JÖKULL JOURNAL
JOKULL ISSN: 0449-0576
Vol 63, No. 7;Jul 2013

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 that their protocols were secure, and that they could withstand various attacks. However, we found some security loopholes in each of their schemes. For example, the schemes of both Tsai *et al*. and Liao *et al.* are vulnerable to a server spoofing attack by an insider server while that of Li *et al.* is exposed to to the lost smart card password-guessing attack. In addition, the scheme of Liao *et al.* is vulnerable to the off-line password-guessing attack. In this study, we review and demonstrate the effects of these attacks on each scheme. Then based on the scheme of Li *et al.*, we developed a novel method and examined its security using several features. The security analysis confirmed that our protocol outperformed the scheme of Li *et al.* in terms of its security features when subjected to the lost smart card password-guessing attack.

Keywords: *multi-server, password authentication protocol, server-spoofing attack, parallel session attack*

1. Introduction

The two-party password authentication protocol used for client-server architectures is often insufficient when the network increases in size. Thus, several multi-server protocols have been proposed [1-16] to address this problem.

In 2003, Li *et al*. [5] proposed a multi-server protocol based on the ElGamal digital signature and geometric transformations on an Euclidean plane. Unfortunately, their protocol was vulnerable, and it 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 were based on a Lagrange interpolation polynomial, which is computationally intensive, and they 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 environments. Both schemes assumed 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-servers using mobile equipment. However, their scheme did not allow a server to be added freely, because when a brand new server was added, all the users who wanted to login to this brand new server had to re-register at the registration center to obtain a new smart card. This would increase the registration center's card issue cost. In 2008, Tsai [1] also proposed an efficient multi-server authentication scheme, and he claimed that his protocol could withstand seven known attacks. After analysis, however, we found that it was vulnerable to a server spoofing attack**.** In 2009, Liao and Wang [2] proposed a secure dynamic ID scheme for multi-server environments. They claimed that their protocol was safe and sound but we found that their scheme was vulnerable to server spoofing attacks.

In the last two years, new studies [19-22] have highlighted the security flaws of previous schemes (as stated above) but also proposed a secure protocol for multi-server environment. Lee *et al.* [19] and Sood *et al.* [20] both found the improved method in Hsiang et al. [14] was still insecure. In 2011, Tsaura *et al.* [21] pointed out that [13] was in danger to a man-in-the-middle attack, while Li *et al.* [22] indicated that [20] was at risk to the leak-of-verifier attack and the stolen smart card attack in 2012. Finally, also in 2012, Hwang *et al.* [18] proposed an improved multi-server authentication protocol based on bilinear pairings, while Liao *et al.* [23] proposed a novel multi-server authentication scheme for mobile clients. However, Chou *et al.* [15] found that there were still weaknesses in each method [19-23].
In 2013, Li *et al*. [16] also proposed a novel multi-server scheme and claimed that their scheme was secure. However, we found it was vulnerable to the smart card lost password-guessing attack. In this study, we first demonstrate the attacks on [1] and [2]. We then demonstrate the attack on [16] and propose a novel method based upon this scheme. Security analysis confirmed that our scheme resisted the lost password-guessing attack and was more efficient than [16] in terms of the protocol's number of passes.

The remainder of this paper is organized as follows. In Sections 2 and 3, we review and demonstrate the attacks on Tsai's protocol and Liao-Wang's protocol, respectively. Section 4 first demonstrates the attacks on Li *et al.*'s protocol, then describes our novel method and presents its security analysis. Finally, our conclusions are given in Section. 5.

2. Review of Tsai's protocol

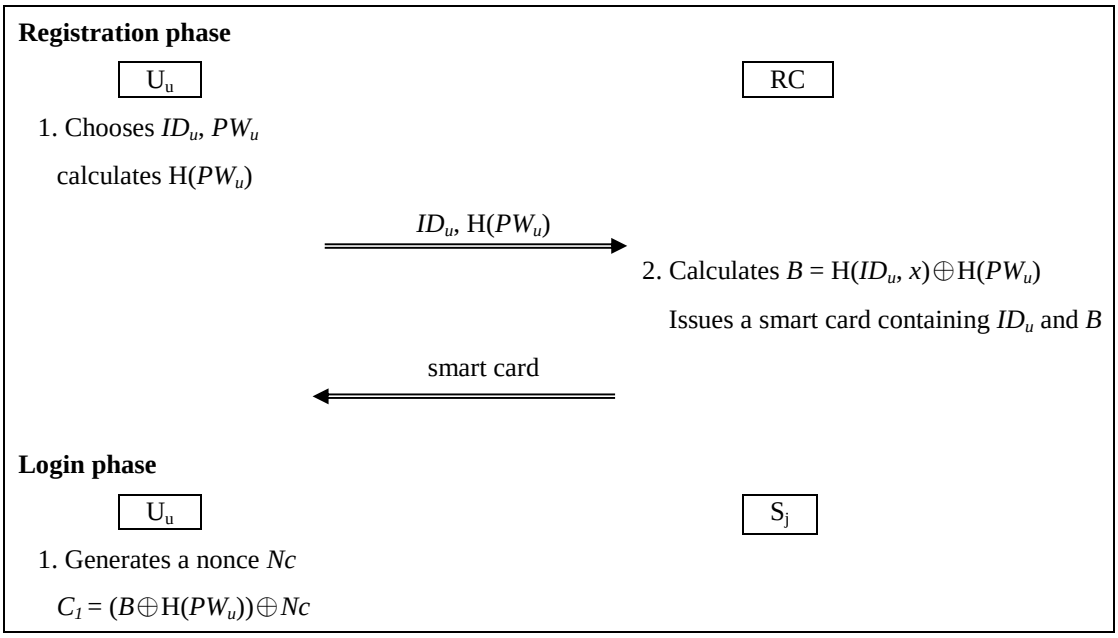
In this section, we review Tsai's protocol (Section 2.1) and examine its security (Section 2.2). First, we define the notations used throughout this paper, as follows.

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 that 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 which denotes that string *a* is concatenated with string *b*
ΔT: a tolerant time delay for messages transmission over network
$\rightarrow$: a common channel

2.1 The protocol

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

We assume that there are *s* servers on the system. Initially, RC computes and sends $H(SID_j,y)$ to S_j, where S_j keeps it secret for j = 1 to *s* via a secure channel.



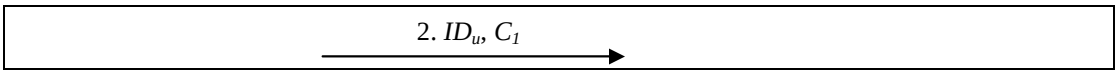


Fig. 1. The registration phase and login phase of Tsai's protocol

(1) Registration phase

In this phase, U_u performs the following steps to obtain 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 via a secure channel.
2. RC calculates $B = H(ID_u, x) \oplus H(PW_u)$ and issues U_u with a smart card that contains ID_u and B, which also occurs via a secure channel.

