A Secure Dynamic Identity Based Authentication Protocol with Smart Cards for Multi-Server Architecture

CHUN-TA LI, CHENG-CHI LEE, CHI-YAO WENG AND CHUN-I FAN

1Department of Information Management
Tainan University of Technology
Tainan City, 710 Taiwan
E-mail: th0040@mail.tut.edu.tw

2Department of Library and Information Science
Fu Jen Catholic University
New Taipei City, 242 Taiwan
E-mail: cclee@mail.fju.edu.tw

3Department of Photonics and Communication Engineering
Asia University
Taichung City, 413 Taiwan

4Department of Computer Science
National Pingtung University
Pingtung City, 900 Taiwan
E-mail: cyweng@mail.nptu.edu.tw

5Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung City, 804 Taiwan
E-mail: cifan@cse.nsysu.edu.tw

Due to the rapid growth of computer networks and service providing servers, many network environments have been becoming multi-server architecture and various multi-server authentication protocols have been proposed. In such an environment, a user can obtain different network services from multiple network servers without repeating registration to each server. Recently, Li et al. proposed a secure dynamic ID based authentication protocol for multi-server architecture using smart cards. They claimed that their protocol preserves mutual authentication and protected from several attacks. However, in this paper, we find that Li et al.’s protocol cannot provide the protection against leak-of-verifier attack, impersonation attack, session key disclosure attack and many logged-in users’ attack. To remedy these security flaws, we propose an improved version of dynamic ID based authentication protocol, which covers all the identified weaknesses of Li et al.’s protocol and is more secure and efficient for practical multi-server environments.

Keywords: dynamic identity, password authentication, smart card, multi-server architecture, user anonymity

1. INTRODUCTION

With the rapid growth of the Internet and network technologies, recently smart card based password authentication protocols have been extensively deployed in the Internet or wireless network services for authenticating the validity of a login user. Considering the low-computing capability of smart cards, the authentication scheme design based on
traditional symmetric or asymmetric key cryptosystems is a nontrivial challenge [5, 9]. For reducing the computation costs, numerous smart card based password authentication protocols using the one-way hash function and exclusive-OR operation had been proposed [1, 4, 6, 7, 10, 14, 15, 18].

Currently, more and more single-server services are provided with all walks of life, such as online games, mobile shopping, telecare medicine information systems, integrated electronic medical record systems etc. On the other hand, the multi-server environment emphasizes that any user can obtain different services from multiple network servers without repeating registration to each server because it is impractical for a user to remember numerous different identities and passwords to login and access different service providing servers. In 1994, a well-known Kerberos system [12] in such an environment is proposed. However, it is insecure against password guessing attacks if a user uses a weak password. In 2001, Li et al. [17] proposed a user authentication protocol using neural networks and asymmetric key cryptosystems. For solving the efficiency problem of Li et al.’s protocol, in 2003, Lin et al. [21] proposed a remote user authentication protocol based on discrete logarithm problem for multi-server environments. Later, in 2004, Juang [11] proposed an efficient multi-server authentication protocol based on one-way hashing function and symmetric key cryptosystem. However, in the same year, Chang and Lee [3] showed that Juang’s protocol is vulnerable to off-line dictionary attack. In 2008, Tsai [23] proposed a multi-server authentication protocol based on one-way hashing function and without maintaining a verifier table. However, Tsai’s protocol is vulnerable to impersonation attack and any attacker who intercepts a login message sent from a user, the attacker can resubmit it to the server for gaining illegal access. Furthermore, all the above password authentication protocols for multi-server environment are based on static ID and user’s login ID is changeless and sent in the form of plain-text via public channels. It gives the attacker an opportunity to intercept the login ID from the public channels, uses it to trace the legal user and damages user privacy over public networks.

Recently, Liao and Wang [20] proposed a dynamic ID based authentication protocol and they claimed that their protocol can prevent various attacks. However, Hsiang and Shih [8] showed that Liao-Wang’s protocol cannot resist insider attack, masquerade attack, server spoofing attack and registration center spoofing attack and proposed an improvement on Liao-Wang’s protocol. However, Sood et al. [22] and Lee et al. [13] pointed that Hsiang-Shih’s protocol is still vulnerable to leak-of-verifier attack, stolen smart card attack, masquerade attack and server spoofing attack, and it is not easily repairable. To resist these security weaknesses, Sood et al. proposed a secure dynamic ID based authentication protocol for multi-server architecture and claimed their protocol can prevent attacks and achieve user anonymity. However, Li et al. [16] pointed out that Sood et al.’s protocol is vulnerable to leak-of-verifier attack, stolen smart card problem and incorrect authentication and session key agreement phase. Li et al. further proposed an efficient and security dynamic ID based authentication protocol for multi-server architecture using smart cards to tackle these problems. Unfortunately, through carefully analysis, we find that Li et al.’s protocol is still vulnerable to leak-of-verifier attack, impersonation attack, session key disclosure attack and many logged-in users’ attack. We proposed an improved protocol to solve these security weaknesses. Moreover, our protocol is more efficient than some dynamic ID based authentication protocols regarding
performance by using lightweight one-way hash function and exclusive OR computation.

The rest of the paper is organized as follows: Section 2 briefly reviews Li et al.’s authentication protocol and shows the weaknesses on their protocol in Section 3. In Section 4, our dynamic ID based authentication protocol in multi-server environments is proposed. The security and performance analysis are described in Section 5. Finally, some concluding remarks are included in Section 6.

2. REVIEW OF LI ET AL.’S PROTOCOL

In this section, a brief review of Li et al.’s protocol [16] that contains four phases: registration, login, authentication and session key agreement and password change, are given below, where the following notations have been used in Table 1. Li et al.’s protocol involves three participants: the control server (CS), the service providing server (SJ) and the user (Ui). CS is a trusted party and it is responsible for registration and authentication of Ui and SJ. When SJ register with CS use identity SIDj, CS computes h(SIDj||y) and h(x||y) and submit them to SJ through a secure channel. Detailed steps of four phases are described as follows.

