Applying LU Decomposition of Matrices to Design Anonymity Bilateral Remote User Authentication Scheme

Xiong Li,1,2 Jianwei Niu,2 Muhammad Khurram Khan,3 and Zhibo Wang4

1 School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
2 State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China
3 Center of Excellence in Information Assurance, King Saud University, Riyadh 11653, Saudi Arabia
4 College of Software, East China Institute of Technology, Nanchang 330013, China

Correspondence should be addressed to Jianwei Niu; niujianwei@buaa.edu.cn

Received 10 October 2012; Revised 9 January 2013; Accepted 9 January 2013

Academic Editor: Oleg V. Gendelman

Copyright © 2013 Xiong Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We apply LU decomposition of matrices to present an anonymous bilateral authentication scheme. This paper aims at improving security and providing more excellent performances for remote user authentication scheme. The proposed scheme can provide bilateral authentication and session key agreement, can quickly check the validity of the input password, and can really protect the user anonymity. The security of the proposed scheme is based on the discrete logarithm problem (DLP), Diffie-Hellman problem (DHP), and the one-way hash function. It can resist various attacks such as insider attack, impersonation attack, server spoofing attack, and stolen smart card attack. Moreover, the presented scheme is computationally efficient for real-life implementation.

1. Introduction

The remote user authentication scheme allows the user and the remote server to mutually authenticate each other over public network environments, and then the authorized user can access the services and resources which are provided by the remote server. Generally, the password-based authentication scheme provides an efficient and secure way for mutual authentication and allows the user and the server to establish a shared session key for future secret communication. Since then, the researchers have proposed many password-based remote user authentication schemes [2–7] to ensure the secure communication through the public network, and also many studies [8–18] have been presented to enhance the security or improve the computation and communication costs of the remote user authentication scheme.

In the public network environments, it is important to ensure user anonymity such that the user’s real identity can only be revealed by authorized entities. In 2000, Lee and Chang [19] proposed a user identification scheme with key distribution preserving user anonymity for the distributed computer network. However, Wu and Hsu [20] pointed out that Lee and Chang’s scheme cannot protect user anonymity as they claimed, and they proposed an enhanced scheme. Later, Yang et al. [21] showed that Wu and Hsu’s scheme cannot resist impersonation attack and proposed an improved scheme which is more secure and efficient. Unfortunately, Mangipudi and Katti [22] presented that Yang et al.’s protocol is vulnerable to a Denial-of-Service (DoS) attack and proposed a secure identification and key agreement protocol with user anonymity. Recently, Wang et al. [23] presented a secure and efficient identification and key agreement protocol with user anonymity based on the difficulty of computing the elliptic curve Diffie-Hellman. Their scheme’s computation cost is lower and is suitable for applications in low power computing environments.

In 2004, Choi and Youn [24] proposed a novel data encryption and distribution approach using LU decomposition of matrices. Then, Pathan et al. [25, 26] proposed two efficient bilateral remote user authentication schemes based on LU decomposition of matrices. Nevertheless, these schemes have several weaknesses, such as they cannot resist replay attacks, they cannot preserve the user anonymity, the server and...
mathematical problems in engineering

users cannot agree on a session key, and so forth. To address these issues, Tseng et al. [27] proposed a user authentication scheme based on LU decomposition of matrices. They claimed that their scheme can resist replay attack, forgery attack, and insider attack and provide user anonymity. Whereas, after careful analysis, we find that Tseng et al.'s scheme is still vulnerable to insider attack, stolen smart card attack and inefficient for wrong password login and does not provide user anonymity. To overcome these existing weaknesses of Tseng et al.'s scheme, we propose a novel bilateral authentication scheme with user anonymity using LU decomposition of matrices. Analysis shows that our scheme not only can provide better security properties but also is more efficient than the other authentication schemes.

The rest of this paper is organized as follows: Section 2 introduces the necessary preliminaries of this paper. The brief review of Tseng et al.'s scheme is provided in Section 3. Section 4 describes a cryptanalysis of Tseng et al.'s scheme. A brief review of Tseng et al.'s scheme is provided in Section 3. The proposed scheme and the corresponding analysis are introduced in Section 4. The rest of this paper is organized as follows: Section 2 introduces the necessary preliminaries of this paper. Section 3 briefly reviews Tseng et al.'s scheme. Section 4 describes a cryptanalysis of Tseng et al.'s scheme. The proposed scheme and the corresponding analysis are made available in Section 5. The conclusions are presented in Section 7.

2. Preliminaries

In this section, we introduce some basic information about the LU decomposition of matrices and Discrete logarithm problem, and they are the mathematical basis of our proposed bilateral remote user authentication protocol with user anonymity.