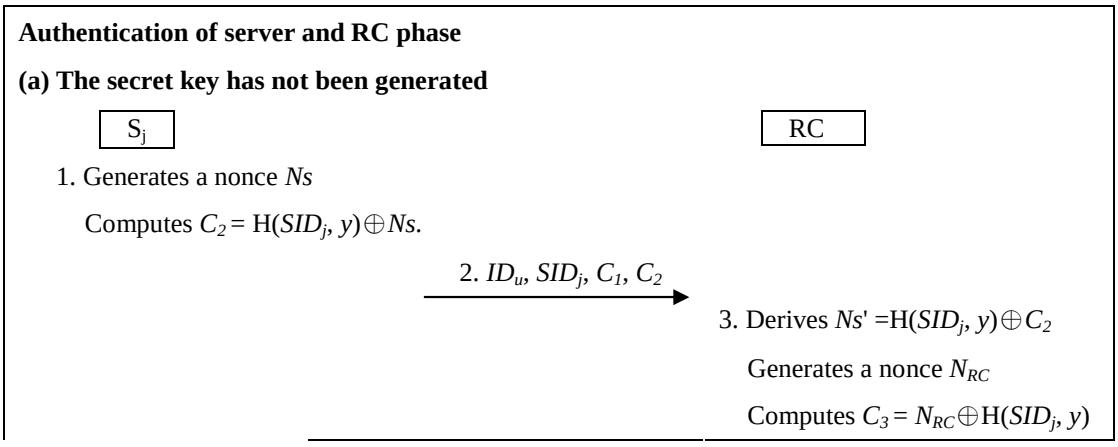
(2) Login phase

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, after receiving the message $\{ID_u, C_1\}$ from U_u, S_j will run the following steps to allow himself to be authenticated by RC, to verify U_u's legitimacy, and to 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 has not been generated, and (b) the secret key has been generated. We describe these scenarios below.



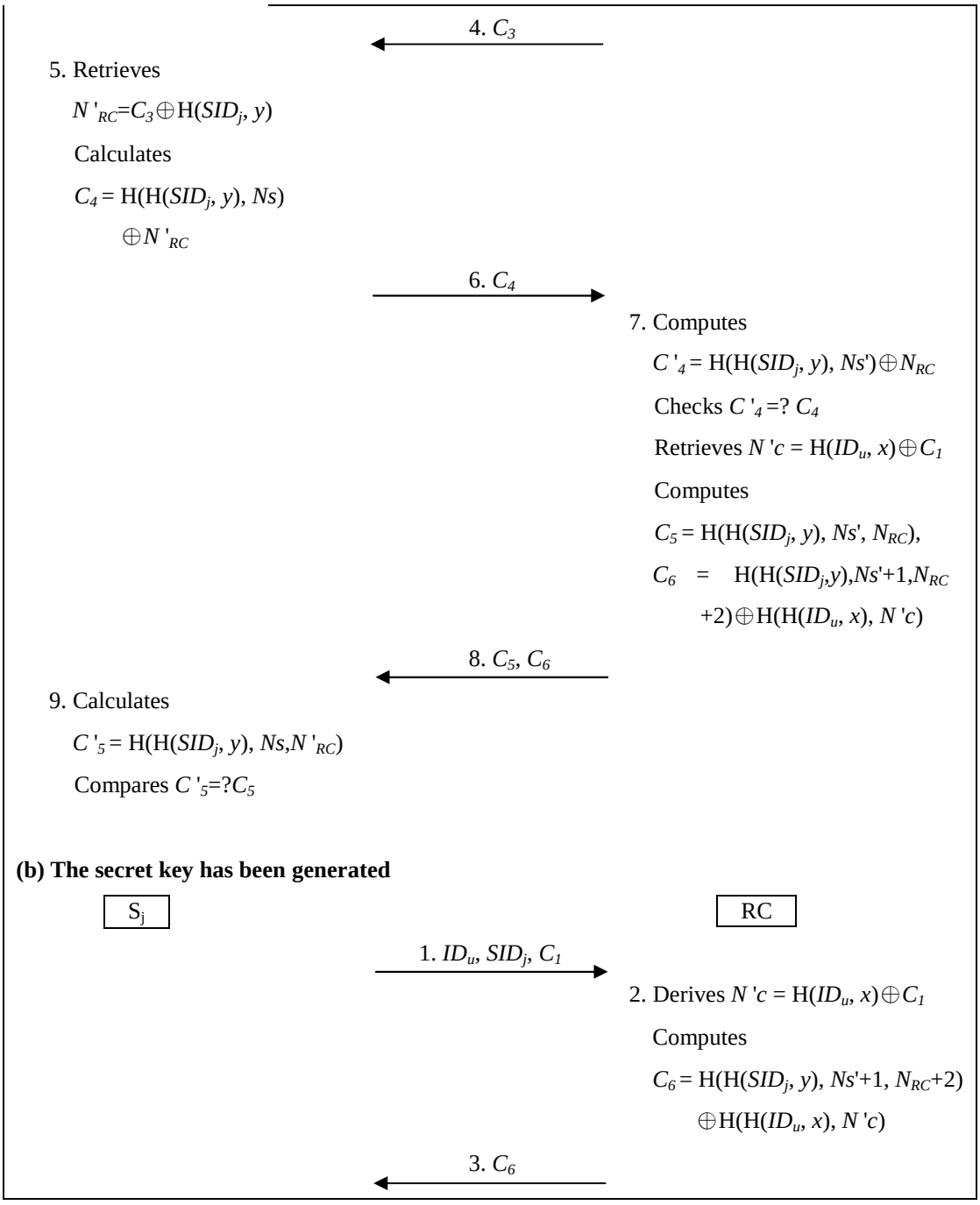


Fig. 2. Authentication of server and RC phase of Tsai's protocol

(a) The secret key has not been generated

1. S_j generates a random nonce Ns and computes $C_2 = H(SID_j, y) \oplus Ns$.
2. S_j sends $\{ID_u, SID_j, C_1, C_2\}$ to RC.
3. RC derives $Ns' = H(SID_j, y) \oplus C_2$. He then generates a random nonce N_{RC} and computes $C_3 = N_{RC} \oplus H(SID_j, y)$.
4. RC sends $\{C_3\}$ to S_j.

5. After receiving the message from RC, S_j retrieves $N'_{RC} = C_3 \oplus H(SID_j, y)$ and calculates $C_4 = H(H(SID_j, y), Ns) \oplus N'_{RC}$.
6. S_j sends $\{C_4\}$ to RC.
7. RC computes $C'_4 = H(H(SID_j, y), Ns') \oplus N_{RC}$ and checks to determine whether C'_4 is equal to the received C_4. If this is the case, 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 performs a comparison to determine whether C'_5 is equal to the received C_5. If this is the case, RC is authentic. Both S_j and RC store the common secret key $Auth_{S\text{-}RC} = H(H(SID_j, y), Ns+1, N'_{RC}+2)$ for the next execution of this authentication procedure, the authentication of the 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 to facilitate mutual authentication.