2.1 Registration Phase

When a user Ui wants to access the service and perform registration with CS, Ui chooses his/her identity IDi, password Pi and a random number b, computes Ai=h(b||Pi) and submits (IDi, Ai) to the control server CS through a secure channel. Upon receiving the registration request, CS computes Bi=h(IDi||x), Ci=h(IDi||h(y)||Ai), DiBi \oplus h(IDi||Ai) and EiBi \oplus h(y||x) and stores (Ci, Di, Ei, h(•), h(y)) in Ui’s smart card. Then, CS delivers the smart card to Ui through a secure channel. Ui stores b into his/her smart card and Ui’s smart card contains (Ci, Di, Ei, h(•), h(y), b).

2.2 Login Phase

In this phase, we assume that Ui wants to login to the server SJ and asks a service from SJ. Ui must perform the following steps:

Step 1: Ui inserts his/her smart card into a card reader and inputs identity IDi, password Pi and the server’s identity SIDj. Then, smart card computes Ai=h(b||Pi) and C\prime i=h(IDi||h(y)||A) and checks whether C\prime i=Ci. If it holds, it means Ui is a legal user.

Step 2: After verification, the smart card computes B\prime i=h(IDi||A), F\prime i=h(y)||Ni, Ei \oplus h(y)||Ni||SIDj, CIDiA \oplus h(B||F\prime i||Ni) and G\prime i=h(B||A||Ni) and submits the login request (F\prime i, G\prime i, Ei, CID) to SJ via a public channel, where Ni is a random number chosen by Ui.

2.3 Authentication and Key Agreement Phase

Upon receiving the login request message (F\prime i, G\prime i, Ei, CID) from Ui, SJ and CS performs the following steps to verify the validity of Ui:

...
Step 1: $S_i$ computes $K_i = h(SID_i || y) \oplus N_2$ and $M_i = h(h(x||y)||N_2)$ and sends the login request $(F_i, G_i, P_i, CID_i, SID_i, K_i, M_i)$ to the control server $CS$, where $N_2$ is a random number chosen by $S_j$.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_i$</td>
<td>The $i$th user</td>
</tr>
<tr>
<td>$S_i$</td>
<td>The $j$th server</td>
</tr>
<tr>
<td>$CS$</td>
<td>The control server</td>
</tr>
<tr>
<td>$ID_i$</td>
<td>The identity of $U_i$</td>
</tr>
<tr>
<td>$P_i$</td>
<td>The password of $U_i$</td>
</tr>
<tr>
<td>$SID_j$</td>
<td>The identity of $S_j$</td>
</tr>
<tr>
<td>$x$</td>
<td>The master secret key chosen by $CS$</td>
</tr>
<tr>
<td>$y$</td>
<td>A secret number chosen by $CS$</td>
</tr>
<tr>
<td>$b$</td>
<td>A random number chosen by $U_i$ for registration</td>
</tr>
<tr>
<td>$CID_i$</td>
<td>The dynamic identity generated by $U_i$ for authentication</td>
</tr>
<tr>
<td>$N_1$</td>
<td>A random number generated by $U_i$’s smart card for session key agreement</td>
</tr>
<tr>
<td>$N_2$</td>
<td>A random number generated by $S_j$ for session key agreement</td>
</tr>
<tr>
<td>$N_3$</td>
<td>A random number generated by $CS$ for session key agreement</td>
</tr>
<tr>
<td>$h(\cdot)$</td>
<td>A collision free one-way hash function</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Exclusive OR operation</td>
</tr>
<tr>
<td>$SK$</td>
<td>A common session key shared among $U_i$, $S_j$ and $CS$</td>
</tr>
<tr>
<td>$</td>
<td></td>
</tr>
</tbody>
</table>

Step 2: Upon receiving the login request message $(F_i, G_i, P_i, CID_i, SID_i, K_i, M_i)$ from $S_i$, $CS$ computes $N_2 K_i \oplus h(SID_i || y)$ and $M'_i = h(h(x||y)||N_2)$ and checks whether $M_i = M'_i$. If it holds, the validity of $S_j$ is authenticated by $CS$. Otherwise, $CS$ rejects this session.

Step 3: $CS$ computes $N_1 = F_i \oplus h(y)$, $B_i = P_i \oplus h(h(\cdot)||N_1) || SID_i \oplus h(y) || x$, $A_i = CID_i \oplus h(B_i || F_i || N_1)$ and $G'_i = h(B_i || A_i || N_1)$ and checks whether $G'_i = G'_i$. If it holds, the validity of $U_i$ is authenticated by $CS$. Otherwise, $CS$ rejects this session.

Step 4: $CS$ computes $Q_i N_1 \oplus N_2 \oplus h(SID_i || N_2)$, $R_i = h(A_i || B_i) \oplus h(N_1 \oplus N_2 \oplus N_3)$, $V_i = h(h(A_i || B_i)) || h(N_1 \oplus N_2 \oplus N_3)$ and $T_i = N_2 \oplus N_3 \oplus h(A_i || B_i || N_1)$ and submits the mutual authentication message $(Q_i, R_i, V_i, T_i)$ to $S_j$ via a public channel.

Step 5: Upon receiving the authentication message $(Q_i, R_i, V_i, T_i)$ from $CS$, $S_i$ computes $N_1 \oplus N_2 Q_i \oplus h(SID_i || N_2)$, $h(A_i || B_i) = R_i \oplus h(N_1 \oplus N_2 \oplus N_3)$ and $V'_i = h(h(A_i || B_i)) || h(N_1 \oplus N_2 \oplus N_3)$ and checks whether $V'_i = V'_i$. If it holds, the legitimacy of the control server $CS$ is authenticated by $S_i$. Then $S_i$ sends the response message $(V_i, T_i)$ to $U_i$ via a public channel.