2.1. LU Decomposition of Matrices. From the matrix theory, LU decomposition factorizes a matrix as the product of a lower triangular matrix and an upper triangular matrix. Let $A$ be a square matrix; an LU decomposition of matrix $A$ is the form $A = LU$, where $L$ is a lower triangular matrix and $U$ is an upper triangular matrix. This means that $L$ has only zeros above the diagonal and $U$ has only zeros below the diagonal. For example, for a $4 \times 4$ matrix $A$, its LU decomposition looks like

$$
A = \begin{pmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44}
\end{pmatrix} = 
\begin{pmatrix}
l_{11} & 0 & 0 & 0 \\
l_{21} & l_{22} & 0 & 0 \\
l_{31} & l_{32} & l_{33} & 0 \\
l_{41} & l_{42} & l_{43} & l_{44}
\end{pmatrix} \begin{pmatrix}
u_{11} & u_{12} & u_{13} & u_{14} \\
u_{22} & u_{22} & u_{23} & u_{24} \\
u_{33} & u_{33} & u_{34} \\
u_{44}
\end{pmatrix}.
$$

(1)

If $A$ is a singular matrix of rank $k$, it admits an LU decomposition if all the $k$-leading principal minors are nonzero.

In the identity authentication system, we assume that $n$ is the number of users the system can support. We can introduce the LU decomposition into the user authentication system to ensure the security of the system. In the system initialization phase, the remote server generates a symmetric matrix $A_{n\times n}$ as his/her master secret key. With the LU decomposition, the server can separate the symmetric key matrix $A_{n\times n}$ to the product of a lower triangular matrix $L_{n\times n}$ and an upper triangular matrix $U_{n\times n}$, that is, $A_{n\times n} = L_{n\times n}U_{n\times n}$, and stores these matrices in other servers.

Since $A$ is a symmetric matrix, we have that $a_{ij} = a_{ji}$, for $1 \leq i \leq n$ and $1 \leq j \leq n$, and the product of the $x$th row of matrix $L_{n\times n}$ and the $y$th column of matrix $U_{n\times n}$ is equal to the product of the $y$th row of matrix $L_{n\times n}$ and the $x$th column of matrix $U_{n\times n}$. For example, suppose $A$ is a $4 \times 4$ symmetric matrix with LU decomposition as follows:

$$
A = \begin{pmatrix} 1 & 2 & 3 & 4 \\
2 & 5 & 8 & 7 \\
3 & 8 & 11 & 9 \\
4 & 7 & 9 & 6
\end{pmatrix}.
$$

(2)

We can perform elementary row operations to get the lower matrix $L$ and upper matrix $U$ as follows:

$$
L = \begin{pmatrix} 1 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 \\
3 & 2 & 1 & 0 \\
4 & -1 & 1 & 2
\end{pmatrix},
$$

(3)

and

$$
U = \begin{pmatrix} 1 & 2 & 3 & 4 \\
0 & 1 & 2 & -1 \\
0 & 0 & -2 & -1 \\
0 & 0 & 0 & -21/2
\end{pmatrix}.
$$

(4)

Given $i = 3$ and $j = 4$, we can compute $a_{34}$ and $a_{43}$ as follows:

$$
a_{34} = L_L (3) \times U_C (4) = (3 \ 2 \ 1 \ 0) \times (4 \ -1 \ -1 \ -21/2)^T = 9,
$$

(4)

$$
a_{43} = L_L (4) \times U_C (3) = (4 \ -1 \ 1/2 \ 1) \times (3 \ 2 \ -2 \ 0)^T = 9,
$$

where $L_L (3)$ denotes the 3rd row of the matrix $L$ and $U_C (4)$ denotes the 4th column of the matrix $U$, and we have that $a_{34} = a_{43}$.

2.2. Discrete Logarithm Problem. The detailed information about discrete logarithm problem can be found in the literature [28], and we briefly introduce the discrete logarithm problem as follow. In a multiplicative group $(g)$ of order $q$, where $p = 2q + 1$ is the modulus for the group, both $p$ and $q$ are public large prime numbers. This implies

1. $(g) = \{X | X = g^x \mod p,$ for $x = 1, 2, 3 \}$ is a finite set of size $q$, where $2 \leq q \leq p - 1$ and $g^q \mod p = 1$.

2. Given $x$ and $g$, computing the modular exponentiation $X = g^x \mod p$ is relatively easy. However, given $X$ and $g$, it is computationally infeasible to find $x$ such that $X = g^x \mod p$; namely, in $(g)$, the discrete logarithm problem is intractable [28].
Moreover, given \( g, p, X = g^x \mod p \) and \( Y = g^y \mod p \), computing \( K = g^{xy} \mod p \), which is known as the Diffie-Hellman problem, is also intractable [28].