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	

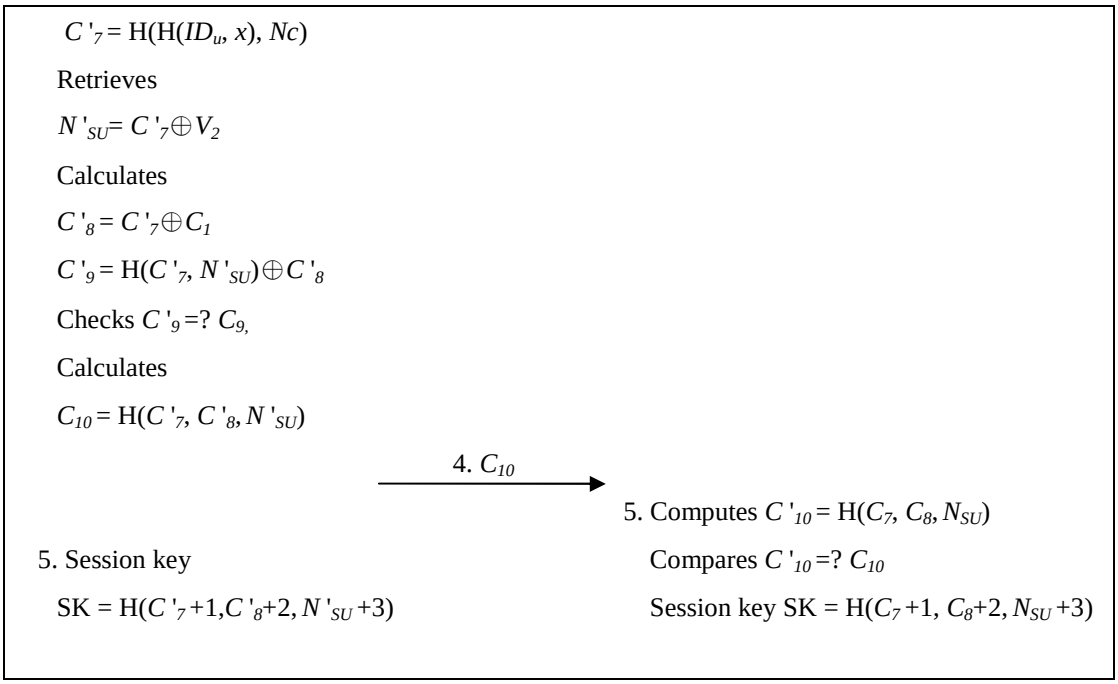


Fig. 3. Authentication of server and user phase of Tsai's protocol

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 determine whether the computed C'_9 is equal to the received C_9. If this is the case, 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 $\{C_{10}\}$, Sj computes $C'_{10} = H(C_7, C_8, N_{SU})$ and performs a comparison to determine whether C'_{10} is equal to the received C_{10}. If this is the case, U_u is authentic. They then have the same session key, $SK = H(C'_7 +1, C'_8+2, N'_{SU} +3) = H(C_7 +1, C_8+2, N_{SU} +3)$.

2.2 Attack on Tsai's protocol

Our analysis showed that Tsai's protocol was vulnerable to a server-registered spoofing attack in both scenarios. We demonstrate the security loopholes in the following sections.

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

We assume that S_i is a legal server at RC. The attacker 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 a user does not check whether the message was actually sent from the correct server during the authentication of server and user phase. Next, we demonstrate the server spoofing attacks in the two aforementioned scenarios: (1) where the secret key has not been generated, and (2) where the secret key has been generated, while we also illustrate them in Figures 4 and 5, respectively.

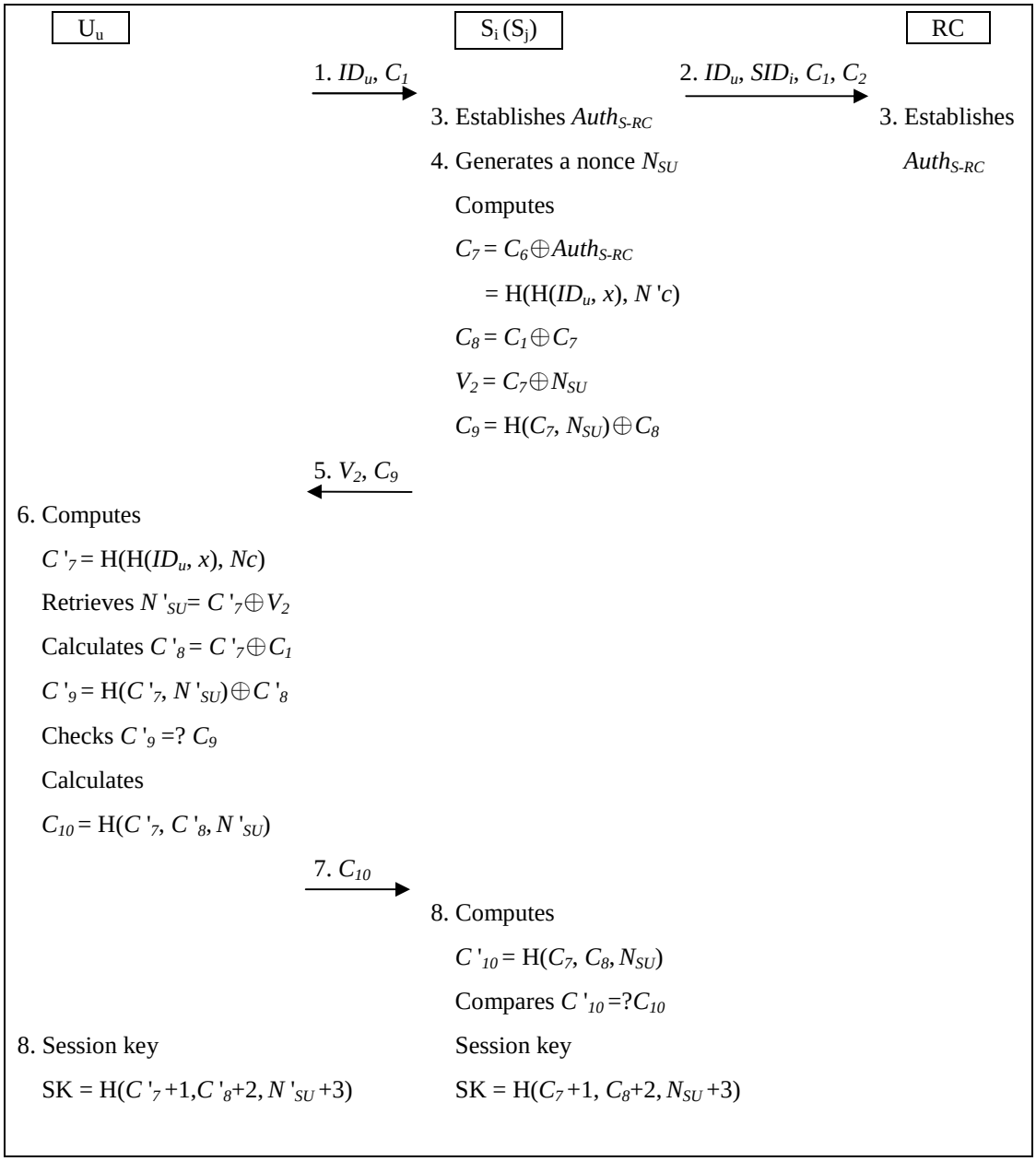


Fig. 4. Server spoofing attack by an insider server on Tsai's protocol: (a) the secret key has not been generated

