An anonymous user identification and key distribution technical protocol

Fuw-Yi Yang          Zhen-Wei Liu
Department of Computer Science and Information Engineering,
Chaoyang University of Technology
yangfy@cyut.edu.tw   s9727628@cyut.edu.tw

Abstract

Owing to the dynamic development of the Internet and information technology, user can now access information and services provided by remote service provider through the Internet. The Internet is an open medium of communication; hence, in order to ensure private communication, this study designs an "anonymous user identification protocol" to determine user identities, as well as negotiating a private key for information encryption to prevent unauthorized user from accessing the communication data. Previous similar protocols employed timestamps to prevent replay attacks, requiring both sides of the communication to synchronize their clocks and causing a considerable burden on distributed systems. This study uses random numbers instead of synchronized clocks to prevent replay attacks, and thus reduces system construction and maintenance costs, besides being more applicable in distributed computer network systems. User identification has been reduced from four rounds of communications to three, providing evidence for the improvement in recognition efficiency and more effective bandwidth management.

Keywords: user identification, mutual authentication, anonymity, Diffie–Hellman problem

1. Introduction

With the vigorous development of network technology and the rising popularity of the Internet, the use of e-commerce related services has become prevalent. In the distributed network environment, it is crucial to secure communications that take place over an insecure channel. User identification and private key distribution are therefore crucial components in a distributed network environment.

In 1976, Diffie and Hellman [6] pioneered the concept of public-key cryptography. The main purpose of the key exchange system is that when two entities want to conduct communications through the Internet, both participants can obtain a shared key through modular exponentiation to ensure the security of communication; this security is established based on the solution of the Diffie–Hellman problem. However, this protocol is vulnerable to man-in-the-middle attacks, because, the transmission end and receiving end do not undergo user identification prior to the communication, and the attacker can easily impersonate one of the roles to deceive either party.

Lee and Chang proposed an anonymous user identification scheme [8] in 2000. Their scheme provides user identification and key exchange simultaneously while protecting user’s personal information from disclosing. As indicated by Wu and Hsu [13], there are security leaks in scheme [8]. Due to one-way authentication, an attacker can masquerade as the service provider and share a session key with the user. Also with compromised session key, enables an attacker
to compute the identity of user who completed the login procedure. Literature [12] not only presents security analysis but also proposes improvement. But their improvement still remains security leaks, attacker and service provider can compute secret token of a victim user, as shown in literatures [14-15]. As an amendment, Yang et al. [14] proposed an efficient user identification and key distribution scheme, providing user anonymity despite communicating in an open, distributed network environment. However, Mangipudi and Katti [10] indicated that Yang et al.’s proposal is vulnerable to denial-of-service attacks, suggesting that an attacker can tamper with the information sent by a user and causing the server to reject the legitimate user request; Mangipudi and Katti therefore proposed an improved scheme that utilized digital signatures, which are not easily alterable, to protect messages to be sent from denial-of-service attacks. In 2009, Hsu and Chuang [7] indicated that the schemes proposed by Yang et al. and Mangipudi and Katti make user vulnerable to attacks; the attacker may, through publicly available information, determine the identities of legitimate user; Hsu and Chuang thus introduced a novel user identification scheme with key distribution preserving user anonymity for distributed computer networks, which improves not only the shortcomings in the Mangipudi and Katti scheme but also the overall efficiency of the protocol.

This paper suggests that Hsu and Chuang’s scheme is vulnerable to common modulus attacks, and that in the previously mentioned scheme mutual authentication is only achieved after four rounds of communications from both sides; this process is more time consuming and less safe for today’s distributed systems. In order to solve these problems, the authors propose the use of an anonymous user identification and key distribution technical protocol. The proposed scheme requires only three rounds of communications to complete the process of mutual authentication, and thus solves the problem of vulnerability toward common modulus attacks present in Hsu and Chuang’s protocol.