### 3. Review of Tseng et al.’s Scheme

In this section, we briefly review Tseng et al.’s scheme, and more details can be found in [27]. Tseng et al.’s scheme contains four phases, that is, the registration phase, the login phase, the authentication phase, and the password change phase. The notations used throughout this paper are listed in Table 1.

#### 3.1. Registration Phase

Suppose a new user \( U_i \) with the identity ID\(_i\) wants to register himself/herself with the server for access the remote services. \( U_i \) randomly chooses his/her password PW\(_i\) and sends ID\(_i\), \( h(PW_i) \) to the server through a secure channel. Upon receiving the registration request message, the server takes the following steps:

1. Generates two random numbers \( x_i, y_i \) between 1 and \( n \). Then selects the \( x_i \)th row from the matrix \( L \) (denoted as \( L_{x_i}(x_i) \)), the \( y_i \)th column from matrix \( U \) (denoted as \( U_{C}(x_i) \)), and the \( y_i \)th column from the matrix \( U \) (denoted as \( U_{C}(y_i) \));

2. Computes \((K_{x_iy_i}, \theta_i, v_i)\) as follows:
   \[
   K_{x_iy_i} = L_{x_i}(x_i) \times U_{C}(y_i),
   \theta_i = h(ID_i \oplus K_{x_iy_i}) \oplus h(PW_i) \oplus h(K_S),
   v_i = h(K_S) \oplus y_i;
   \]

3. Stores \((K_{x_iy_i}, \theta_i, v_i, U_{C}(x_i), v_i, h(\cdot), g, p)\) into a smart card and submits the smart card to \( U_i \) via a secure channel.

#### 3.2. Login Phase

When \( U_i \) wants to login into the system, \( U_i \) first inserts the smart card to the card reader and inputs his/her password PW\(_i\)\(^*\). The smart card performs the following steps to generate the login request message:

1. Generates a random number \( r_i \);

2. Computes \( H_i = K_{x_iy_i} \oplus h(r_iT) \) and \( S_i = \theta_i \oplus h(PW_i) \oplus r_i \) where \( T \) is the current timestamp;

3. Generates a random number \( a \) and then computes \( r_i = g^a \mod p \) and \( R_i = h(\theta_i \oplus r_i) \);

4. Encrypts \((ID_i, r_i, U_{C}(x_i), y_i, T)\) with \( R_i \) and computes \( C_i = \theta_i \oplus h(ID_i \oplus K_{x_iy_i}) \oplus h(PW_i) \oplus r_i = h(K_S) \oplus r_i \);

5. Sends the login request message \( M_i = (C_i, E_{K_i}(ID_i, r_i, U_{C}(x_i), v_i, T), H_i, S_i, T) \) to the server.

#### 3.3. Authentication Phase

Upon receiving the login request message \( M_i \), the server and the user \( U_i \) perform the following operations for mutual authentication:

1. The server computes \( R_i = C_i \oplus h(K_S) \) and decrypts \( E_{K_i}(ID_i, r_i, U_{C}(x_i), y_i, T) \) with \( R_i \).

2. The server checks the validity of ID\(_i\). If ID\(_i\) is invalid, the server rejects the login request.

3. The server verifies whether the time interval \((T' - T) \leq \Delta T\), where \( T' \) is the current timestamp when the server received the message. If \((T' - T) \geq \Delta T\), the login request is considered out of date and is rejected.

4. The server computes \( y_i = v_i \oplus h(K_S) \).

5. The server computes \( K_{y_i, x_i} = L_{y_i}(y_i) \times U_{C}(x_i), t = h(ID_i \oplus K_{y_i, x_i}), \) and \( r' = S_i \oplus T \oplus h(K_S) \oplus t \).

6. The server verifies whether \( K_{y_i, x_i} \) equals \( H_i \oplus h(r') \). If not, the server rejects the login request. Otherwise, it proceeds to the next steps.

7. The server generates a random number \( b \) and computes \( r_i = g^b \mod p \).

8. The server computes the authenticated session key \( AK_i = r_i^b = g^{ab} \mod p \).

9. At last, the server sends \( E_{K_i}(K_{y_i, x_i} \oplus r_i, r_i + 1, T''') \) to \( U_i \), where \( T''' \) is the current timestamp.

10. When receiving the message \( E_{K_i}(K_{y_i, x_i} \oplus r_i, r_i + 1, T''') \), \( U_i \) decrypts the message, gets \( K_{y_i, x_i} \oplus r_i \), and verifies whether \((T''' - T'') \leq \Delta T\), where \( T''' \) is the current timestamp. If so, \( U_i \) proceeds to the next steps.

11. \( U_i \) checks whether decrypted data contain the value \( r_i + 1 \). If so, \( U_i \) uses \( K_{y_i, x_i} \) to compute \( r_s = (K_{y_i, x_i} \oplus r_i) \oplus K_{y_i, x_i} \).
(12) $U_i$ generates the authenticated session key $AK_i$ as $AK_i = r^a_i = (g^b)^a = g^{ab} \mod p$. Then $U_i$ can communicate with the server secretly by using $AK_i$.