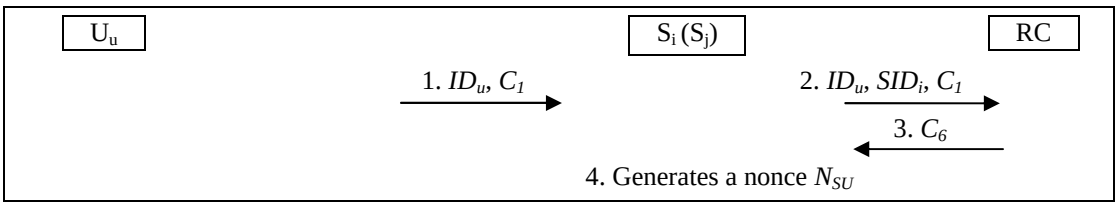
(1) The secret key has not been generated

1. If 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. 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 of U_u's secrecy $H(H(ID_u, x), Nc)$, as shown in scenario (a) in Figure 2. Thus, RC and S_i will be able to achieve mutual authentication successfully.
3. RC and S_i then negotiate to establish the common secret key $Auth_{S\text{-}RC}=H(H(SID_i, y), Ns+1, N'_{RC}+2)=H(H(SID_i, y), Ns'+1, N_{RC}+2)$ during the server and RC authentication phase. Next, S_i and U_u perform the following steps during 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 determine whether C'_9 is equal to the received C_9. If this is the case, U_u confirms that the message is from the server who received his C_1 during 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 performs a comparison to determine whether C'_{10} is equal to the received C_{10}. If this is the case, U_u is authentic. Next, they compute the common session key $SK = H(C'_7+1, C'_8+2, N'_{SU}+3) = H(C_7+1, C_8+2, N_{SU}+3)$.

These steps show that a server spoofing attack can be launched successfully by the insider attacker S_i in this case.

(2) The secret key has been generated

In this case, we describe the attack as follows, which is also illustrated in Figure 5.



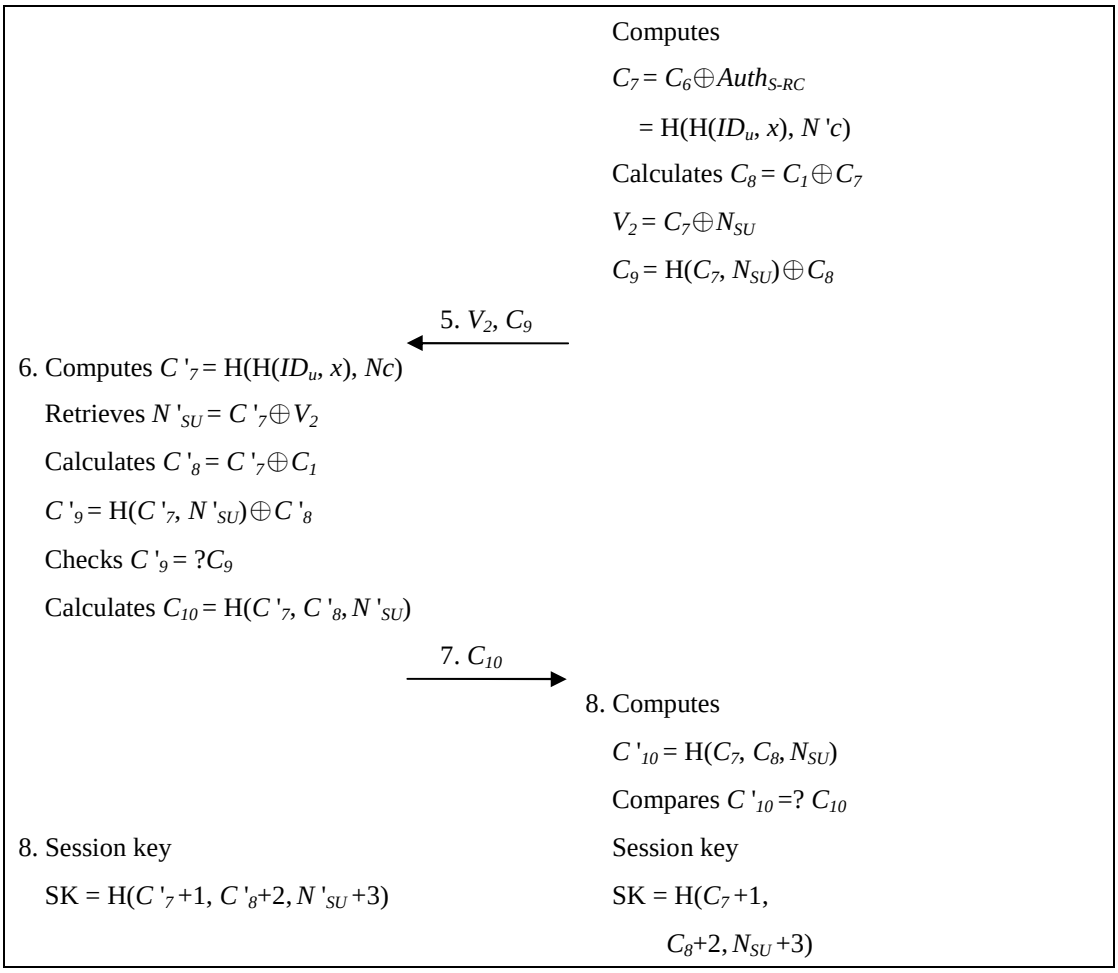


Fig. 5. Server spoofing attack by an insider server on Tsai's protocol: (b) the secret key has been generated

1. U_u starts the protocol and sends $\{ID_u, C_1\}$ to S_i who masquerades as S_j.
2. If S_i runs the authentication of server and RC phase, he simply sends $\{ID_u, SID_i, C_1\}$ to RC. RC deduces $N'c = H(ID_u, x) \oplus C_1$ and computes $C_6 = H(H(SID_i, y), Ns'+1, N_{RC}+2) \oplus H(H(ID_u, x), N'c)$.
3. RC sends $\{C_6\}$ to S_i, as shown in scenario (b) in Figure 2. S_i then continues the following steps with U_u during 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 $C_7 = C_6 \oplus Auth_{S\text{-}RC} = H(H(ID_u, x), N'c)$. Next, he 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 determine whether C'_9 is equal to the received C_9. If this is the case, U_u confirms that the message was sent from the correct server who received his C_1 during the login phase, and S_i disguising as S_j is regarded as 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 performs a comparison to determine whether C'_{10} is equal to the received C_{10}. If this is the case, U_u is authentic. They can then compute the common session key $SK = H(C'_7+1, C'_8+2, N'_{SU}+3) = H(C_7+1, C_8+2, N_{SU}+3)$.