The second section of this paper reviews Hsu and Chuang’s protocol, while the third section examines the loopholes in Hsu and Chuang’s scheme and proposes ways of attack. The fourth section describes the scheme proposed by the authors, the fifth section analyzes the security features of the scheme, and the final section concludes the article.

2. Review of Hsu and Chuang’s scheme

Hsu and Chuang proposed an anonymous user identification and key distribution technical protocol that is suitable for application in distributed computer networks. This system operation can be divided into three phases: 1. the system setup phase, 2. the registration phase, and 3. the user identification phase. The participants in the protocol included user $U_i$, service provider $P_j$, as well as a trusted third-party $SCPC$ (smart card producing center).

2.1 The system setup phase

This paper uses some parameters and notation to describe technical protocols. Brief explanations for them are as follows:

- $p, q$: two random large prime numbers $p$ and $q$ selected by a trusted third party
- $N$: the product of two prime numbers, i.e., $N = p \cdot q$, where $N$ is the common modulus for subsequent calculations
- $g$: an element selected from $Z_n^*$, $g$ is the primitive root of $Z_p^*$ and $Z_q^*$
- $e$: the public key of a trusted third party, such that $e \in \mathbb{Z}_q$
- $d$: the private key of a trusted third party, such that $e \cdot d = 1 \mod \phi(N)$, where $\phi(N) = (p - 1)(q - 1)$
- $K_{ij}$: the shared key common to user and service provider
- $E(\cdot), D(\cdot)$: symmetric encryption and decryption functions
- $h(\cdot), H(\cdot)$: the collision-resistant one-way hash functions
- $||$: the concatenation operation

2.2 The registration phase

First, user (or service provider) must send their identity information $ID_i$ to a trusted third party for registration; afterward, the trusted third party uses a private key $d$ to calculate $S_i = ID_i^d \mod N$ and sends $S_i$ to the user (or the service provider) for future identity validation purposes. The communication processes in the registration phase are all transmitted through secure channels as shown in Fig. 1:

![Fig. 1 The registration phase in Hsu and Chuang’s scheme](image)

2.3 The user identification phase

In this phase, if the user requires access to the services provided by service provider, they must first reach consensus with the service provider in the form of a shared key and mutual authentication. The communication processes are shown in Fig. 2:

![Fig. 2 The user identification phase in Hsu and Chuang’s scheme](image)
Step 1
First, the user must send a service request to the service provider.

Step 2
Once the service provider receives the user service request, it will select a random number $k$ from $Z_N^*$ and use its own private parameter $S_j$ to calculate $Z = g^k \cdot S_j \mod N$ before sending $(Z)$ back to the user.

Step 3
After the user receives the message $(Z)$, $a = Z \cdot ID_j^{-1} \mod N$ is calculated using the public information of the service provider; afterward, a random number $t$ is selected from $Z_N^*$ to generate a shared key $K_j = a^t \mod N$ to be used by both user and service provider, as well as solving for $w = g^{e_t} \mod N$ and $x = S_j^{h(K_j) \mod T} \mod N$. Finally, the shared key $K_j$ is used to encrypt the identity data $ID_i$ to generate $y = E_{K_j}(ID_i) \mod N$; the results $(w, x, y, T)$ are then sent back to the service provider. In the equation, where $T$ is the current timestamp.

Step 4
After the service provider receives the message $(w, x, y, T)$, the timestamp $T$ is verified to ensure that it is within the range of delay; if the answer is negative, communication is terminated. Afterward, the selected random number $k$ is used to obtain the shared key $K_j = w^k \mod N$ and the acquired code is decrypted to obtain the user identity $ID_i = D_{K_j}(y)$, while a validation equation $ID_i^{h(K_j) \mod T} \mod N \equiv x^t \mod N$ is used to validate the derived data $ID_i$. Successful validation indicates that the current user is legitimate; otherwise, communication is terminated. Afterward, the service provider will select a current timestamp $T'$ and calculate $D_i = h(K_j \parallel T' \parallel Z \parallel ID_i \parallel ID_j)$, before sending a message $(D_i, T')$ to the user.

