses his/her identity ID_i, the password PW_i, and computes $A_i = h(b \oplus PW_i)$ and $C_1 = h(ID_i || h(x) || A_i)$. Here, b is a random number generated by U_i. Then, U_i sends ID_i and A_i to RC for registration through a secure channel. RC chooses a random number r_j and computes $hr_j = h(SID_j || h(y_j)) \oplus h(SID_j || h(x|| y_j) || r_j)$, and $V_j = hr_j \oplus A_i$, for each server j. It then stores { b, $h()$,C_1, r_m, V_m, } in the user's smart card.

(2) Login phase

The user inserts smart card and inputs ID_i and PW_i. Smart card generates a random number N_i and computes parameters $F = N_i \oplus A_i$, $CID_i = h(N_i || A_i) \oplus ID_i$, and $M_1 = h(F|| CID_i || V_j || N_i)$. It then sends r_j, V_j, F, CID_i, and M_1 to S_j.

(3) Verification phase

After receiving the message, S_j computes $hr_j{'} = h(SID_j || h(y_j)) \oplus h(SID_j || h(x|| y_j) || r_j)$, $A_i{'} = V_j \oplus hr_j{'}$, $N_i{'} = F \oplus A_i{'}$, $ID_i{'} = h(N_i{'} || A_i{'}) \oplus CID_i$, and $M_1{'} = h(F|| CID_i || V_j || N_i{'})$. S_j then compares the received M_1 with $M_1{'}$. If they are equal, S_j authenticates U_i successfully. It then computes the session key as $h(A_i{'} || N_i{'} || N_s || SID_j)$ and generates a random number N_s. Then it computes $R_j = h(SID_j || h(y_j)) \oplus h(SID_j || h(x|| y_j) || N_i{'} \oplus N_s) \oplus N_s$, $M_2 = h(hr_j{'} || N_i{'} || SID_j)$, and $M_3 = h(F|| A_i || M_2 || N_i{'} || SID_j) \oplus N_s$ and sends them to the smart card. After receiving the message, the smart card computes $hr_j = A_i \oplus V_j$, $M_2{'} = h(hr_j || N_i || SID_j)$. It then compares the received M_2 with this calculated value $M_2{'}$. If they are equal, U_i authenticates S_j successfully. The smart card then computes $N_s{'} = M_3 \oplus h(F|| A_i || M_2 || N_i || SID_j)$ and the session key as $h(A_i|| N_i || N_s{'} || SID_j)$. For the next time login, U_i computes $r_j = N_i \oplus N_s{'}$, $hr_j = R_j \oplus N_s{'}$, and $V_j = hr_j \oplus A_i$.

(4) Password change phase

This phase is the same as the original one except for the value $h(y)$ in C_i should be replaced with $h(x)$.

4.3 security analysis

In this section, we discuss the security features of the proposed improvement according to the features is defined in [16].

(1) known-key secrecy

In our scheme, the session key is $h(A_i|| N_i || N_s^{'} || SID_j)$. If the attacker get a previous session key, he cannot get the other session keys, because he doesn't know the parameters A_i, N_i, and N_s.

(2) forward secrecy

If the master secret key x of the system is compromised, the secrecy of previously established session keys should not be affected. Since the session key in our scheme is $h(A_i|| N_i || N_s^{'} || SID_j)$, it has no relationship with the value x. Therefore this security feature is assured.

(3) resist replay attack

In our improvement, each session's transcript is identified by the session's random variables, N_i and N_s. That is, all the transmitted parameters are randomised and different from other sessions. More clearly, if an attacker lunches such an attack, due to lack of the knowledge of A_i, he cannot obtain the session key. Therefore this attack fails.

(4) resist forgery attack

If an attacker lunches such an attack, he must be able to forge the login request to fool the server. However, without the knowledge of A_i and V_j, the attacker can not make a valid login request. Beside, in the attacker got the smart card and extracted the parameters stored in the smart card, he cannot also forge a login request to the server, because he cannot use the stored parameters to compute A_i without the knowledge of password. Therefore this attack fails

(5) resist server spoofing attack and the registration center spoofing attack

On the server's spoofing attack, if the attacker is an insider user, he must be able to forge a valid response message R_j= $h(SID_j || h(y_j) \oplus h(SID_j || h(x|| y_j)|| N_i^{'}) \oplus N_s$, M_2=$h(hr_j^{'} || N_i^{'} || SID_j)$, and M_3= $h(F|| A_i || M_2 || N_i^{'} || SID_j) \oplus N_s$. However the attacker cannot compute $h(x|| y_j)$, $hr_j^{'}$, N_i, $h(y_j)$and N_s from his smart card. If the attacker is an insider server, he also can not spoof at another server to fool and legal user, because he doesn't have the other server's secret $h(y_j)$ and $h(x|| y_j)$ to compute N_i and A_i to produce valid response message. Therefore this attack fails

(6) resistance to stolen smart card password guessing attacks

Even the smart car has been stolen, to change the user's password or log into the system by using this is lost smart card, the attacker cannot determine whether the password guessed is right or not, because A_i is not stored in the smart card.