These steps demonstrate that the server spoofing attack launched by insider attacker S_i was successful in this case.

3. Review of Liao-Wang's protocol

In this section, we review Liao-Wang's protocol, which has four phases: (1) registration phase, (2) login phase, (3) mutual verification and session key agreement phase, and (4) password change phase. In this protocol, *y* is a secret number shared among RC and all servers. We describe their protocol in the following section, and it is also illustrated 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 $CID_u = H(PW_u) \oplus H(B', y, Nc)$ $C_1 = B' \oplus H(y, Nc, SID_j)$ $C_2 = H(B_2, y, Nc)$		

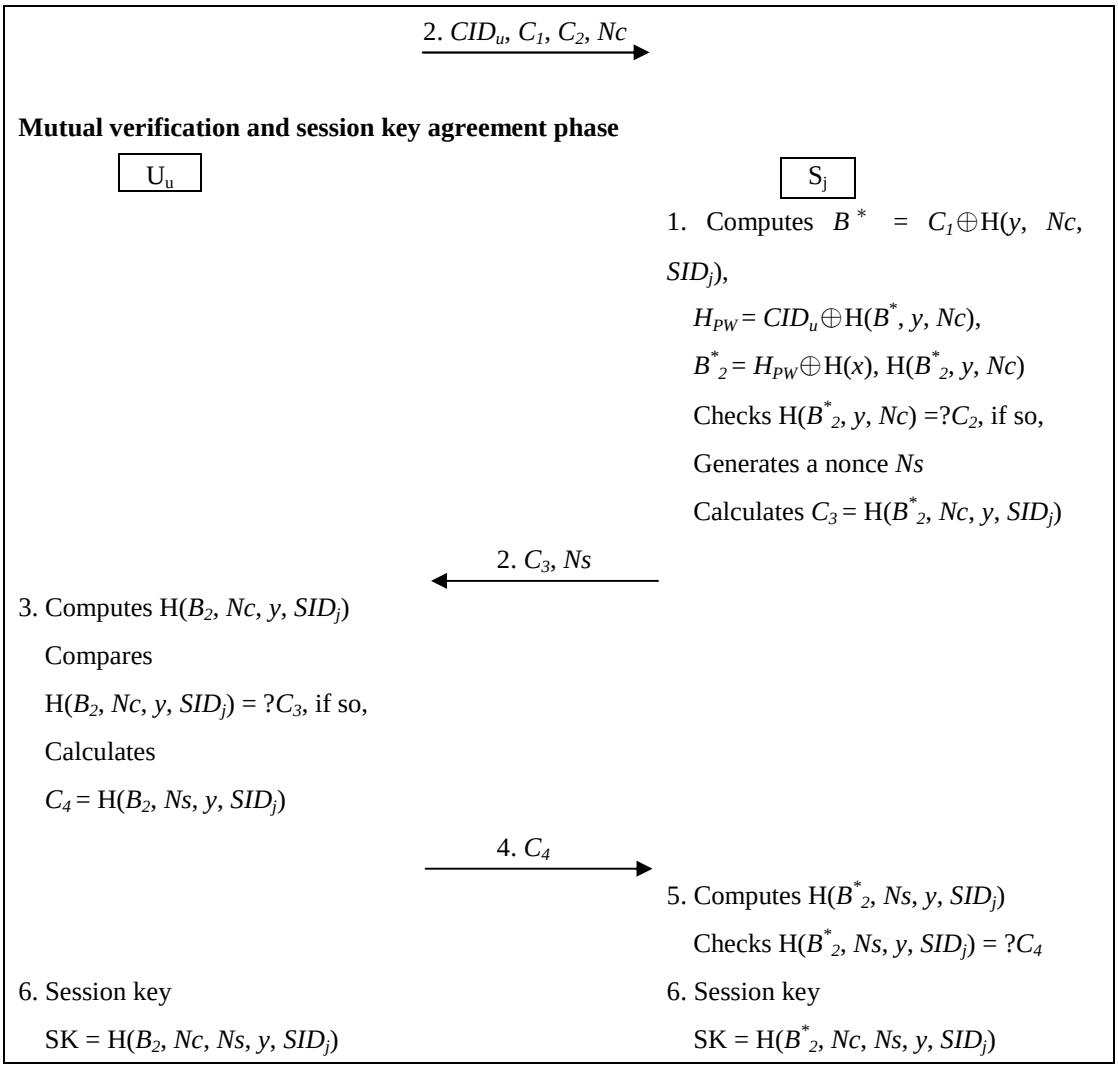


Fig. 6. Liao-Wang's protocol

3.1 The protocol

(1) Registration phase

During this phase, U_u performs the following steps to register at RC and obtains a smart card, so he can access the resources on all the servers.

1. U_u selects his ID_u and PW_u, and sends $\{ID_u, PW_u\}$ to RC via 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 that contains B_1, B_2, B_3, and y via 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 performs a comparison to determine whether the stored value of B_3 is equal to B'_3. If this is the case, the smart card knows that 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 with U_u so they can 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, whether it is equal to the received C_2. If this is the case, S_j afterwards 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 performs a comparison to determine whether it is equal to the received C_3. If this is the case, 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, whether it is equal to the received C_4. If this is the case, U_u is authentic.
6. After finishing the 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

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

1. U_u keys his ID_u and PW_u into the smart card.
2. The smart card computes $B' = B_1 \oplus H(ID_u, PW_u)$, $B'_3 = H(B')$ and performs a comparison to determine whether the value of B_3 in the smart card is equal to B'_3. If this is the case, 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 and B_2 with B_1^{ne} and B_2^{new}, respectively.

3.2 Attack on Liao-Wang's protocol

In Liao-Wang's protocol, an insider peer (either a server or a user) can easily 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 the 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 can also be derived by all legal users by computing H(x) =$B_2 \oplus$H(PW), where B_2 is the value stored in the smart card and PW is the user's password.