Step 5
After the user receives the message $(D_i, T')$, the timestamp $T'$ is verified to ensure that it is within the range of delay; if the answer is negative, communication is terminated. Afterward, the user calculates $D_i' = h(K_j \parallel T' \parallel Z \parallel ID_i \parallel ID_j)$ and verifies whether the generated results are equal, i.e., $D_i' = D_i$: if they are equal, it indicates that the service provider in this communication session is legitimate; otherwise, communication is terminated.

3. Discussion
This section will address the security loopholes that can be found in the scheme proposed by Hsu and Chuang, as well as suggesting methods that can be used to attack these deficiencies. The following section shows that the attacker can calculate the secret parameters $S_i$ of the user after intercepting public information $(w, x, y, T)$ multiple times and the corresponding session keys. The process is described as follows:

**Common modulus attack**
One of the most common attacks on RSA cryptosystem is common modulus attack [4, 11].
Assume that \((e, n)\) are two RSA public keys. Two ciphertexts, \(c_1 = m^e \mod n\) and \(c_2 = m^f \mod n\), are constructed if a message \(m\) is encrypted to two RSA public keys \(e\) and \(f\) that use the same modulus \(n\). If \(\gcd(e, f) = 1\), then by means of extended Euclidean algorithm, within \(O(\beta^2)\) bit operations two integer coefficients \(a\) and \(b\) are obtained such that \(a \cdot e + b \cdot f = 1\), where \(\beta\) is maximum bit length of \(e\) or \(f\) [5]. With coefficients \(b\) and \(a\), the message \(m\) can be recovered by computing \(m = c_1^a \cdot c_2^b \mod n\) [11].

In the scheme proposed by Hsu and Chuang, if the user and the service provider exchange messages through an open channel and the attacker intercepts each communication message \((w, x, y, T)\) and selects two sets of messages, \(x_i\) and \(x_2\), from the exchange,

\[
x_1 = S_i^{h(K_{ij}^l \mid Z_1 \mid w_1 \mid T)} \mod N,
\]

\[
x_2 = S_i^{h(K_{ij}^l \mid Z_2 \mid w_2 \mid T)} \mod N.
\]

Let \(h_1 = h(K_{ij}^l \mid Z_1 \mid w_1 \mid T)\) and \(h_2 = h(K_{ij}^l \mid Z_2 \mid w_2 \mid T)\). \(h(\cdot) : \{0, 1\}^* \rightarrow \{0, 1\}^t\) reflects a word string of up to \(l\) bits; therefore, the number reflected by \(h(\cdot)\) should not exceed \(2^l\). There are at least \(\frac{2^l}{ln^2}\) prime numbers between \(1 \sim 2^l\). As a result, the probability of \(h_i\) being a prime number is at least \(\frac{2^l}{ln^2} \cdot \frac{1}{ln^2}\), and the probability of \(h_1\) and \(h_2\) being co-primes is at least \(\left(\frac{1}{ln^2}\right)^2\). Using SHA-1 as an example, \(l = 160\); therefore the probability of \((h_1, h_2)\) being co-primes is \(\frac{1}{160^2}\). If \((h_1, h_2)\) are co-primes, then it satisfies the condition \(\gcd(h_1, h_2) = 1\).

Using the extended Euclidean algorithm, the integer coefficients \(a\) and \(b\) are derived such that \(a \cdot h_1 + b \cdot h_2 = 1\), and \(x_1^a \cdot x_2^b = S_i^{a \cdot h_1 + b \cdot h_2} \mod N = S_i \mod N\) is derived from the equations of \(x_1\) and \(x_2\). In this way, if both session keys \(K_{ij1}\) and \(K_{ij2}\) are compromised, the attacker can derive the secret parameter \(S_i\) of legitimate user with probability at least \(1/160^2\). Thus allowing the attacker to impersonate a legitimate user to deceive the service provider and obtain the related services.