Step 6: Upon receiving the response message $(V_i, T_i)$ from $S_i$, $U_i$’s smart card computes $N_2 \oplus N_3 = T_i \oplus h(A_i || B_i || N_1)$ and $V''_i = h(h(A_i || B_i)) || h(N_2 \oplus N_3 \oplus N_1)$ and checks whether $V''_i = V''_i$. If it holds, the legitimacy of the control server $CS$ and the server $S_j$ are authenticated by
A SECURITY DYNAMIC IDENTITY BASED AUTHENTICATION PROTOCOL

2.4 Password Change Phase

When a user \( U_i \) wants to change his/her password \( P_i \) with a new password \( P_{\text{new}} \) without the help of the control server \( CS \). \( U_i \) inserts the smart card into a card reader and inputs \( ID_i \) and \( P_i \). Then smart card computes \( A_i = h(b || P_i) \), \( B_i = D_i \oplus h(ID_i || A_i) \) and \( C_i = h(ID_i || A_i) \) and checks whether \( C_i = C'_{\text{new}} \). If it holds, \( U_i \) is asked to inputs a new password \( P_{\text{new}} \) and the smart card computes \( A_{\text{new}} = h(b || P_{\text{new}}) \), \( C_{\text{new}} = h(ID_i || A_{\text{new}}) \) and \( D_{\text{new}} = B_i \oplus h(ID_i || A_{\text{new}}) \) and replaces \((C_i, D_i)\) with \((C_{\text{new}}, D_{\text{new}})\).

3. SECURITY ANALYSIS OF LI ET AL.’S PROTOCOL

Although Li et al. claimed that their protocol can resist many types of attacks and satisfy all the essential requirements for multi-server architecture authentication. However, the actual situation is not the case and the cryptanalysis of Li et al.‘s protocol has been made in this section. The detailed analysis are described as follows.

3.1 Leak-of-verifier Attack

In Li et al.’s protocol, we found that their protocol may suffer from leak-of-verifier attack and a malicious privileged user \( U_k \) who possesses the smart card can easily derive the values of \( h(y) \) and \( h(y||x) \). Note that \( h(y) \) and \( h(y||x) \) are used in all the privileged users. Thus, Li et al.’s protocol cannot prevent leak-of-verifier attack.

3.2 Impersonation Attack

Due to \( h(y) \) and \( h(y||x) \) are used in all the privileged users, in case of system secrets \( h(y) \) and \( h(y||x) \) are successfully derived by \( U_k \), \( U_k \) can use them to derive any privileged user \( U_i \)’s \( B_i h(ID_i||x) \). Once \( h(y) \), \( h(y||x) \) and \( U_i \)’s \( B_i \) are leaked accidentally to \( U_k \), he/she can easily make a valid login request to impersonate the user \( U_i \) to login to the server \( S_j \). The details of this attack are described as follows.

Step 1: \( U_k \) eavesdrops \( U_i \)’s login request message and gets \( \{F_i, G_i, P_{ij}, CID_i\} \) over a public channel.
Step 2: \( U_k \) derives \( U_i \)’s \( N_{i1} \) by computing \( N_{i1} = F_i \oplus h(y) \), where \( h(y) \) is derived from \( U_k \)’s
Step 3: $U_i$ computes $h(h(y)\|N_i)\|\langle SID\rangle$ and derives $E_i$ by computing $E_i=\pi_i \oplus h(h(y))\|N_i \| \langle SID\rangle$.

Step 4: $U_i$ derives $U_i$’s secret value $B_i$ by computing $B_i=E_i \oplus h(y|x)$, where $h(y|x)$ is derived from $U_i$’s smart card.

As shown in above-mentioned steps, $U_i$’s secret value $B_i$ can be easily derived. Then, $U_i$ can choose a new random number $N_i$, generate a meaningless value $A_i$, and make a valid login request $\{F_i, G_i, P_i, CID_i\}$ to $S_j$, where $F_i=h(y) \oplus N_i, G_i=h(B_i)\|A_i\|N_i\|\langle SID\rangle$, and $CID_i=A_i \oplus h(B_i)\|F_i\|N_i$. Moreover, $U_i$ can establish a new session key $SK=h(A_i\|B_i)(N_i\|N_i\|N_i)$ with $S_j$ and $CS$ and achieve the purpose of a cheat. So, Li et al.’s protocol cannot resist the impersonation attack as they claimed.

### 3.3 Session Key Disclosure Attack

As shown in the impersonation attack, in case of $U_i$’s secret value $B_i$ is successfully derived by $U_j$, $U_i$ can use it to disclose the session key shared among the user $U_i$, the server $S_j$ and the control server $CS$. The details of this attack are described as follows.

Step 1: $U_i$ eavesdrops $U_i$’s login request message and gets $\{F_i, G_i, P_i, CID_i\}$ over a public channel.

Step 2: $U_i$ eavesdrops $S_j$’s response message and gets $\{V_i, T_i\}$ over a public channel.

Step 3: $U_i$ derives $U_i$’s random number $N_i$ by computing $N_i=F_i \oplus h(y)$, where $h(y)$ is derived from $U_i$’s smart card.

Step 4: $U_i$ derives $U_i$’s secret value $B_i$ by launching the attack mentioned in Section 3.2.

Step 5: $U_i$ derives $U_i$’s secret value $A_i$ by computing $A_i=CID_i \oplus h(B_i)\|F_i\|N_i$, where $CID_i$ and $F_i$ are collected from Step 1.

Step 6: $U_i$ derives $N_i \oplus N_j$ by computing $N_i \oplus N_j=T_i \oplus h(A_i)\|B_i\|N_i$, where $T_i$ is collected from Step 2.

Step 7: Finally, $U_i$ can easily disclose the session key $SK$ by computing $SK=h(A_i\|B_i)(N_i\|N_i\|N_i)$.