Anyone who possesses U_u's smart card can also launch a password-guessing attack by comparing B_3 with his computation result $B_1 \oplus$H(ID_u, PW ')., where B_3 and B_1 are the values stored in U_u's smart card and PW ' is his guessing password.

In this section, we also present two scenarios for the server spoofing attack on Liao-Wang's protocol.

(1) Server spoofing attack by an insider server

We assume that S_i is a legal server who has registered at RC. He also has his secrets H(x), y for authenticating users. We show that S_i can masquerade as any server (in this study, without any loss of generality, we assume that 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 it is also illustrated in Figure 7.

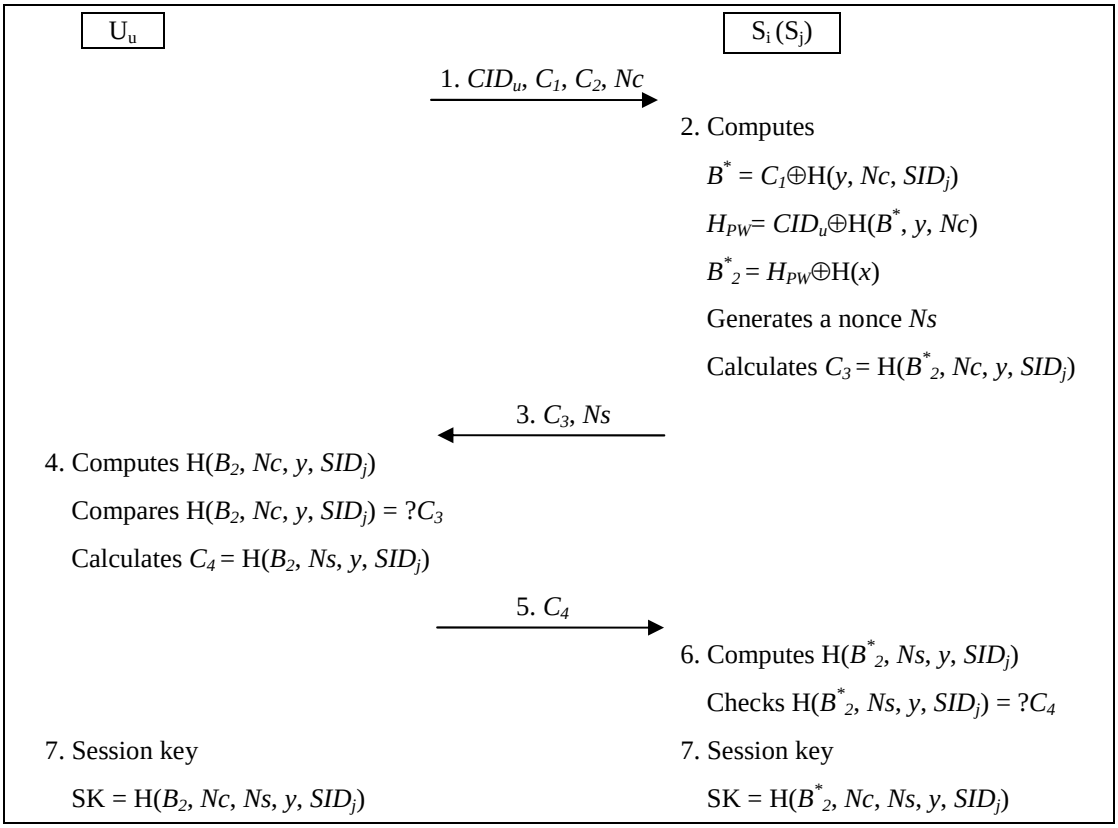


Fig. 7. Server spoofing attack by an insider server on Liao-Wang's protocol

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,

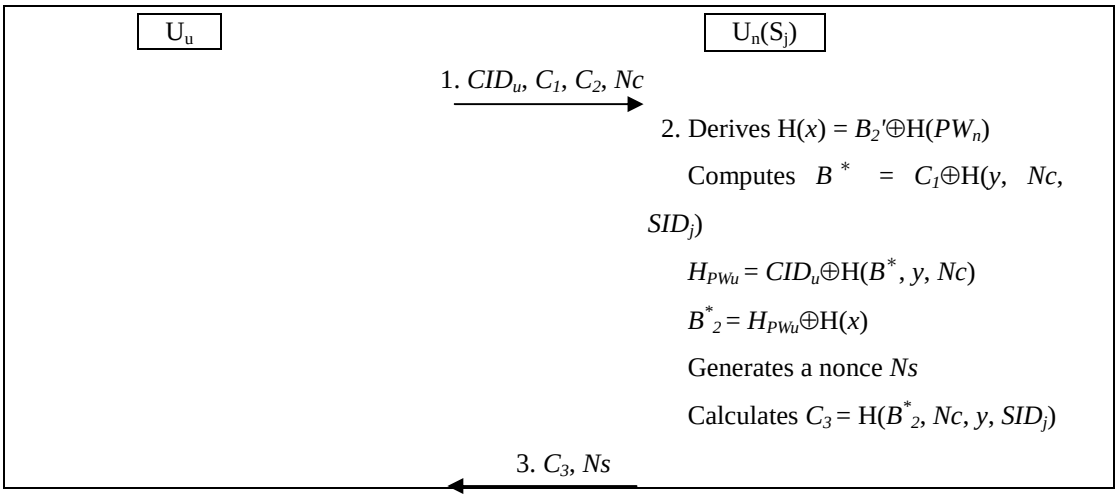
SID_j), which is the same as that in the login phase shown in Figure 6.

2. After receiving the message {CID_u, C_1, C_2, Nc} from U_u, S_i conducts 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 $B^*_2 = H_{PW} \oplus H(x)$. Next, he 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 performs a comparison to determine whether it is equal to the received C_3. If this is the case, 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, whether it is equal to the received C_4. If this is the case, U_u is authentic.
7. After finishing the mutual authentication, U_u and S_i can compute the common session key $SK = H(B_2, Nc, Ns, y, SID_j) = H(B^*_2, Nc, Ns, y, SID_j)$.

These steps demonstrate that the server spoofing attack was successful when S_i masqueraded as S_j.

(2) Server spoofing attack by an insider user

We 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 show that U_n can use both stored values B_2' and y to masquerade as any server to cheat a remote user. First, he can use B_2' and his password PW_n to compute $B_2' \oplus H(PW_n)$, which obtains H(x), before using H(x) and y to fake the desired messages to cheat the remote user. We describe this attack using the following steps and it is also illustrated in Figure 8.