### 3.4. Password Change Phase

When $U_i$ wants to change his password $PW_i$ to $PW'_i$, he sends $(ID_i, h(PW_i), h(PW'_i))$ to the server. Upon receiving the password-changing message, the server takes the following steps:

1. computes $\theta'_i = \theta_i \oplus h(PW'_i) \oplus h(PW_i) = h(ID_i \oplus K_{x,y_i}) \oplus h(PW'_i) \oplus h(K_S)$;
2. replaces $\theta_i$ with $\theta'_i$ in the smart card.

### 4. Cryptanalysis of Tseng et al.’s Scheme

Tseng et al. claimed that their scheme can protect user anonymity and can resist various known attacks. However, after careful analysis, we find that their scheme cannot completely protect the user anonymity and is vulnerable to insider attack, server spoofing attack. Besides, their scheme is inefficient for wrong password login. We analyze the security weaknesses of Tseng et al.’s scheme as below.

#### 4.1. Attacks against the User Anonymity

In order to prevent the attacker from tracking the user’s movements, it is important to ensure user anonymity such that the user’s real identity can only be recognized by the server.

Kocher et al. [29] and Messerges et al. [30] have pointed out that the confidential information stored on the smart card can be extracted by physically monitoring its power consumption. So, in Tseng et al.’s scheme, a legal but malicious user $U_A$ can extract information $(K_{x,y_i}, \theta_A, U_C(x_A), V_A, h(\cdot), p, g)$ from his/her own smart card, and with his/her own identity $ID_A$ and password $PW_A$, he/she can compute the value of $h(K_S) = \theta_A \oplus h(ID_A \oplus K_{x,y_i}) \oplus h(PW_A)$. When the valid login request message $M_i = (C_i, E_R(ID_i, r_i, U_C(x_i), \theta_i, V_i, T_i, H_i, S_i, T)$ of a legal user $U_i$ was to be intercepted by this malicious user $U_A$ from the public communication channel, the malicious user $U_A$ can compute $R_i = C_i \oplus h(K_S)$, and then he/she can decrypt $E_R(ID_i, r_i, U_C(x_i), \theta_i, V_i, T_i)$ using $R_i$ to obtain $(ID_i, r_i, U_C(x_i), \theta_i, V_i, T_i)$ of the user $U_i$. Obviously, the malicious user $U_A$ can obtain the real identity $ID_i$ of the user $U_i$. From the above discussion, we can see that Tseng et al.’s scheme cannot really protect user anonymity.

#### 4.2. Insider Attack

In the registration phase of Tseng et al.’s scheme, the user $U_i$ sends $ID_i, h(PW_i)$ to the server for registration, and this information can be acquired by the privileged insider. However, in password change phase, the server simply not checks the validity of user $U_i$’s ID and $PW_i$. So, this privileged insider of the remote system can masquerade as the user $U_i$ to send the triple $(ID_i, h(PW_i), h(PW'_i))$ to the server to perform the password-changing phase. Upon receiving the password-changing message, the server takes the following steps:

1. computes $\theta'_i = \theta_i \oplus h(PW'_i) \oplus h(PW_i) = h(ID_i \oplus K_{x,y_i}) \oplus h(PW'_i) \oplus h(K_S)$;
2. replaces $\theta_i$ with $\theta'_i$ in the smart card.

Therefore, since the server does not check the validity of the user’s identity and password when the user wants to change his/her password, Tseng et al.’s scheme is vulnerable to insider attack, and the privileged insider can easily change the legal user’s password.

#### 4.3. Stolen Smart Card Attack

Stolen smart card attack is that if the user’s smart card is lost or stolen, the attacker can extract the information stored in the smart card and can easily change the password of the smart card, which can guess the password of the user by using password guessing attacks, or can impersonate the user to login to the system.

In Tseng et al.’s scheme, a legal but malicious user $U_A$ having his own smart card can gather information $(K_{x,y_i}, \theta_A, U_C(x_A), V_A, h(\cdot), p, g)$ from his own smart card, and he/she can get the value of $h(K_S)$ as shown in Section 4.1. Now the malicious user $U_A$ can intercept a valid login request message $M_i = (C_i, E_R(ID_i, r_i, U_C(x_i), \theta_i, V_i, T_i, H_i, S_i, T)$ of the legal user $U_i$ from the public communication channel. Then the malicious user $U_A$ can compute $R_i = C_i \oplus h(K_S)$ and can get the identity $ID_i$ of user $U_i$ by decrypt $E_R(ID_i, r_i, U_C(x_i), \theta_i, V_i, T_i)$ using $R_i$. In case the user $U_i$’s smart card is stolen by this malicious user $U_A$, he/she can extract the information $(K_{x,y_i}, \theta_i, U_C(x_i), V_i, h(\cdot), g, p)$ from the memory of the smart card. With the information $ID_i, h(K_S), K_{x,y_i}$, and $\theta_i$, the malicious user $U_A$ can guess $U_i$’s password by the following processes.