### 3.4 Many Logged-In Users’ Attack

In Li et al.’s protocol, the simultaneous access of a legitimate user’s account in the control server by multiple non-registered users using the same identity and password of the user and the control server is not aware of having caused weakness. We assume that a registered user’s smart card is massively duplicated and $U_i$’s ID$_i$ and $P_i$ is intentionally exposed to $n$ attackers $U_k$, where $x_1, 2, \ldots, n$. Then all who has smart card and knows ID$_i$ and $P_i$ can login to the control server $CS$ at the same time by performing the following steps:

Step 1: Each $U_k$ generates his/her random number $N_k$ and sends a valid login request $(F_k, G_k, P_k, CID_k)$ to $CS$, where $F_k=h(y) \oplus N_k, G_k=h(B_i)\|A_i\|N_k, P_k=E_\pi \oplus h(h(y))\|N_k\|\langle SID\rangle, CID_k=A_i \oplus h(B_i)\|F_k\|N_k$ and $x=1, 2, \ldots, n$. 
Step 2: Upon receiving all the login requests \( (F_{k1}, G_{k1}, P_{ij1}, CID_{ij1}), (F_{k2}, G_{k2}, P_{ij2}, CID_{ij2}), \ldots, (F_{kn}, G_{kn}, P_{ijn}, CID_{ijn}) \) from \( U_{k1}, U_{k2}, \ldots, U_{kn} \), CS gets the same identity \( ID_i \) with different random numbers \( N_{k1}, N_{k2}, \ldots, N_{kn} \). Finally, CS allows all of \( U_{k1}, U_{k2}, \ldots, U_{kn} \) to login and access \( U_i \)'s account simultaneously.

4. THE PROPOSED PROTOCOL

In this section, we propose a strong and efficient protocol to overcome the security weaknesses of Li et al.'s protocol. Our protocol consists of four phases namely: registration, login, authentication and session key agreement and password change. \( S_j \) registers itself with CS by using identity \( SID_i \). Then, CS computes \( h(SID_i||y) \) and \( h(x||y) \) and submit them to \( S_j \) through a secure channel. These four phases are further illustrated below and shown in Fig. 1.

Fig. 1. The proposed protocol.
4.1 Registration Phase

When a user \( U_i \) wants to perform registration with the control server \( CS \), \( U_i \) chooses his/her \( ID_i \), \( P_i \) and a large random number \( b \), computes \( A_i = h(ID_i \| b \| P_i) \) and submits \( \{ ID_i, A_i \} \) to \( CS \) through a secure channel. Upon receiving the registration request, \( CS \) generates a pseudonym \( TID_i \), computes \( B_i = h(ID_i \| x) \), \( C_i = h(ID_i \| h(y) \| A_i) \) and \( D_i = B_i \oplus h(x) \oplus A_i \) and stores \( \{ TID_i, C_i, D_i, h(\cdot), h(y) \} \) in \( U_i \)'s smart card. Moreover, \( CS \) stores each legal user's pseudonym, user-verifier and a status-bit in a protected verifier table as depicted in Table 2, where the status-bit indicates the login status of the user to withstand many logged-in users' attack. If the user is logged-in to \( CS \), the status-bit is set to 1, otherwise it is set to 0. Finally, \( CS \) delivers the smart card to \( U_i \) through a secure channel. \( U_i \) stores \( b \) into his/her smart card and \( U_i \)'s smart card contains \( \{ TID_i, C_i, D_i, h(\cdot), h(y), b \} \).

<table>
<thead>
<tr>
<th>User pseudonym</th>
<th>User-verifier</th>
<th>Status-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( TID_i )</td>
<td>( B_i ), ( h(ID_i | x) )</td>
<td>0/1</td>
</tr>
<tr>
<td>( TID_j )</td>
<td>( B_j ), ( h(ID_j | x) )</td>
<td>0/1</td>
</tr>
</tbody>
</table>

4.2 Login Phase

In this phase, we assume that \( U_i \) wants to login to the server \( S_j \) and asks a service from \( S_j \), \( U_i \) must perform the following steps:

**Step 1:** \( U_i \) inserts his/her smart card into a card reader and inputs identity \( ID_i \), password \( P_i \) and the server's identity \( SID_j \). Then, smart card computes \( A_i = h(ID_i \| b \| P_i) \) and \( C_i = h(ID_i \| h(y) \| A_i) \) and checks whether \( C_i \oplus C_i \). If it holds, it means \( U_i \) is a legal user.

**Step 2:** After verification, the smart card generates a random number \( N_i \) and computes \( F_i = D_i \oplus A_i \oplus N_i \) and \( G_i = h(h(y) \| N_i \| |SID_j|) \). Then, the smart card submits the login request \( \{ TID_i, F_i, G_i \} \) to \( S_j \) via a secure channel.

4.3 Authentication and Session Key Agreement Phase

Upon receiving the login request message \( \{ TID_i, F_i, G_i \} \) from \( U_i \), \( S_j \) and \( CS \) performs the following steps to verify the validity of \( U_i \):

**Step 1:** \( S_j \) computes \( K_i = h(SID_j \| y) \oplus N_{i2} \) and \( M_i = h(x \| y) \| N_{i2} \) and sends the login request \( \{ TID_i, F_i, G_i, K_i, M_i \} \) to the control server \( CS \), where \( N_{i2} \) is a random number chosen by \( S_j \).

**Step 2:** Upon receiving the login request message \( \{ TID_i, F_i, G_i, K_i, M_i \} \) from \( S_j \), \( CS \) computes \( N_{i2} = K_i \oplus h(SID_j \| y) \) and \( M'_i = h(x \| y) \| N_{i2} \) and checks whether \( M_i = M'_i \). If it holds, the validity of \( S_j \) is authenticated by \( CS \). Otherwise, \( CS \) rejects this session.

**Step 3:** According to user's pseudonym \( TID_i \), \( CS \) retrieves user's verifier \( B_i \), computes \( N_{i1} = F_i \oplus B_i \oplus h(x \| y) \) and \( G'_i = h(y) \| N_{i1} \| |SID_j| \) and checks whether \( G_i = G'_i \). If it holds, the
validity of $U_i$ is authenticated by $CS$. Otherwise, $CS$ rejects this session.

**Step 4:** $CS$ refreshes a new pseudonym $TID_{new}$ for $U_i$ by computing $TID_{new}=TID_i\oplus N_i$ and replaces $TID_i$ with $TID_{new}$.