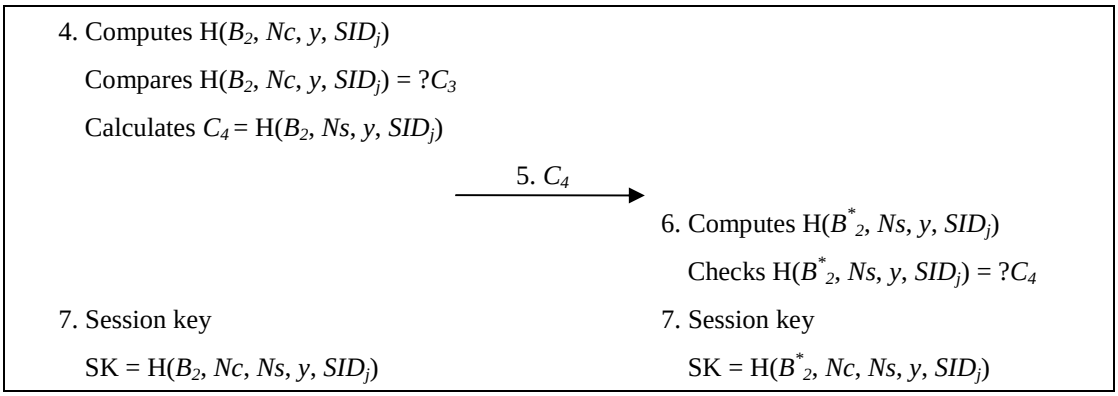


Fig. 8. Server spoofing attack by an insider user on Liao-Wang's protocol

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 $B_2' \oplus H(PW_n)$. Next, he 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 $B^*_2 = H_{PWu} \oplus H(x)$. 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 performs a comparison to determine whether it is equal to the received C_3. If this is the case, U_u authenticates U_n as S_j. Next, he 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 determine whether it is equal to the received C_4. If this is the case, 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).

These steps show that the insider spoofing attack launched by U_n to masquerade as S_j was achieved successfully.

4. Review of Li *et al.*'s protocol

In 2013, Li *et al.* [16] also proposed a multi-server protocol to enhance the scheme of Lee *et al.* [19], the vulnerability of which to insider server attacks was identified by Chou *et al.* [15]. They claimed that their protocol was secure. However, after examining the protocol, we found that it was vulnerable to the smart card lost password-guessing attack if the lost smart card was obtained by an insider user. We illustrate the original scheme in Figure 9. In the following section, we demonstrate the attack only. The

details about the protocol can be found in [16].

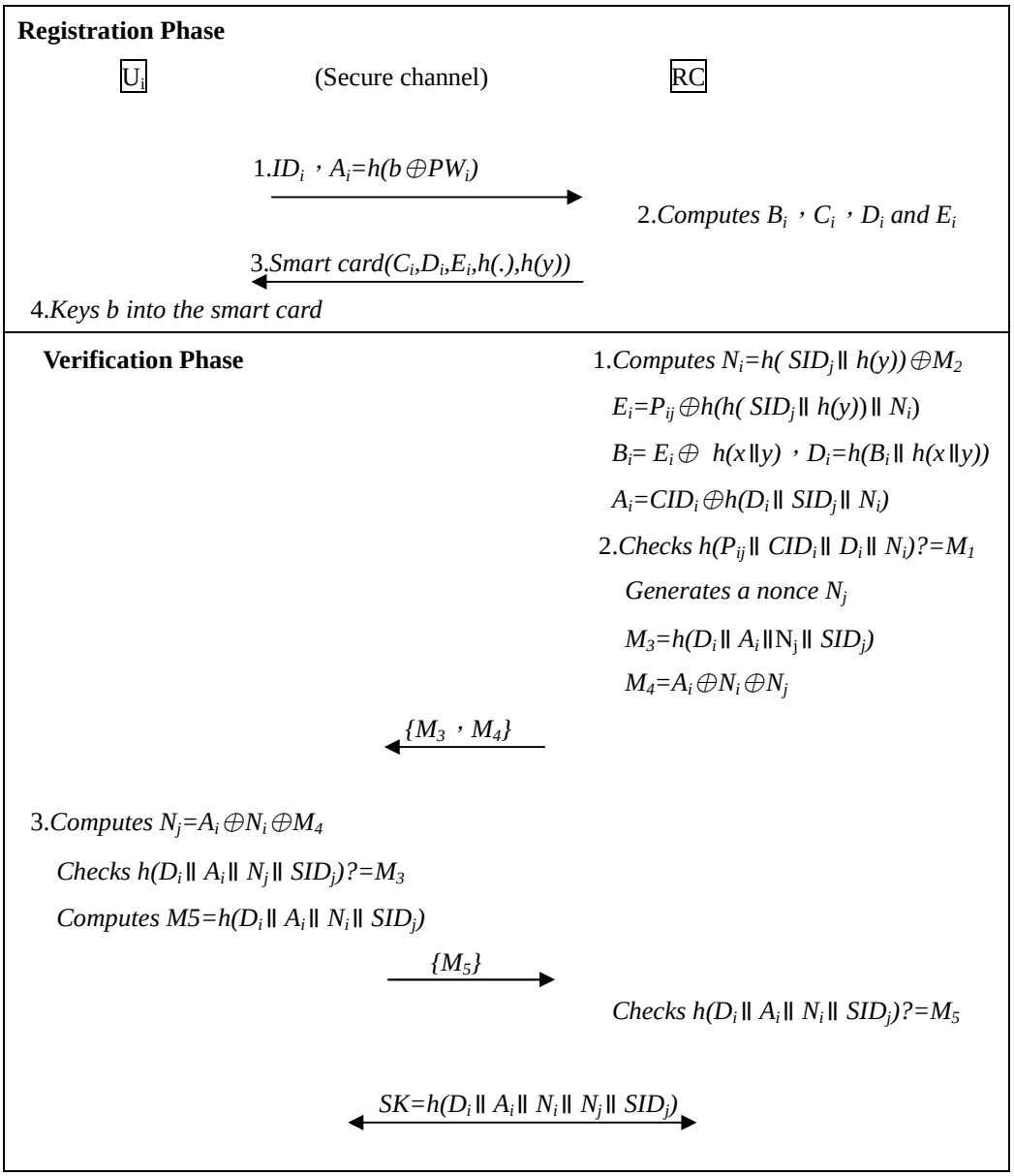


Fig. 9. Li *et al.*'s protocol

4.1 Attack on the protocol

This protocol is vulnerable to the smart card lost password-guessing attack launched by an insider, because after message 3, an insider user who possesses the value of $h(y)$ can obtain the value N_i, and subsequently obtain E_i. Next, based on parameter D_i stored in the lost smart card and CID_i in the transferred message 3, he can obtain A_i. Using the value b stored in the missing smart card and A_i, he can guess the password as *psw* and check to determine whether A_i is equal to $h(b \oplus psw)$. If this is the case, he obtains the right password. Moreover, if a user colludes with a server to obtain the values of $h(y)$

and $h(x \| y)$, their scheme is wholly infeasible.

