Efficient remote mutual authentication and key agreement

Wen-Gong Shieh*, Jian-Min Wang

Department of Information Management, Chinese Culture University, 55, Hwa-Kang Road, Yang-Ming-Shan, Taipei 11114, Taiwan, ROC

Abstract

A smart card based scheme is very practical to authenticate remote users. In 2004, Juang [Juang WS. Efficient password authenticated key agreement using smart cards. Computers and Security 2004;23:167–73] proposed a mutual authentication scheme using smart cards. The advantages in the scheme include freely chosen passwords, no verification tables, low communication and computation cost, and session key agreement. In addition, synchronized clocks are not required in the scheme due to its nonce based approach. In this paper, however, we shall discuss the weakness of Juang’s [Juang WS. Efficient password authenticated key agreement using smart cards. Computers and Security 2004;23:167–73] scheme and propose another similar scheme to improve the weakness. Our scheme not only preserves all the advantages of Juang’s scheme but also improves its efficiency.

1. Introduction

Remote mutual authentication is a mechanism for two communicating parties to mutually authenticate each other through an insecure communication channel. Since Lamport (1981) proposed his remote authentication scheme in 1981, many scholars (Chang and Wu, 1993; Chien et al., 2002; Hsu, 2004; Hwang and Li, 2000; Hwang et al., 2002; Jan and Chen, 1998; Sun, 2000; Tan and Zhu, 1999; Wang and Chang, 1996; Wu, 1995; Wu and Sung, 1996; Yang and Shieh, 1999) have proposed new schemes and improved the efficiency and security of remote authentication. In 2000, Sun (2000) proposed a cost effective unilateral remote authentication scheme in which only a server can authenticate a user’s legitimacy. In 2002, Chien et al. (2002) proposed an efficient remote mutual authentication scheme using smart card allowing server and user to authenticate each other. The advantages in the scheme include freely chosen passwords, no verification tables, low communication and computation cost. However, as demonstrated by Hsu (2004), Chien et al.’s scheme is vulnerable to the parallel session attack. Recently, in 2004, Juang (2004) proposed another scheme preserving all the advantages of Chien et al.’s scheme. Unlike Chien et al.’s scheme, Juang’s scheme is nonce based. Therefore, no synchronized clocks are required in the scheme. In addition, Juang’s scheme generates a session key for the user and server in their subsequent communication. In this paper, however, we shall discuss the weakness of Juang’s (2004) scheme and propose another similar scheme to improve the weakness. Our scheme not only preserves all the advantages of Juang’s scheme but also improves its efficiency.
2. Review of related papers


2.1. Chien et al.’s scheme

The security of Chien et al.’s (2002) scheme is based on the secure one-way hash function (NIST FIPS PUB 180, 1993; Rivest, 1992). This scheme consists of three phases: the registration phase, the login phase, and the verification phase. Fig. 1 is an illustration of messages transmitted in Chien et al.’s scheme during the verification phase.

2.1.1. Registration phase

Assume a user $U_i$ submits his identity $ID_i$ and password $PW_i$ to a server for registration via secure channel. If the server accepts the request, it computes $R_i = h(ID_i \oplus x) \oplus PW_i$ and issues $U_i$ a smart card containing $R_i$ and $h()$, where $h()$ is a one-way hash function, $x$ is the secret key maintained by the server, and the symbol “$\oplus$” denotes the exclusive-or operation.

2.1.2. Login phase

When $U_i$ wants to login to the server, he inserts his smart card into a terminal and inputs his identity $ID_i$ and password $PW_i$. Then, the smart card performs the following steps:

1. Compute $C_1 = R_i \oplus PW_i$.
2. Compute $C_2 = h(C_1 \oplus T)$, where $T$ is the current time stamp.
3. Send the message $(ID_i, T, C_2)$.

2.1.3. Verification phase

After receiving the message $(ID_i, T, C_2)$ from $U_i$, the server performs the following steps to authenticate $U_i$:

1. Checks whether $ID_i$ is valid, and whether the time $T$ is fresh by comparing $T$ and the current time $T'$. If either one fails, reject $U_i$’s login request.
2. Compute $C'_2 = h(ID_i \oplus x)$ and check whether $h(C'_2 \oplus T)$ is equivalent to the received $C_2$. If either one fails, reject $U_i$’s login request. Otherwise, accept $U_i$’s request.
3. Compute $C_3 = h(C'_2 \oplus T')$, where $T'$ is the current time.
   Then, send the message $(C_3, T')$ back to $U_i$.

After receiving the message $(C_3, T')$ from the server, $U_i$ performs the following steps to authenticate the server:

1. Check whether the time $T'$ is fresh. If it is not, $U_i$ gives up the login request.
2. Check whether $h(C'_2 \oplus T')$ is equivalent to the received $C_3$. If yes, $U_i$ believes that the responding party is the real server, and the mutual authentication is done. Otherwise, $U_i$ gives up the login request.

2.2. Hsu’s parallel session attack

In Hsu’s (2004) paper, he demonstrates a parallel session attack to Chien et al.’s scheme. Fig. 2 is an illustration of messages transmitted in the parallel session attack. An intruder can send immediately the message $(ID_i, T', C_3)$ to login to the server when observing the legal message $(T', C_3)$ back from the server. In such a case, the time $T'$ is still fresh to the server and $C_3 = h(C'_2 \oplus T')$ is given by the server. In other words, the intruder can pass the checking of verification phase and login to the server successfully. Hsu (2004) further pointed out that the attack is workable due to the symmetric structure of messages exchanged between $U_i$ and the server.