In the case of insider attack, namely, the attacker is a malicious service provider; the result may be lethal since attacker knows every session’s session key. Assume that the attacker has collected messages \(x_1, x_2, x_3, h_1, h_2,\) and \(h_3\), where \(\gcd(h_1, h_2, h_3) = 1\). By iterative application of extended Euclidean algorithm, there exists three integer coefficients \(a, b,\) and \(c\) such that \(a \cdot h_1 + b \cdot h_2 + c \cdot h_3 = 1\), and \(x_1^a \cdot x_2^b \cdot x_3^c = S_i^{a \cdot h_1 + b \cdot h_2 + c \cdot h_3} \mod N = S_i \mod N\). Since the probability \(\gcd(h_1, h_2, h_3) = 1\) is greater than that \(\gcd(h_1, h_2) = 1\), therefore the insider attack speeds up computing the secret token of legitimate user.

4. Our protocol

In this article, the authors use the Diffie–Hellman public key technique to ensure the security of the protocol and achieve mutual authentication through the shared key used by both user and service provider. Therefore, it would hinder attackers should they attempt to obtain a legitimate shared key by using the public information available, because they would first have to solve the Diffie–Hellman mathematical problem.

The proposed scheme of the system operation is divided into three phases: 1. the system
setup phase; 2. the registration phase; and 3. the user identification phase. The participants in the protocol include the user $U_i$, the service provider $P_j$, and the trusted third-party SCPC. The system setup parameters and notations are similar to those in Hsu and Chuang’s protocol. The registration phase and user identification phase are described in the following section.

4.1 The registration phase

The procedure of registration is almost the same as the scheme of Hsu and Chuang [7] except the value of $ID_i$ is assigned by SCPC rather than selected by user (or service provider). This will resist the attack of computing secret token as demonstrated in Section 5 and literature [15]. Namely, during the registration procedures, user (or service provider) sends a request of registration to SCPC. After accepting the request, SCPC sends user a secret token $S_i = (ID_i)^d \mod N$ and $ID_i$ with a secure channel to complete the registration.

4.2 The user identification phase

In this phase, if the user wants to obtain access to the services provided by the service provider, both parties must first undergo mutual authentication before jointly generating a shared key, $K_{ij} = g^{e_{k_i,k_j}} \mod N$. The communication process is detailed in this section and illustrated in Fig. 3:

![Fig. 3 The user identification phase](image-url)
Step 1
The user selects a random number $k$ from $\mathbb{Z}_N^*$ and calculates $Z = g^k \mod N$ before sending a service request $(Z, \text{Service Request})$ to the service provider.

Step 2
The service provider selects a random number $k_j$ from $\mathbb{Z}_N^*$ and generates a session key $K_M = Z^{e^{k_j}} = g^{e^{k_j}} \mod N$ before calculating $Z_j = g^{k_j \cdot S_j} \mod N$ using its own private parameter $S_j$. Finally, the verification message $MAC_j = H(K_M, Z, Z_j, ID_j)$ is calculated and a message $(Z_j, MAC_j)$ is sent to the user.

Step 3
Once the user receives the message $(Z_j, MAC_j)$, the service provider’s identification data are used to calculate $g^{e^{k_j}} = Z_j^{e^{k_j}} \mod N$ before using $g^{e^{k_j}}$ to obtain the session key $K_M = g^{e^{k_j} \cdot k} \mod N$ and verifying whether the generated results are equal, i.e. $MAC_j = H(K_M, Z, Z_j, ID_j)$: if they are equal, it indicates that the service provider in this communication session is legitimate; otherwise, communication is terminated. The user selects a random number $k_i$ from $\mathbb{Z}_N^*$ and calculates $Z_i = g^{k_i \cdot S_i} \mod N$ using the private parameter $S_i$; afterward, the session key $K_M$ is used to encrypt the identification data $ID_i$ for obtaining the anonymous identification data $ID_i' = E_{K_M}(ID_i)$. Finally, the service provider data $g^{e^{k_j}}$ and random number $k_i$ are used to calculate the shared key $K_{ij} = g^{e^{k_j} \cdot k_i} \mod N$, common to both user and service provider, and to obtain the verification data $MAC_i = H(K_{ij}, Z, Z_i, ID_i, ID_i', ID_j)$ and send the message $(Z_i, ID_i', MAC_i)$ to the service provider.