**Step 5:** $CS$ computes $Q=h(N_i+TID_i\oplus h(SID_i)\|N_i)$, $R_i=h(B_i\oplus h(x\|y))\oplus h(N_i+TID_i\oplus N_i)$, $V_i=(h(B_i\oplus h(x\|y)))\|h(N_i+TID_i\oplus N_i))$ and $T_i=N_i\oplus N_i\oplus h(B_i\oplus h(x\|y))\oplus h(TID_{new}))$ and submits the mutual authentication message $(Q_i, R_i, V_i, T_i)$ to $S$ via a public channel.

**Step 6:** Upon receiving the authentication message $(Q_i, R_i, V_i, T_i)$ from $S$, $S_i$ computes $N_i+TID_i\oplus h(SID_i)\|N_i)$, $h(B_i\oplus h(x\|y))=R_i\oplus h(N_i+TID_i\oplus N_i)$ and $V_i'=h(h(B_i\oplus h(x\|y)))\|h(N_i+TID_i\oplus N_i))$ and checks whether $V_i'=V_i$. If it holds, the legitimacy of the control server $CS$ is authenticated by $S_i$. Then $S_i$ sends the response message $(V_i', T_i)$ to $U_i$ via a public channel.

**Step 7:** Upon receiving the response message $(V_i', T_i)$ from $S_i$, $U_i$’s smart card computes $TID_{new}=TID_i\oplus N_i$, $B_i\oplus h(x\|y)=D_i\oplus A_i$, $N_i\oplus TID_i\oplus h(B_i\oplus h(x\|y)\oplus h(TID_{new}))$ and $V_i'=h(h(B_i\oplus h(x\|y)))\|h(N_i+TID_i\oplus N_i))$ and checks whether $V_i'=V_i$. If it holds, the legitimacy of the control server $CS$ and the server $S_i$ are authenticated by $U_i$ and the smart card replaces $TID_i$ with $TID_{new}$.

Finally, $U_i$, $S_i$ and $CS$ agree on a common session key $SK$ by computing $SK=h(h(B_i\oplus h(x\|y)))\|h(N_i+TID_i\oplus N_i))$.

### 4.4 Password Change Phase

When a user $U_i$ wants to change his/her password $P_i$ with a new password $P_{new}$ without the help of the control server $CS$, $U_i$ inserts the smart card into a card reader and inputs $ID_i$ and $P_i$. Then smart card computes $A_i=h(ID_i\|b\|P_i)$ and $C_i=h(ID_i\|h(y)|A_i)$, and checks whether $C_i$. If it holds, $U_i$ is asked to inputs a new password $P_{new}$. Then, the smart card computes $A_{new}=h(ID_i\|b\|P_{new})$ and $C_{new}=h(ID_i\|h(y)|A_{new})$, and store $C_{new}$ into the smart card to replace $C_i$ to finish the password change phase.

### 5. ANALYSIS OF THE PROPOSED PROTOCOL

In this section, we evaluate the security of our protocol and make comparisons with other related dynamic ID based multi-server authentication protocols [8, 16, 19, 20, 22] in terms of performance and functionality. The proposed protocol can not only resist malicious attacker from launching several well-known cryptographical attacks but also provide much better performance when compared to related dynamic ID protocols for multi-server architecture. Prior to demonstration of preventing possible attacks, some assumptions of security are given.

**Assumption 1:** $U_i$’s identity and password are well-protected by the user and securely sent to $CS$ in the registration phase.

**Assumption 2:** There is a secure channel between each server $S_i$ and the control server $CS$ in the registration phase.

**Assumption 3:** During the login, authentication and session key agreement phases, the attacker $U_h$ has control over the communication channels between $U_i$, $S_i$ and $CS$ such as
eavesdropping, intercepting, inserting and deleting any transmitted messages in these public channels.

**Assumption 4:** A privileged user can extract the stored values from his/her smart card by monitoring its power consumption.

**Assumption 5:** The bit length of \((x, y, b, N_{i1}, N_{i2}, N_{i3})\) are large enough and these values are high entropy secret keys and random numbers.

The details of several security attributes in the proposed protocol are described in Fig. 1.

### 5.1 Security Analysis

As mentioned in previous researches, the following security features are important for dynamic ID based user authentication protocol and we compare the proposed protocol with Li et al.’s protocol [16], Sood et al.’s protocol [22], Liao et al.’s protocol [20], Hsiang et al.’s protocol [8] and Li et al.’s protocol [19]. The security comparisons of the proposed protocol with some related dynamic ID based multi-server authentication protocols are given in Table 3.

<table>
<thead>
<tr>
<th>Protocols/Security criteria</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
<th>S8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hsiang et al.’s protocol (2009) [8]</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Liao et al.’s protocol (2009) [20]</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sood et al.’s protocol (2011) [22]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Li et al.’s protocol (2011) [16]</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Li et al.’s protocol (2013) [19]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>The proposed protocol</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**S1: Resistance to smart card lost problem**

The smart card lost problem is an inherent limitation of user authentication protocol. We found that the best solution is to prohibit the guessestimate chance of the malicious off-line password guessing attack and the secret values stored on the smart card are \(\{TID, C_i, D_i, h(\cdot), h(y), b\}\) in our proposed protocol. We assume that the secrets and \(b\) extracted from user’s smart card, it still cannot help an attacker \(U_k\) to derive or update the user’s password without knowing the user’s ID\(_i\) and \(P_y\). On the other hand, since \(x\) and \(y\) are unknown to \(U_k\), he/she cannot compute \(B_i \oplus h(y||x)\) and cannot perform the impersonation attack using the stolen or lost smart card.