1. The attacker computes $h(PW'_i) = \theta_i \oplus h(ID_i \oplus K_{x,y_i}) \oplus h(K_S)$.
2. The attacker chooses a password $PW'_i$ from a uniformly scattered dictionary.
3. The attacker computes $h(PW'_i)$ and verifies the correctness of password $PW'_i$ by checking whether $h(PW'_i)$ equals $h(PW_i)$ or not.
4. The attacker repeats steps (2) and (3) until $h(PW'_i)$ equals $h(PW_i)$ to guess a correct password.

After getting the correct $ID_i$ and $PW_i$, the malicious user $U_A$ can easily change the password of user $U_i$ and can impersonate $U_i$ to login to the system.

#### 4.4. Inefficient for Wrong Password Login

Generally speaking, in practical applications, the user $U_i$ may keep different passwords for different applications to ensure security. Users are easy to confuse the password such that the user cannot match the application with the correct password; in other words, it is possible that the user enters a wrong password in the login phase.

In the login phase, the smart card does not verify the correctness of the entered password by the user. If the user $U_i$ inputs a wrong password $PW'_i (\neq PW_i)$ by mistake, the smart card and the server will perform the following steps:

1. generates a random number $r$, gets the current timestamp $T$, and computes $H_i = K_{x,y_i} \oplus h(r \oplus T)$ and $S_i = \theta_i \oplus h(PW'_i) \oplus r$.
(2) generates a random number \( a \) and computes \( r_i = g^a \mod p \) and \( R_i = h(\theta_i \oplus r_i) \);

(3) encrypts \((ID_i, r_i, U_C(x_i), v_i, T)\) with \( R_i \) and computes \( C_i = \theta_i \oplus h(ID_i \oplus K_{x,y} \oplus h(PW_i) \oplus R_i = h(K_\delta) \oplus h(PW_i) \oplus h(PW_i') \oplus R_i) \);

(4) sends the login request message \( M_i = (C_i, E_R(ID_i, r_i, U_C(x_i), v_i, T), H_i, S_i, T) \) to the server;

(5) when the server receives the login request message \( M_i \), the server computes \( R'_i = C_i \oplus h(K_\delta) = h(K_\delta) \oplus h(PW_i) \oplus h(PW_i') \oplus R_i \).

(6) when the server decrypts \( E_R(ID_i, r_i, U_C(x_i), v_i, T) \) using \( R'_i \), the server will find that the user \( U_i \)'s identity is invalid. Thus, the server rejects the user \( U_i \)'s login request.

In this case, the user \( U_i \) is unaware of the fact that he/she has entered his/her password incorrectly in the login phase, which results in unnecessary extra communication and computation costs.

5. The Proposed Scheme

In this section, we apply the LU decomposition of matrices to design a novel bilateral remote user authentication scheme with user anonymity, where LU decomposition of matrices ensures secretly information exchange between the user and the server, and enhances the security of the authentication scheme. To initiate the scheme, the server chooses a \( n \times n \) symmetric matrix \( A \) with LU decomposition as \( A = LU \) and secretly stores these matrices as his/her secret key in other servers, where \( n \) is the number of users the system can support. The server chooses a secret key \( K_\delta \) with 256 bits, which makes the \( K_\delta \) have a high entropy and can resist brutal force attack. The proposed scheme also contains four phases, that is, the registration phase, the login phase, the authentication and key agreement phase, and the password change phase. The proposed scheme contains the timestamps, so the authentication system needs to deploy a mechanism such as NTP (Network Time Protocol) to ensure clock synchronization between the user and the remote server. The detailed information about these phases is described as follows and also shown in Figure 1.

5.1. Registration Phase. When a user \( U_i \) wants to become a legal user of the system, \( U_i \) generates his own identity \( ID_i \) and easy-to-remember password \( PW_i \) and selects and remembers a random number \( b_i \) (the bit length of \( b_i \) is assumed to be 128). Then, \( U_i \) computes \( RPW_i = h(b_i \| PW_i) \) and submits \( ID_i \) and \( RPW_i \) to the server over a secure communication channel for registration. \( h(\cdot) \) used throughout the proposed scheme is a collision-free one-way hash function such as SHA-1 [31], which maps any message with the length less than \( 2^{64} \) bit to a 160-bit message digest. Upon receiving the registration request message, the server \( S \) and the user \( U_i \) take the following steps:

(1) The server \( S \) computes \( A_i = h(ID_i \| K_\delta) \), \( B_i = A_i \oplus h(ID_i \oplus RPW_i) \), and \( C_i = h(A_i) \).