Step 4
After the service provider receives the message $(Z_i, ID_i', MAC_i)$, the previously generated session key $K_M$ is used to decrypt the anonymous identity data $ID_i'$ for obtaining the user identity data $ID_i = D_{K_M}(ID_i')$ and calculating the user-related information $g^{e^{k_i}} = Z_i^{e^{k_i}} \mod N$. Afterward, $g^{e^{k_i}}$ and the random number $k_j$ is used to obtain the shared key $K_{ij} = g^{e^{k_i} \cdot k_j} \mod N$, common to both user and service provider. Finally, the generated results are verified for equality, i.e. $MAC_i = H(K_{ij}, Z, Z_i, ID_i, ID_i', ID_j)$: if they are equal, it indicates that the service provider in the given communication session is legitimate; otherwise, communication is terminated.

5. Security analysis
Some attacks on the previous schemes related to anonymous user identification have been suggested on literatures [7-8, 10, 13-15]. Literature [3] lists some desirable security attributes for authenticated key agreement protocols such as known-key security, full forward secrecy (perfect forward secrecy), and key-compromise impersonation. This section analyzes whether the above-mentioned attacks would similarly pose a threat to our proposed scheme, and reviews some of the common forms of attack and security requirements.
5.1 Common modulus attack

If an attacker attempts to perform a common modulus attack to decipher the user private parameter $S_j$ based on the intercepted messages $(Z_j, ID', MAC_j)$ obtained from each communication process, it would prove to be unfeasible. In our protocol, the user chooses a random number $k_j$ from $Z^*_k$ and calculates for $Z_j = g^{k_j} \cdot S_j \mod N$. Hence, it would be difficult for an attacker to obtain the private parameter $S_j$, because each communication employs a different $k_j$ to achieve the purpose of obtaining random $S_j$.

5.2 Replay attack

If an attacker intercepts legitimate messages $(Z_j, ID', MAC_j)$ exchanged between the user and the service provider during their communication sessions and attempts a replay attack on a subsequent communication session, it would prove to be difficult. In each communication session, both the user and the service provider generate different sets of random numbers $k_i$ and $k_j$ and thus produce a different shared session key for every session. Therefore, the attacker will not be able to impersonate any of the user or service provider in any communication session based on previously intercepted messages. Based on this condition, our protocol is capable of resisting replay attacks.

5.3 Forgery attack and Mutual authentication

If an attacker attempts to forge the service provider legitimate message $(Z_j, MAC_j)$ to perform the validation process with the user, it would prove to be unfeasible because $Z_j = g^{k_j} \cdot S_j \mod N$ requires the input of the service provider’s private parameter $S_j$ and this parameter is obtained through a secure channel. Thus, as long as the private parameter $S_j$ is not known, it would be impossible to calculate the corresponding service provider data used for the authentication equation $MAC_j = H(K_u, Z, Z_j, ID_j)$. Therefore the proposed scheme can withstand the attacks described in Wu and Hsu [13]. It would also be impossible for the attacker to attempt to forge the user legitimate message $(Z_i, ID', MAC_i)$ for misleading the service provider, because the attacker would similarly be unable to obtain the user private parameter $S_j$. Therefore, the attacker would similarly be unable to calculate the corresponding user data used for the authentication equation $MAC_i = H(K_u, Z, Z_i, ID_u, ID', ID_j)$. As discussed above, user and service provider authenticate the validity of each other, the proposed scheme achieves mutual authentication.