**S2: Resistance to leak-of-verifier attack**

In the proposed protocol, a malicious privileged user \(U_k\) who possesses the smart card may try to derive the values \(h(y||x)\) and \(h(ID_i||x)\). However, \(U_k\) still cannot successfully launches a leak-of-verifier attack on it because \(h(y||x)\) and \(h(ID_i||x)\) are protected by the one-way hash function. Since \(x\) and \(y\) are protected by the one-way hash function, \(U_k\) cannot ensure which ID is corresponding to \(h(ID_i||x)\) and cannot forge a valid login message \((TID, F, G)\). As a result, \(U_k\) can impersonate as a service proving server or a privi-
A SECURITY DYNAMIC IDENTITY BASED AUTHENTICATION PROTOCOL

S3: Resistance to impersonation attack

In this attack, the attacker $U_k$ may try to forge a valid login request ($TID_i, F_i, G_i$) to impersonate as a legitimate user using the previously eavesdropped messages. However, in the proposed protocol, $U_k$ cannot generate $TID_i, FB_i \oplus h(y) || x \oplus N_{k1}$ and $G_i = h(h(y)) || N_{k1} || SID_j$ since he/she without the knowledge of $N_{k1}$, $h(y)$, $B_i \oplus h(y) || x$ and user’s pseudonym $TID_i$ is updated and refreshed in every session, so $U_k$ cannot impersonate as the legitimate user $U_i$, where $N_{k1}$ is a random number generated by $U_k$.

S4: Resistance to session key disclosure attack

In this type of attack, we assume that the attacker $U_k$ eavesdrops the previously session messages ($TID_i, F_i, G_i, K_i, M_i$) and ($Q_i, R_i, V_i, T_i$). Since the user verifier $B_i \oplus h(x) || y$ is protected by $U_i$, $U_k$ cannot derive $U_i$’s nonce value $N_{k1}$ from $F_i$ and cannot derive $S_j$’s and $CS$’s nonce values $N_{k2} \oplus N_{k3}$ from $T_i$. Therefore, $U_k$ cannot derive the common session key $SK = h(h(B_i \oplus h(x) || y)) || (N_{k1} \oplus N_{k2} \oplus N_{k3})$ and the proposed protocol resists the session key disclosure attack.

S5: Resistance to stolen-verifier attack

The stolen-verifier attack means that an attacker $U_k$ steals the verifier table from the control server and launches an impersonation attack. Thus, $U_k$ can masquerade as a legitimate user $U_i$ to login control server. In the proposed protocol, the verifier table contains $U_i$’s pseudonym $TID_i$ and user-verifier $B_i h(ID_i || x)$. However, it is impossible to make a valid value $B_i \oplus h(x) || y$ because $x$ and $y$ are well-protected by $CS$. So the proposed protocol can resist the stolen-verifier attack.

S6: Resistance to time synchronization problem

The replay attack is an important issue in any multi-party based authentication protocol and the malicious attacker may masquerade as a legal user by replaying the login request eavesdropping from a previous session. Therefore, many dynamic ID based user authentication protocols used timestamp to provide the security against the replay attack in a remote login system. However, the timestamp arises the time synchronization problem and the proposed protocol replaces the timestamp with nonce values to avoid this problem. We use three different nonce values ($N_{k1}, N_{k2}, N_{k3}$) to prevent replay attack and time synchronization problem and three participating parties establish a common session key by finishing four-round challenge-response mechanism.

S7: Resistance to many logged-in users’ attack

In the proposed protocol, we assume that the user’s pseudonym $TID_i$ and $B_i \oplus h(x) || y$ are leaked to more than one impostor. However, the control server maintained a status-bit in its verifier table and no one allowed to login control server at the same time out of all who know the valid pseudonym $TID_i$ and $B_i \oplus h(x) || y$. Based on the verifier table, we can say that our proposed protocol can resist the many logged-in users’ attack.

S8: Provision of user anonymity

In the registration phase of the proposed protocol, a user pseudonym and a secure
channel between the user and the control server are used for protecting the user’s real
identity from disclosure. In the login phase, the user sends the pseudonym as a substitute
for the real identity for its authentication to the service providing server and the control
server. Moreover, the attacker cannot distinguish different sessions corresponding to a
certain user since user’s dynamic identity $TID_i$ is different for each session when the user
logins to the server. Therefore, the user anonymity is ensured in our proposed.

5.2 Correctness Analysis

In this subsection, we use the BAN logic [2] to analyze the correctness of the ses-
sion key between user, service server and control server. Some notations used in BAN
logic analysis are described as follows:

- $A \equiv X$: It means that $A$ believes the formula $X$ is true.
- $A \prec X$: It means that $A$ sees the formula $X$.
- $A \Rightarrow X$: It means that $A$ has complete control over the formula $X$.
- $A \sim X$: It means that $A$ has once said the formula $X$.
- $(X)$: It means that $X$ is fresh. The formula $X$ has not been used before or $X$ is a nonce.
- $BKA$: It means that principals $A$ and $B$ may use the shared key $K$ to communicate.
- Note that $K$ will never be discovered by any principals except $A$ and $B$.
- $<X>Y$: It means that formula $X$ is combined with a secret parameter $Y$.
- $(X)_K$: It means that formula $X$ is hashed with a key $K$.
- $2_1$ Rule: It can infer Rule 2 from Rule 1. For example, $A$ creates random $X$, so $A$ believes $X$ is fresh.
- $SK$: A session key established in each session.

According to the analytic procedures of BAN logic, two participators $U_i$ and $S_j$ co-
operatively run the proposed protocol with the help of the control server $CS$ and we list
the verification goals of our protocol as follows:

(G.1): $U_i \equiv U_i SK S_j$
(G.2): $S_j \equiv U_i SK S_j$
(G.3): $U_i \equiv S_j \equiv U_i SK S_j$
(G.4): $S_j \equiv U_i \equiv U_i SK S_j$

Next, we use BAN logic to transform our protocol, illustrated in Fig. 1 into the idealized
form. The protocol generic types are shown in the following:

Message 1: $U_i \rightarrow S_j$: $TID_i, h(ID_i)\oplus h(x)\oplus h(y)\oplus N_{11}, h(h(y)\oplus N_{11}\oplus SID_j)$
Message 2: $S_j \rightarrow CS$: $TID_i, h(ID_i)\oplus h(x)\oplus h(y)\oplus N_{11}\oplus SID_j, h(SID_j)\oplus h(x)\oplus h(y)\oplus N_{12}, h(h(x)\oplus h(y)\oplus N_{12})$
Message 3: $CS \rightarrow S_j$: $N_{11} \oplus N_{12} \oplus h(SID_j)\oplus h(x)\oplus h(y)\oplus h(N_{11} \oplus N_{12} \oplus N_{13})$, $h(h(B_{12} \oplus h(x)\oplus h(y))\oplus h(N_{11} \oplus N_{12} \oplus N_{13}) \oplus h(B_{12} \oplus h(x)\oplus h(y))\oplus h(TID_{new}))$
Message 4: $S_j \rightarrow U_i$: $h(h(B_{12} \oplus h(x)\oplus h(y))\oplus h(N_{11} \oplus N_{12} \oplus N_{13}) \oplus h(B_{12} \oplus h(x)\oplus h(y))\oplus h(TID_{new}))$
A SECURITY DYNAMIC IDENTITY BASED AUTHENTICATION PROTOCOL