(2) The server \( S \) chooses two random numbers \( x_i, y_i \in [1, n] \) and computes \( K_{x,y} = L_g(x_i) \times U_C(y_i) \), \( D_i = h(K_{x,y} \oplus K_\delta) \oplus h(ID_i \| RPW_i) \), \( E_i = h(K_\delta) \oplus y_i \), where the meaning of symbols \( L_g(x_i) \) and \( U_C(y_i) \) are the same as in Section 3.1.

(3) The server stores \( \{B_i, C_i, K_{x,y}, D_i, E_i, U_C(x_i), g, p, b_i\} \) into a smart card and issues the smart card to \( U_i \) via a secure channel, where \( p \) is a big prime and \( q = 2q+1 \), \( q \) is also a big prime, and \( g \) is a primitive element with \( g^q = 1 \mod p \).

(4) At last, in order to facilitate the subsequent verification, the user \( U_i \) enters the remembered random number \( b_i \) into the smart card, and the smart card contains \( \{B_i, C_i, K_{x,y}, D_i, E_i, U_C(x_i), g, p, b_i\} \).

5.2. Login Phase. When the user \( U_i \) wants to login to the system, \( U_i \) inserts his/her smart card into the card reader and inputs his/her identity \( ID_i \), password \( PW_i \). Then the smart card performs the following operations:

(1) Computes \( RPW_i = h(b_i \| PW_i) \), \( A_i = B_i \oplus h(ID_i \oplus RPW_i) \), and \( C_i = h(A_i) \) and compares \( C_i' \overset{?}{=} C_i \). If they are equal, it means the user inputs the right identity and password. Otherwise, the input identity or password is not valid, and the smart card terminates the session;

(2) generates a random number \( r \), gets current timestamp \( T \), and computes \( F_i = D_i \oplus h(ID_i \| RPW_i) \), \( G_i = F_i \oplus r \), \( CID_i = r \oplus ID_i \), and \( H_i = E_i \oplus T \);

(3) generates a random number \( r_i \in (1, q) \) and computes \( R_i = g^r \mod p \), \( J_i = R_i \oplus h(r \oplus T) \), and \( M_i = h(h(ID_i \| K_{x,y}) \| R_i \| T) \), where \( M_i \) is used to resist the forgery attack such that any change of the login request message is invalid login message;

(4) submits the login request message \( \{G_i, CID_i, H_i, J_i, U_C(x_i), M_i, T\} \) to the server.

5.3. Verification and Key Agreement Phase. Upon receiving the login request message \( \{G_i, CID_i, H_i, J_i, U_C(x_i), M_i, T\} \), the server performs the following steps for mutual authentication and key agreement:

(1) The server \( S \) verifies the validity of the time interval between \( T' \) and \( T \). If \( (T' - T) \geq \Delta T \), \( S \) rejects the login request. Here \( T' \) is the timestamp, when the login request message was received, and \( \Delta T \) is the expected valid time interval for transmission delay.

(2) The server \( S \) computes \( y_i = H_i \oplus h(K_\delta) \oplus T \), \( K_{x,y} = L_g(y_i) \times U_C(x_i) \).

(3) The server \( S \) computes \( r' = h(K_{x,y} \oplus K_\delta) \oplus G_i \), \( R'_i = J_i \oplus h(r' \oplus T) \), and \( ID_i = CID_i \oplus r' \) and checks whether \( ID_i \) is the registered identity of a valid user. If so, the server \( S \) performs the following steps. Otherwise, the session is terminated.
(4) The server S computes $M_j' = h(h(ID_j||K_{y_i}||r')||T)$ and checks $M_j' = M_j$. If they are equal, the validity of the user $U_j$ is authenticated by the server S. Otherwise, the session is terminated by the server S.

(5) For achieving mutual authentication, the server chooses a random number $r_j \in (1, q)$, gets the current timestamp $T''$, and computes $R_j = g^{r_j} \mod p$, $K_j = R_j \oplus R_j'$, and $M_j = h(h(ID_j||K_{y_i}||r')||R_j'\||T'')$.

(6) The server S submits the reply message $\{K_j, M_j, T''\}$ to the user $U_j$ for mutual authentication.

(7) After receiving the mutual authentication message $(K_j, M_j, T'')$, the smart card verifies the validity of the time interval between $T''$ and $T'''$. If $T''' - T'' \geq \Delta T$, the user $U_j$ terminates the session. Here $T'''$ is the timestamp, when the mutual authentication message was received.