5.4 User anonymity

Using passwords selected from a collection of small space, an attacker might enumerate all possible passwords. The enumeration can be done offline. Ultimately, the attacker may get the correct password. It is called dictionary attack or password-guessing attack [1-2]. Since the space of identity is small, Hsu and Chuang [7] proposed a dictionary attack on schemes [10, 14] to disclose the identity of user. By searching an identity $ID_i$ to satisfy $w \cdot (ID_i)^{k(w, T)} = x^e \mod N$, a victim user is identified, where $e, N, T, w$, and $x$ are publicly available. To protect user’s identity from revealing, the proposed scheme uses shared key to encrypt the identity of users and session key to verify the authenticity of messages.

That is, identities of legitimate user are not immediately apparent from the publicly
makes models, 1Dk, 1.

This is demonstrated in Literature [7, 8, 10, 13-14]. The proposed scheme restrains user and service provider from freely choosing their IDi. Thus prevents malicious user from launching attack to compute other user’s secret token.

5.6 Known-key security

In this protocol, the establishment of a session key protects the message exchange between the user and the service provider. The shared key $K_{ij} = g^{r_{ik}k_j} \mod N$ is generated by random numbers $(k_i, k_j)$; a different set of random numbers $(k_i, k_j)$ is generated for every communication session. In this way, even if an attacker were able to intercept the session key in one session, it still cannot be used to decipher the contents of a previous communication session. Namely, a compromised session key does not leak information of other session keys.

5.7 Man-in-the-middle attack

It is difficult for attackers to replace information $(Z_j, MAC_j)$. This is primarily because when users receive messages, they must first verify the legitimacy of the authentication code $MAC_j \equiv H(K_M, Z_j, ID_j)$ . Only messages that are sent from legitimate service providers can provide authentication. When service providers receive messages $(Z_i, ID'_i, MAC_i)$ from legitimate users, they must also confirm the legitimacy of the authentication code $MAC_i \equiv H(K_i, Z_i, ID_i, ID_i, ID_i)$. Messages with legitimate authentication codes are can only be sent by legitimate users. In this manner, cross validation mechanisms can prevent man-in-the-middle attacks.
5.8 Perfect forward secrecy

According to the protocol established in this study, the session key \( K_{ij} = g^{ek_i k_j} \mod N \) used on both the server side and user side is composed of random numbers \( k_i \) and \( k_j \), which means that the session key generated for each communication run is different. Learning the quantities of \( Z_i \) and \( Z_j \) cannot benefit attacker to compute the session key \( K_{ij} \), since the hardness of Diffie-Hellman problem. Even if the long-term private keys of both parties, \( S_i \) and \( S_j \) are inadvertently acquired by a third party, the previously generated session keys \( K_{ij} \) would not be compromised.

5.9 Denial-of-service attack (DoS)

Denial-of-Service attack is an attempt to make service provider (server) unavailable to legitimate registered users. Shu and Chuang [7] and Magipudi and Katti [10] indicated that the scheme proposed in [14] suffers from denial-of-service attack since the user end does not verify the authenticity of received messages. During user identification phase (login phase), an attacker simply modifies the messages sending to the user end will cause the server to deny the request of providing service. By the same rationale, the schemes in literatures [8, 13, 15] all have the same weakness. The proposed scheme verifies the authenticity of messages received from server by executing the authentication equation, \( MAC_{ij} = H(K_m, Z_i, Z_j, ID_i) \). This will make the proposed scheme immune from denial-of-service attack.

5.10 Key-compromise impersonation attack

If an adversary learns the secret token \( S_i \) of user \( ID_i \), of course the adversary can impersonate user \( ID_i \) to other entities. A key agreement protocol resists to key-compromise impersonation attack must prevent this adversary from trying to impersonate other entities to \( ID_i \). As explained in subsection 5.3, to achieve mutual authentication the service provider using his secret token \( S_j \) to show the authenticity of message \( Z_j \); the user using his secret token \( S_i \) to show the authenticity of message \( Z_i \). Therefore, the proposed scheme can withstand this attack. Anyone can impersonate service provider in the scheme Lee-Chang [8], since no secret information of the service provider is required during the identification procedure. Thus discussing the key-compromise impersonation attack in this scheme, service providers are excluded from consideration. Namely entities are restricted to users only.