Idealize form of the proposed protocol:

**Message 1.** $U_i \rightarrow S_j$: $TID_i$, $(ID_i, x, y)$, $h(y)$, $SID_j$ > $N_{i1}$

**Message 2.** $S_j \rightarrow CS$: $TID_i$, $(ID_i, x, y)$, $h(y)$, $SID_j$ > $N_{i1}$, $(x, y)$, $TID_{new}$ > $N_{i2}$

**Message 3.** $CS \rightarrow S_j$: $(SID_j, y)$, $h(x, y)$, $h(ID_i, x)$, $h(x, y)$, $h(ID_i, x)$, $h(x, y)$, $TID_{new}$ > $N_{i2}$

**Message 4.** $S_j \rightarrow U_i$: $(x, y)$, $h(ID_i, x)$, $h(x, y)$, $TID_{new}$ > $N_{i2}$

To analyze the proposed protocol, the following assumptions are also required:

(A.1): $U_i \equiv\#(N_{i1})$

(A.2): $S_j \equiv\#(N_{i2})$

(A.3): $CS \equiv\#(N_{i3})$

(A.4): $U_i \equiv U_j(h(ID_j, x))$ CS

(A.5): $CS \equiv U_j h(ID_j, x)$ CS

(A.6): $U_i \equiv U_j h(ID_j, x)$ CS

(A.7): $CS \equiv U_j h(ID_j, x)$ CS

(A.8): $CS \equiv S_j h(SID_j, y)$ CS

(A.9): $CS \equiv S_j h(SID_j, y)$ CS

(A.10): $S_j \equiv S_j h(x, y)$ CS

(A.11): $CS \equiv S_j h(x, y)$ CS

(A.12): $U_i \equiv U_j h(y)$ CS

(A.13): $CS \equiv U_j h(y)$ CS

(A.14): $U_i \equiv CS \Rightarrow U_j SK S_j$

(A.15): $S_j \equiv CS \Rightarrow U_j SK S_j$

(A.16): $U_i \equiv S_j \Rightarrow U_i SK S_j$

(A.17): $S_j \equiv U_j SK S_j$

Based on the above-mentioned assumptions, the preliminary procedures of BAN logic are well prepared and we show the main steps of the verification proof as follows:

According to the Message 1, we could obtain:

(V.1): $S_j \equiv (TID_i, (ID_i, x), (x, y), N_{i1}, (h(y), SID_j), N_{i1})$

According to the Message 2, we could obtain:

(V.2): $CS \equiv (TID_i, (ID_i, x), (x, y), N_{i1}, (h(y), SID_j), N_{i1}, (SID_j, y), N_{i2}, (x, y), N_{i2})$

According to the assumption (A.9), we would apply the message meaning rule to get:

(V.3): $CS \equiv S_j \equiv N_{i2}$

According to the assumption (A.2), we could apply the freshness conjunct-catenation...
rule to get:
(V.4): \( CS' \models \#(x, y)_{N_2} \)

According to the (V.3) and (V.4), we could apply nonce verification rule to get:
(V.5): \( CS' \models S_j \models (x, y)_{N_3} \)

According to the assumption (A.11) and (V.5), we could apply the jurisdiction rule to get:
(V.6): \( CS' \models N_{12} \)

According to the assumptions (A.5) and (A.7), we could apply the message meaning rule to get:
(V.7): \( CS' \models U_i \sim N_{11} \)

According to the assumption (A.13), we could apply the freshness conjunction rule to get:
(V.8): \( CS' \models \# < (h(y), SID) > N_{11} \)

According to the (V.7) and (V.8), we could apply nonce verification rule to get:
(V.9): \( CS' \models U_j \models < (h(y), SID) > N_{11} \)

According to the assumption (A.13) and (V.9), we could apply the jurisdiction rule to get:
(V.10): \( CS' \models N_{11} \)

According to the assumption (A.1), we could apply the freshness conjunction rule to get:
(V.11): \( S_j \models < (SID_j, N_{12}) > (N_{11}, N_{12}, N_{13}) > (h(ID, x), h(x, y), < (ID, x) > (x, y), TID_{new} > (N_{12}, N_{13}) \)

According to (V.11) and (V.6), we could obtain:
(V.12):
\[
\frac{S_j \models S_j \models N_{12} \models CS \models < (SID_j, N_{12}) > (N_{11}, N_{12})}{S_j \models CS \models < (SID_j, N_{12}) > (N_{11}, N_{12})}
\]

According to (V.12), we could obtain:
(V.13): \( S_j \models CS \models \sim (N_{11}, N_{12}) \)

According to (V.13), (V.11) and assumption (A.2), we could obtain:
(V.14):
\[
\frac{S_j \models \#(N_{12}), S_j \models CS \models \sim (N_{11}, N_{12}) \models S_j \models < (N_{11}, N_{12}, N_{13}) > (h(ID, x), h(x, y))}{S_j \models CS \models \sim (N_{11}, N_{12}, N_{13}) > (h(ID, x), h(x, y))}
\]

According to (V.14), we could obtain:
(V.15): \( S_j \models CS \models \sim (h(ID, x), h(x, y)) \)

According to \( SKh(h(h(ID, x) \oplus h(x, y))(N_{11} \oplus N_{12} \oplus N_{13})) \), (V.13), (V.15) and assumption (A.2), we could obtain:
(V.16): \( S_j \models \#(U_i)^{SK} S_j \)