(8) The smart card computes $R_j' = K_j \oplus R_j$, $M_j'' = h(h(ID_j||K_{y_i}||R_j')\||T''')$ and checks $M_j'' = M_j$. If they are equal, the server S is authenticated by the user $U_j$, and the server S and user achieve mutual authentication. Otherwise, the smart card terminates this session.

(9) At last, the user $U_j$ and the server S can compute $AK_j = R_j'' \mod p = g^{r_j''} \mod p$ and $AK_j = R_j'' \mod p = g^{r_j''} \mod p$, respectively, as their shared session key for future secret communication.

5.4. Password Change Phase. When the user $U_j$ wants to renew his/her password to $PW_{jnew}$, the user $U_j$ can update...
his/her password by performing the following steps without communicating with the server $S$.

1. The user $U_i$ inserts his smart card into a card reader and inputs his identity $ID_i$ and old password $PW_i$ and requests to change his/her password.

2. The smart card computes $RPW_i = h(b_i || PW_i)$, $A_i = B_i @ h(ID_i @ RPW_i)$, $C'_i = h(A'_i)$, and compares $C'_i = C_i$. If they are not equal, the password change request is rejected. Otherwise the user $U_i$ inputs a new password $PW_i^{new}$.

3. The smart card computes $RPW_i^{*} = h(b_i || PW_i^{new})$, $B_i^* = A_i @ h(ID_i @ RPW_i^{*})$, and $D_i^* = D_i^* @ h(ID_i || RPW_i^{*})$.

4. Finally, the smart card replaces $B_i$ and $D_i$ with $B_i^*$ and $D_i^*$, respectively, to update his/her password.

### 6. Analysis of the Proposed Scheme

In this section, we first discuss the security features of the proposed anonymity bilateral authentication scheme. Then we evaluate the performance and functionality of our proposed scheme and make comparisons with Tseng et al.’s scheme.

#### 6.1. Security of Session Key

**6.1.1. Known-Key Secrecy.** Known-key secrecy means that compromise of one session key should not compromise other session keys. In our scheme, the session key $AK_i = g^{r_i}$ mod $p$ is associated with $r_i$, $g^{r_i}$ mod $p$, $r_i$, and $g^{r_i}$ mod $p$. According to discrete logarithm problem (DLP) and Diffie-Hellman problem (DHP), knowing a session key $AK_i = g^{r_i}$ mod $p$ and the random number $r_i$, $r_i$ is useless for computing the other session keys $AK_i' = g^{r_i'}$ mod $p$ without knowing $r_i$ and $r_i'$. It is impossible for an attacker to compute the other session key $AK_i'$, and the proposed scheme provides known-key security.

**6.1.2. Forward Secrecy.** Forward secrecy means that if the long-term secret keys (e.g., the server’s secret key $K_S$ and user’s password $PW_i$) are compromised, the secrecy of previously established session keys should not be affected. In our scheme, we assume that the master secret key $K_S$ and the password $PW_i$ of user $U_i$ are compromised for some reasons, and the attacker gets the previous communication message $\{G_i, CID_i, H_i, I_i, U_C(x_i), M_i, T\}$ and $\{K_{x_j}, M_j, T''\}$ from the public channel; then the attacker can get $y_i = H_i @ h(K_S) @ T$. However, since the secret matrix $L$ has been maintained only by the server $S$, the attacker cannot compute $K_{x_j} @ r$ and has no way to know $r_i$, $R_i$, $r_j$, and $R_j$. Therefore, the attacker has no way to get the previous session key $AK_i = g^{r_i}$ mod $p$, and our scheme can ensure perfect forward secrecy.

**6.2. Protect User Anonymity.** In the login phase and authentication phase of the proposed scheme, the real identity $ID_i$ of user $U_i$ is not transmit via plain text form. If the login request message $\{G_i, CID_i, H_i, I_i, U_C(x_i), M_i, T\}$ and the mutual authentication message $\{K_{x_j}, M_j, T''\}$ are eavesdropped by an attacker from the public channel, the attacker has to get the random number $r$ to compute the real identity $ID_i$. However, the attacker has no way to know $K_{x_j}$ and $K_S$, so he/she has no valid method to get the random number $r$ and cannot reveal the real identity $ID_i$ of the user $U_i$. Therefore, our scheme can really protect user anonymity.

**6.3. Resist Impersonation Attack.** In this type of attack, in order to impersonate as a legitimate user, the attacker or a malicious user has to forge a valid login request message $\{G_i, CID_i, H_i, I_i, U_C(x_i), M_i, T\}$ using the previously eavesdropped messages or the information obtained from the lost smart card. However, in the proposed scheme, the attacker and any malicious user $U_A$ cannot forge a valid login request message, since he/she has no knowledge of $ID_i$, $RPW_i$, $E_i$, and $K_{x_j}$, so he/she cannot impersonate as the legitimate user $U_i$.