Table 1. Security analysis of the proposed scheme and previous schemes

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Common modulus attack</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Success</td>
<td>Failure</td>
</tr>
<tr>
<td>Replay attack</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
</tr>
<tr>
<td>Mutual authentication</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User anonymity</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Computing user’s secret token</td>
<td>Success</td>
<td>Success</td>
<td>Success</td>
<td>Success</td>
<td>Success</td>
<td>Failure</td>
</tr>
<tr>
<td>Known-key security</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Man-in-the-middle attack</td>
<td>Success</td>
<td>Success</td>
<td>Success</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
</tr>
<tr>
<td>Perfect forward secrecy</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Denial-of-service attack</td>
<td>Success</td>
<td>Success</td>
<td>Success</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
</tr>
<tr>
<td>Key-compromise impersonation attack</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
<td>Failure</td>
</tr>
</tbody>
</table>

* The scheme depicted in Fig. 4 of literature [7] can’t withstand the attack of Denial-of-Service attack. However, Hsu and Chuang [7] remedied this weakness by adding a digital signature on \( Z \), as described in subsection 4.5.
Table 1 summarizes comparison of security analysis of the proposed scheme and previous schemes. In viewing Table 1, an entry Failure/Success implies an adversary will fail/succeed to launch attack; No/Yes indicates the possession of the corresponding attribute. For example, in the row “User anonymity”, only scheme Hsu-Chuang [7] and our scheme preserve attribute of user anonymity as explained in subsection 5.4; in the row “Replay attack”, the adversary will fail to mount replay attack on any scheme; in the row “Computing user’s secret token”, the adversary will succeed to compute user’s secret token in all schemes except our scheme.

6. Discussion

Some previous schemes related to anonymous user identification and key distribution have been mentioned on the study. This section will discuss performance evaluation of these schemes as well as our proposed scheme, according to communication rounds, communication cost, and computational complexity. For easy of comparison, the following notation is defined firstly.

- \( T_{exp} \): the time of executing a modular exponentiation computation;
- \( T_{div} \): the time of executing a modular division computation;
- \( T_{mul} \): the time of executing a modular multiplication computation;
- \( T_{sym} \): the time of executing a symmetric encryption/decryption computation;
- \( T_h \) : the time of executing a one-way hash function \( h(\cdot) \) or \( H(\cdot) \);
- \(|N|\): the bit length of string \( N \).

Table 2 and 3 illustrate the comparison among the schemes in [7-8, 10, 13-14]. The scheme in [15] is absent from Tables, since it only modifies the registration phase of schemes in [13-14] and the modified schemes have the same performance as the original schemes.

Table 2 summarizes computational requirements of each scheme. Since resistant to denial-of-service attack may require more computation, we discuss only those schemes with this attribute, i.e. schemes [7] \(^b\), [10], and our scheme. Also, the discussion is in terms of the modular exponentiation computation \( T_{exp} \), since it is the heaviest burden of computation. In the user end, the proposed scheme executes five times of \( T_{exp} \). Schemes [7] \(^b\) and [10] execute six and seven times of \( T_{exp} \). For the server end, our scheme, schemes [7] \(^b\) and [10] execute four, five, and five times of \( T_{exp} \), respectively. Therefore, the proposed scheme outperforms schemes [7] \(^b\) and [10] in terms of computational complexities.

Table 3 lists a summary on communication rounds and communication cost. As for the communication rounds, all schemes (includes our scheme) require three rounds, except scheme [7]. For convenient comparison on communication cost, consider a practical parameter setting of cryptosystem.