According to (V.16) and assumptions (A.15) and (A.17), we could obtain:
(V.17): \( S_j \models U_i \sim U_i^{SK} S_j \)

According to the Message 4, we could obtain:
(V.18): \( S_j \models U_j \models U_i^{SK} S_j \)

According to the Message 4, we could obtain:
(V.19): \( U_i \models < (N_{11}, N_{12}, N_{13}) > (h(ID, x), h(x, y), TID_{new} > (N_{12}, N_{13}) \)

According to (V.19) and (V.10), we could obtain:
A SECURITY DYNAMIC IDENTITY BASED AUTHENTICATION PROTOCOL

According to (V.20), we could obtain:

\[(V.21): U_i\triangleright CS\triangleright (N_{i_2}, N_{i_3})\]

According to (V.21), (V.19) and assumption (A.1), we could obtain:

\[(V.22): U_i\triangleright h(N_{i_1}), U_i\triangleright CS\triangleright (N_{i_2}, N_{i_3}), U_i\triangleright (N_{i_1}, N_{i_2}, N_{i_3}) \triangleright (h(ID_1), h(x, y))\]

According to (V.22), we could obtain:

\[(V.23): U_i\triangleright CS\triangleright (h(ID_1, x), h(x, y))\]

According to \(SK_h(h(h(ID_1, x)||x)\oplus h(x||y))||(N_{i_1} \oplus N_{i_2} \oplus N_{i_3})\), (V.21), (V.23) and assumption (A.1), we could obtain:

\[(V.24): U_i\triangleright \#(U_i \ SK \ S_j)\]

According to (V.24) and assumptions (A.14) and (A.16), we could obtain:

\[(V.25): U_i\triangleright S_j \sim U_i \ SK \ S_j\]

According to (V.24) and (V.25), we could obtain:

\[(V.26): U_i\triangleright S_j \triangleright U_j \ SK \ S_j\]

Finally, inferring from formulas A.17, V.18 and V.26, we have proven the proposed protocol achieves the verification goals as well as establishes a common session key between \(U_i\) and \(S_j\).

5.3 Performance Analysis

In this subsection, we compare the computational primitives involved in login, authentication and session key agreement phases of our proposed protocol with some related dynamic ID based multi-server authentication protocols [8, 16, 20, 22] and tabulate the results in Table 4. We define the notation \(T_h\) as the time complexity for one-way hashing function and it is usually negligible considering the computation cost of exclusive-OR operation because it requires very few computations.

Obviously, in our proposed protocol, the user’s smart card adopts only three computations of one-way hash function in the login process, and therefore the total computations of our protocol is lower than most of authentication protocols. On the other hand, Liao et al.’s protocol [20] adopts only 15 computations of one-way hash function to construct a multi-server authentication protocol. However, Hsiang et al. [8] pointed out that Liao et al.’s protocol is not secure against many types of attacks such as server spoofing attack, registration center spoofing attack, insider attacker and masquerade attack.

<table>
<thead>
<tr>
<th>Protocols/Phases</th>
<th>Login phase</th>
<th>Verification phase</th>
<th>Total computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hsiang et al.’s protocol (2009) [8]</td>
<td>7(T_h)</td>
<td>17(T_h)</td>
<td>24(T_h)</td>
</tr>
<tr>
<td>Liao et al.’s protocol (2009) [20]</td>
<td>6(T_h)</td>
<td>9(T_h)</td>
<td>15(T_h)</td>
</tr>
<tr>
<td>Sood et al.’s protocol (2011) [22]</td>
<td>7(T_h)</td>
<td>18(T_h)</td>
<td>25(T_h)</td>
</tr>
<tr>
<td>Li et al.’s protocol (2011) [16]</td>
<td>7(T_h)</td>
<td>21(T_h)</td>
<td>28(T_h)</td>
</tr>
<tr>
<td>Li et al.’s protocol (2013) [19]</td>
<td>6(T_h)</td>
<td>10(T_h)</td>
<td>16(T_h)</td>
</tr>
<tr>
<td>The proposed protocol</td>
<td>3(T_h)</td>
<td>20(T_h)</td>
<td>23(T_h)</td>
</tr>
</tbody>
</table>
6. CONCLUSIONS

In this paper, we have shown that Li et al.’s secure dynamic ID based authentication protocol is vulnerable to leak-of-verifier attack, impersonation attack, session key disclosure attack, many logged-in users’ attack, and is not easily reparable. We have proposed a more efficient and secure dynamic ID based password authentication protocol with smart cards for multi-server architecture and removed the security flaws on Li et al.’s protocol. We demonstrate that our protocol not only satisfies all of the essential requirements but also achieves performance efficiency. Therefore, it is more suitable for adopting our protocol in practical applications.

REFERENCES


Chun-Ta Li (李俊達) received the Ph.D. degree in Computer Science and Engineering from National Chung Hsing University, Taiwan, in 2008. He is currently an Assistant Professor of the Department of Information Management, Tainan University of Technology, Tainan, Taiwan. His research interests include information security, wireless sensor networks, mobile computing, and security protocols for ad hoc networks.
Cheng-Chi Lee (李正吉) received the Ph.D. degree in Computer Science from National Chung Hsing University, Taiwan, in 2007. He is currently an Associate Professor with the Department of Library and Information Science at Fu Jen Catholic University. His current research interests include data security, cryptography, network security, mobile communications and computing, wireless communications.

Chi-Yao Weng (翁麒耀) received the Ph.D degree in Computer Science from National Tsing Hua University, Hsinchu, Taiwan, in 2011. He is currently an Assistant Professor with the Department of Computer Science at National Pingtung University, Pingtung, Taiwan. His current research interests include data analysis, mobile security, multimedia security, digital right management and information forensics.

Chun-I Fan (范俊逸) received the Ph.D. degree in Electrical Engineering at National Taiwan University in 1998. From 1999 to 2003, he was an Associate Researcher and project leader of Telecommunication Laboratories, Chunghwa Telecom Co., Ltd., Taiwan. He has been a Full Professor since 2010. His current research interests include applied cryptology, cryptographic protocols, information and communication security.