(7) proper mutual authentication

In this improvement, the user sends the message r_j, V_j, F, CID_i, and M_1 to S_j. After receiving this message, S_j computes $hr_j' = h(SID_j \,||\, h(y_j) \oplus h(SID_j \,||\, h(x|| y_j) || r_j)$, $A_i' = V_j \oplus hr_j'$, $N_i' = F \oplus A_i'$, $ID_i' = h(N_i' \,||\, A_i') \oplus CID_i$, and $M_1' = h(F || CID_i \,||\, V_j \,||\, N_i')$. S_j then compares the received M_1 with M_1'. If they are equal, S_j authenticates U_i successfully. Any fabricated message cannot pass the verification of M_1. Similarly, any forged message $R_j = h(SID_j \,||\, h(y_j) \oplus h(SID_j \,||\, h(x|| y_j) || N_i') \oplus N_s$, $M_2 = h(hr_j' \,||\, N_i' \,||\, SID_j)$, and $M_3 = h(F || A_i \,||\, M_2 \,||\, N_i' \,||\, SID_j) \oplus N_s$ can not pass the user's authentication. Therefore our improvements provide proper mutual authentication.

From the above security analysis, we come confirms that our improvements outperforms [16] in the security feature of lost smart card password guessing attack.

4. Conclusion

We have analyzed the security of Tsai et al.'s, Liao-Wang's et al.'s, and Li et al.'s protocols and found that they are indeed insecure against several attacks that we have described in this article. After that, based on Li et al.'s protocol, we propose a novel multi-server authentication protocol which not only outperforms the original protocol in the security feature of avoiding lost smart card password-guessing attack but also is more efficient than theirs, because our improvement only composed of the hash and exclusive-or operations and required only two passes

Reference

[1]J.L. Tsai, "Efficient multi-server authentication scheme based on one-way hash function without verification table", *Computers & Security*, Vol. 27, No. 3-4, pp. 115-121, May-June 2008.

[2]Y.P. Liao, S.S. Wang, "A secure dynamic ID based remote user authentication scheme for multi-server environment", *Computer Standards & Interfaces*, Vol. 31,

No. 1, pp. 24-29, January 2009.

[3]W.J. Tsaur, C.C. Wu, W.B. Lee, “An enhanced user authentication scheme for multi-server Internet services”, *Applied Mathematics and Computation*, Vol. 170, No. 1-1, pp. 258-266, November 2005.

[4]W.J. Tsaur, C.C. Wu, W.B. Lee, “A smart card-based remote scheme for password authentication in multi-server Internet services”, *Computer Standards & Interfaces*, Vol. 27, No. 1, pp. 39-51, November 2004.

[5]I.C. Lin, M.S. Hwang, L.H. Li, “A new remote user authentication scheme for multi-server architecture”, *Future Generation Computer Systems*, Vol. 19, No. 1, pp. 13-22, January 2003.

[6]J. H. Lee, D. H. Lee, “Efficient and Secure Remote Authenticated Key Agreement Scheme for Multi-server Using Mobile Equipment”, *Proceedings of International Conference on Consumer Electronics*, pp. 1-2, January 2008.

[7]L. Hu, X. Niu, Y. Yang, “An Efficient Multi-server Password Authenticated Key Agreement Scheme Using Smart Cards”, *Proceedings of International Conference on Multimedia and Ubiquitous Engineering*, pp. 903-907, April 2007.

[8]X. Cao, S. Zhong, “Breaking a remote user authentication scheme for multi-server architecture”, *IEEE Communications Letters*, Vol. 10, No. 8, pp. 580-581, August 2006.

[9]Z.F. Cao, D.Z. Sun, “Cryptanalysis and Improvement of User Authentication Scheme using Smart Cards for Multi-Server Environments”, *Proceedings of International Conference on Machine Learning and Cybernetics*, pp. 2818-2822, August 2006.

[10]C.C. Chang, J.Y. Kuo, “An efficient multi-server password authenticated key agreement scheme using smart cards with access control”, *Proceedings of International Conference on Advanced Information Networking and Applications*, Vol. 2, No. 28-30, pp. 257-260, March 2005.

[11]R.J. Hwang, S.H. Shiau, “Password authenticated key agreement protocol for multi-servers architecture”, *Proceedings of International Conference on Wireless Networks*, Vol. 1, No. 13-16, pp. 279-284, June 2005.

[12]C.C. Chang, J.S. Lee, “An efficient and secure multi-server password authentication scheme using smart cards”, *Proceedings of International Conference on Cyberworlds*, No. 18-20, pp. 417-422, November 2004.

[13]W.S. Juang, “Efficient multi-server password authenticated key agreement using smart cards”, *IEEE Transactions on Consumer Electronics*, Vol. 50, No. 1, pp. 251-255, February 2004.

[14] C.C. Lee, T.H. Lin, R.X. Chang, “A secure dynamic ID based remote user authentication scheme for multi-server environment using smart cards,” *Expert*

Systems with Applications 38 (2011) 13863–13870

[15] J.S. Chou, Y. Chen, C.H. Huang, Y.S. Huang, “Comments on four multi-server authentication protocols using smart card”, *http://eprint.iacr.org/2012/406*

[16]X. Li, J. Ma, W. Wang, Y. Xiong, and J.Zhang, “A novel smart card and the dynamic ID based remote user authentication scheme for multi-server environments”, *Mathematical and Computer Modelling*, Vol. 58, Issues 1-2, July 2013, Pages 85-95 in the world

[17] J.S. Chou, C.H. Huang, Y. Chen, “Cryptanalysis on two multi-server password based authentication protocols,” *International Journal of Computer Science and Information Security*, Vol. 8, No. 2, pp. 16-20, MAY 2010.