Under Level 4 security concerns [9, 15], the key length is 80 bits for symmetric key encryption and 1248 bits for asymmetric key encryption, the bit length of hashed values is 160. Also to secure against guessing identity attack as described in subsection 5.4, the string \( ID_i \) must have enough bit length. Therefore, we have \(|N| = 1248\), \(|ID| = 160\), \(|MAC| = 160\), and \(|T| = 48\). As listed in Table 3, among the schemes [7] \(^b\), [10], and our scheme, secure against denial-of-service attack, our scheme requires least bit size. It saves at least about 20% of the
communication bits. Also, only the scheme proposed in the study requires no system wide timestamp. This will decrease the costs of construction and maintenance, and make it more applicable in distributed network environment.

Table 2. Computational complexities of the proposed scheme and previous schemes

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Computational complexities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>User</td>
</tr>
<tr>
<td>Lee-Chang [8]</td>
<td>$5T_{\text{exp}}+ 5T_{\text{mul}}+ T_h$</td>
</tr>
<tr>
<td>Wu-Hsu [13]</td>
<td>$4T_{\text{exp}} + T_{\text{div}}+ 2T_{\text{mul}}+ T_h$</td>
</tr>
<tr>
<td>Yang et al. [14]</td>
<td>$5T_{\text{exp}}+ 3T_{\text{mul}}+ T_h+ T_{\text{sym}}$</td>
</tr>
<tr>
<td>Mangipudi-Katti [10]</td>
<td>$7T_{\text{exp}}+ 3T_{\text{mul}}+ 2T_h+ T_{\text{sym}}$</td>
</tr>
<tr>
<td>Hsu and Chuang [7]</td>
<td>$4T_{\text{exp}}+ T_{\text{div}}+ T_{\text{mul}}+ T_h+ T_{\text{sym}}$</td>
</tr>
<tr>
<td>Hsu and Chang [7]$^b$</td>
<td>$6T_{\text{exp}}+ T_{\text{div}}+ T_{\text{mul}}+ 2T_h+ T_{\text{sym}}$</td>
</tr>
<tr>
<td>Our scheme</td>
<td>$5T_{\text{exp}}+ T_{\text{div}}+ T_{\text{mul}}+ 2T_h+ T_{\text{sym}}$</td>
</tr>
</tbody>
</table>

$^a$ As described in Fig. 4 of literature [7], it is incapable of resisting to Denial-of-Service attack.
$^b$ Hsu and Chang’s improvement on scheme $^a$, it can withstand the attack of Denial-of-Service.

Table 3. Communicational cost of the proposed scheme and previous schemes

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Communication</th>
<th>Timestamp required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rounds</td>
<td>Cost</td>
</tr>
<tr>
<td>Lee-Chang [8]</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Wu-Hsu [13]</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Yang et al. [14]</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Mangipudi-Katti [10]</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Hsu and Chuang [7]$^a$</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Hsu and Chuang [7]$^b$</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Our scheme</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

$^a$ and $^b$ are the same as that in Table 2.

7. Conclusion

In this article, we found that the scheme proposed by Hsu and Chuang is vulnerable to common modulus attacks; an attacker can impersonate a legitimate user to gain access to the services provided by service provider. In order to solve the problems present in Hsu and Chuang’s scheme, the authors proposed an anonymous user identification and key distribution technical protocol that is resistant to common modulus attacks.

Additionally, in the proposed scheme, mutual authentication can be achieved in only three rounds of communication sessions, which is one round less than that in Hsu and Chuang's scheme. In a distributed network, the focus is mainly on the number of communications; hence, if the number of communications is reduced, the time spent waiting for a response can be shortened. This reduction shortens the amount of time that user and service provider spend on authentication. Meanwhile, costs can be reduced through improvements in the hardware. In this manner, with the application of this protocol in a distributed network, our program is comparatively more efficient. Additionally, the authentication protocol eliminates the use of timestamps to avoid the inconvenience of using synchronized clocks across distributed systems. This amendment makes the proposed scheme more suitable for distributed computer network.
environments.

References