In addition, even if the adversary or a malicious user has obtained the smart card of user $U_i$ and extracts the parameters $\{B_i, C_i, K_{x_j}, D_i, E_i, U_C(x_i), g, p, b_i\}$ which are stored in the smart card by some way, he/she still cannot forge a valid login request message, since he/she has no way to get the valid $ID_i$, $PW_i$, where they are all protected by the one-way hash function.

Therefore, the proposed protocol is secure against impersonation attack. At the same time, the attacker cannot get the valid $ID_i$, $PW_i$, so the proposed protocol can resist the denial of service attack.

**6.4. Resist Insider Attack.** In the registration phase of the proposed scheme, the user $U_i$ freely selects his/her password $PW_i$ and submits the masked password $RPW_i = h(b_i @ PW_i)$ instead of $h(PW_i)$ to the server for registration. In the proposed scheme, the password must first be verified by the smart card in login and password change phase, only the adversary gets the valid password $PW_i$ of the user $U_i$, and he/she can impersonate the user $U_i$ to access service. However, if the insider of the remote system gets the information $ID_i$ and $RPW_i = h(b_i @ PW_i)$, he/she cannot obtain the password $PW_i$ since it is protected by the one-way hash function and cannot impersonate the user $U_i$ to login to the system or change the user’s password. Therefore, the proposed scheme can resist insider attack properly.

**6.5. Resist Stolen Smart Card Attack.** Assume that the user $U_i$’s smart card has been lost or stolen, the attacker can extract the stored information $\{B_i, C_i, K_{x_j}, D_i, E_i, U_C(x_i), g, p, b_i\}$ from the smart card using differential power analysis [29] and simple power analysis [30]. Even after gathering these information, in order to change the user’s password or login into the system by using the lost smart card, the attacker has to get real identity $ID_i$ and the password $PW_i$ correctly at the same time. However, because the attacker has not the knowledge of the master secret key $K_S$ and meanwhile the $ID_i$ and the $PW_i$ are protected by one-way hash function, it is not possible for an attacker to guess the $ID_i$ and the $PW_i$ correctly.
In this section, we pointed out the security weaknesses of Tseng et al.’s scheme, that is, their scheme is vulnerable to insider attack and stolen smart card attack. Therefore, we evaluated the performance and functionality of our proposed scheme and make comparisons with Tseng et al.’s scheme. In order to facilitate the computational complexity analysis of the scheme, we define the following notations.

\[ T_h \]: the time for executing a one-way hash function \( h(\cdot) \),

\[ T_M \]: the time for performing a vector multiplication operation,

\[ T_{\text{EXP}} \]: the time for performing an exponentiation operation,

\[ T_{\text{ENC}} \]: the time for performing a symmetric encryption operation, and

\[ T_{\text{DEC}} \]: the time for performing a symmetric decryption operation.

Because exclusion-OR operation requires very few computations, we neglect considering its computational cost in this paper. We list the result of performance comparison in Table 2, and we can see that the total computational cost of our scheme and Tseng et al.’s scheme are \( 2T_M + 18T_H + 4T_{\text{EXP}} \) and \( 2T_M + 9T_H + 4T_{\text{EXP}} + 2T_{\text{ENC}} + 2T_{\text{DEC}} \), respectively. Since the symmetric cryptosystem needs more computational costs than the one-way hash functions, our scheme is more efficient than Tseng et al.’s scheme.

Table 3 shows the functional comparison of our proposed scheme and Tseng et al.’s scheme. Compared with Tseng et al.’s scheme, our scheme can resist various attacks and can really protect user anonymity. Besides, our scheme can quickly check the validity of the password in the very beginning of login phase. Therefore, our scheme is more secure and efficient than Tseng et al.’s scheme.

7. Conclusions

In this paper, we have applied the LU decomposition of matrices to present a novel anonymity bilateral authentication scheme. First, we pointed out the security weaknesses of Tseng et al.’s scheme, that is, their scheme is vulnerable to insider attack and stolen smart card attack, is inefficient...
for wrong password login, and does not really provide user anonymity. To surmount these identified weaknesses, we have proposed a novel scheme using the LU decomposition of matrices to reduce computational complexity and improve security, where LU decomposition of matrices ensures secretly information exchange between the user and the server, and enhances the security of the authentication scheme. Hence, our proposed protocol is more efficient and practical.

Acknowledgments

The authors are grateful to the editor and anonymous reviewers for their valuable suggestions which improved the paper. This work was supported by the Research Fund of the State Key Laboratory of Software Development Environment under Grant no. BUAA SKLSDE-2012ZX-17, the National Natural Science Foundation of China under Grant no. 61271041, 61702965, and 61101920, and the Program for New Century Excellent Talents in University under Grant no. NECT-09-0028.

References