4.2 Improvement of the protocol

The key point of the smart card lost password-guessing attack is the transfer of M_2 where N_i can easily be calculated by an insider user. To overcome this problem, we reconstructed parts of the original phases in the scheme. First, we reconstructed the registration phase. Next, in the following two phases, the login phase and the verification phase, all the values of *ys* in the original scheme were replaced with y_js. We only list the modifications used to improve the scheme in these two phases, which prevent N_i from being calculated easily when the user's smart card is lost. The other parts that we do not mention remained unchanged. We describe the changes as follows, which are also shown in Figure 10.

Login Phase

U_i (Public channel) S_i

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

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

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

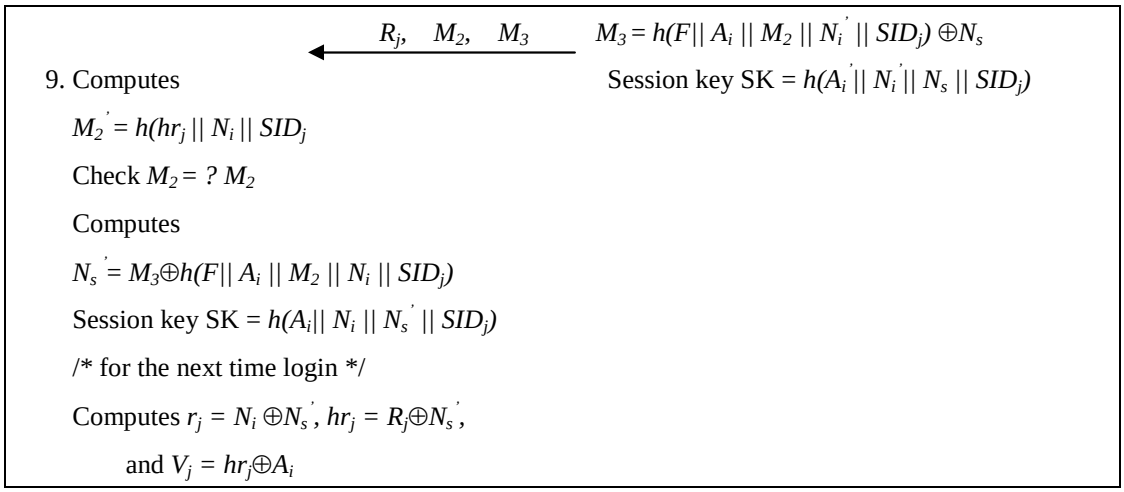


Fig. 10. The proposed improved scheme

(1) Registration phase

During this phase, *RC* chooses a secret number y_j for each server S_j. And computes $h(x|| y_j)$ and $h(SID_j||h(y_j))$, where x is RC's master secret key. It then shares them with S_j via a secure channel. Each user's smart card contains two small arrays V_m and r_m, where m is the number of servers and $1 \le j \le m$. U_i freely chooses his/her identity ID_i, the password PW_i, and computes $A_i = h(b \oplus PW_i)$ and $C_1 = h(ID_i|| h(x)|| A_i)$, where b is a random number generated by U_i and x is RC's secret. Next, U_i sends ID_i and A_i to *RC* for registration via 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 the smart card and inputs ID_i and PW_i. The smart card generates a random number N_i and computes the 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 generates a random number N_s and computes the session key as $h(A_i' || N_i' || N_s || SID_j)$. Next, 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 the 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 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 phase, except the value $h(y)$ in C_i needs to be replaced with $h(x)$.

4.3 Security analysis

In this section, we discuss the security features of our proposed improved scheme based upon the features 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 acquired a previous session key, he cannot get the other session keys because he does not 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 privacy of previously established session keys should not be affected. The session key in our scheme is $h(A_i || N_i || N_s^{'} || SID_j)$, so it has no relationship with the value x. Therefore, this security feature is assured.

(3) Resistance to replay attacks

In our improved scheme, each session's transcript is identified by the session's random variables, N_i and N_s. Thus, all the transmitted parameters are randomized and different from other sessions. More specifically, if an attacker launches this type of attack, he cannot obtain the session key because he lacks the knowledge of A_i. Therefore, this type of attack will fail.

(4) Resistance to forgery attacks

If an attacker launches this type of attack, he must be able to forge the login request to fool the server. However, without any knowledge of A_i and V_j, the attacker cannot make a valid login request. Even if the attacker obtains the smart card and extracts the parameters stored in the smart card, he also cannot forge a login request for the server because he cannot use the stored parameters to compute A_i without knowing the password. Therefore, this type of attack will fail.

(5) Resistance to server spoofing attacks

During server spoofing attacks, 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 cannot spoof another server to fool a legal user because he does not have the other server's secret $h(y_j)$ and $h(x||\ y_j)$, which are required to compute N_i and A_i to produce a valid response message. Therefore, these types of attack will fail.

(6) Resistance to stolen smart card password guessing attacks

If the smart card has been stolen, and the attacker wants to guess the user's password using the stolen smart card, the attacker cannot determine whether the password has been guessed correctly or not because A_i is not stored in the smart card.

(7) Correct mutual authentication

In the improved scheme, 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. Thus, a fabricated message cannot satisfy 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' || $SID_j)\oplus N_s$ cannot satisfy the user's authentication. Therefore, our improved scheme ensures correct mutual authentication.

Based on the above security analyses, we confirmed that our improved scheme outperformed [16] in terms of its security features in the aspect of lost smart card password guessing attack.

5. Conclusion

We analyzed the security of the protocols proposed by Tsai *et al.*, Liao-Wang *et al.*, and Li *et al.* and showed that they were vulnerable to several types of attacks, which we described in this article. Based on the protocol proposed by Li *et al.*, we developed a novel multi-server authentication protocol that outperforms the original scheme in the aspect of avoiding lost smart card password-guessing attack, and is also more efficient, because our improved scheme merely uses hash and exclusive-or operations, and requires 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] H.C. Hsiang, W.K. Shih, "Improvement of the secure dynamic ID based remote user authentication scheme for multi-server environment,"*Computer Standards & Interfaces*, Volume 31, Issue 6, November 2009, Pages 1118 – 1123.

[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.

[18] C.H. Huang, J.S. Chou, Y. Chen, "Improved multi-server authentication protocol" *International journal of Security and Communication Networks*, Volume 5, Issue 3, pages 331–341, March 2012

[19] 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

[20] S.K. Sood, A. K. Sarje, K. Singh, "A secure dynamic identity based authentication protocol for multi-server architecture,"*Journal of Network and Computer Applications* 34 (2011) 609–618

[21] W.J. Tsaura, J.H. Li, W.B. Lee, " An efficient and secure multi-server authentication scheme with key agreement,"*The Journal of Systems and Software* 85 (2012) 876–882

[22] X. Li, Y. Xiong, J. Ma, W. Wang, "An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards," *Journal of Network and Computer Applications* 35 (2012) 763–769

[23] Y.P. Liao, C.M. Hsiao, "A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients'" *Future Generation Computer Systems*, article in press