2.3. Juang’s scheme

The security of Juang’s (2004) scheme is based on the symmetric encryption and adopts nonce to avoid the time-synchronization problem of using time stamps. His scheme consists of two phases: the registration phase, and the login and session key agreement phase.

2.3.1. Registration phase

This phase is almost the same as that of Chien et al.’s scheme. A user $U_i$ submits his identity $ID_i$ and password $PW_i$ to a server for registration. The server computes $V_i = h(ID_i, x)$, $W_i = V_i \oplus PW_i$, and issues $U_i$ a smart card containing $W_i$, $ID_i$ and $h()$, where $h()$ is a one-way hash function, $x$ is the secret key maintained by the server, and the symbol “$\oplus$” denotes the exclusive-or operation. Note that $ID_i$ is not stored in smart card in Chien et al.’s scheme.

2.3.2. Login and session key agreement phase

When $U_i$ wants to login to the server, he inserts his smart card into a card reader and inputs his identity $ID_i$ and password $PW_i$. Fig. 3 is an illustration of messages transmitted in Juang’s scheme during the jth login and session key agreement phase. The protocol is presented below.

1. The smart card computes secret information $V_i = W_i \oplus PW_i$ then sends the message $(N, ID_i, E_{V_i}(ru, C_j))$ to the server, where $C_j = h(ID_i || N_i)$, $E_{V_i}( )$ denotes a symmetric encryption algorithm using $V_i$ as the secret key, and

---

**Fig. 1**– Messages transmitted in Chien et al.’s scheme.

**Fig. 2**– Messages transmitted in parallel session attack (Hsu, 2004).
the symbol “||” denotes the string concatenation operation. $N_i$ is a nonce and $r_u$ is a random value chosen by the smart card to generate the session key $K_i$.

2. After receiving the message, the server computes $V_i = h(ID_i, x)$ and $(r_u, C_i) = D_{K_i}(E_{K_i}(r_u, C_i))$, where $D_{K_i}(\cdot)$ denotes the corresponding symmetric decryption algorithm of $E_{K_i}(\cdot)$ using $V_i$ as the secret key. After decryption, if $C_i$ is not equal to $h(ID_i || N_i)$, or $N_i$ is not fresh, the server rejects $U_i$’s request. Otherwise, the server sends the message $(E_{K_i}(r_s, N_i + 1, N_2))$ to $U_i$, where $N_2$ is a nonce and $r_s$ is a random value chosen by the server to generate the session key $K_s$.

3. When the second message is received, the smart card decrypts and checks whether $N_i + 1$ is in it. If yes, the smart card computes the session key $K_i = h(r_s, r_u, V_i)$ and sends the message $(E_{K_i}(N_2 + 1))$ back to the server. On receiving the last message, the server computes $D_{K_i}(E_{K_i}(N_2 + 1))$ to check whether $N_2 + 1$ is in it. If $N_2 + 1$ is found, the server and $U_i$ have achieved mutual authentication and session key agreement.

3. Our scheme

In this section, we propose a remote mutual authentication and key agreement scheme using smart card with secure one-way hash function (Merkle, 1989; NIST FIPS PUB 180, 1993; Rivest, 1992). In our scheme, there is no time-synchronization requirement. Our scheme consists of two phases: the registration phase, and the login and key agreement phase. The symbols in our scheme are defined as following:

\[ h(\cdot): \text{secure one-way hash function} \]
\[ x: \text{the secret key maintained by the server} \]
\[ \oplus: \text{exclusive-or operation} \]
\[ ||: \text{string concatenation operation} \]

3.1. Registration phase

Our registration phase is exactly the same as that of Chien et al.’s scheme. Assume a user $U_i$ submits his identity $ID_i$ and password $PW_i$ to the server over a secure channel for registration. If the request is accepted, the server computes $R_i = h(ID_i \oplus x) \oplus PW_i$ and issues $U_i$ a smart card containing $R_i$ and $h(\cdot)$.

3.2. Login and key agreement phase

Fig. 4 is an illustration of messages transmitted during the login and key agreement phase in our scheme.

When the user $U_i$ wants to login to the server, he first inserts his smart card into a card reader then inputs his identity ID$_i$ and password PW$_i$. The smart card then performs the following steps to begin an access session:

1. Compute $a_i = R_i \oplus PW_i$.
2. Acquire current time stamp $T_u$, store $T_u$ temporarily until the end of the session, and compute $MAC_u = h(T_u \parallel a_i)$.
3. Send the message $(ID_i, T_u, MAC_u)$ to the server and wait for a response from the server. If no response is received in time or the response is incorrect, report login failure to the user and stop the session.

After receiving the message $(ID_i, T_u, MAC_u)$ from $U_i$, the server performs the following steps to assure the integrity of the message, respond to $U_i$, and challenge $U_i$ to avoid replay:

1. Check if the received $T_u$ has already appeared in a current executing session of user $U_i$, reject $U_i$’s login request and stop the session. Otherwise, $T_u$ is fresh.
2. Compute $a'_i = h(ID_i \oplus x)$, $MAC'_u = h(T_u \parallel a'_i)$, and check whether $MAC'_u$ is equal to the received $MAC_u$. If it is not, reject $U_i$’s login and stop the session.
3. Acquire the current time stamp $T_u$. Store temporarily paired time stamps $(T_u, T_s)$ and $ID_i$ for freshness checking until the end of the session. Compute $MAC_u = h(T_u \parallel T_s \parallel a'_i)$ and session key $K_s = h(T_u \parallel T_s \parallel a'_i)$. Then, send the message $(T_u, T_s, MAC_u)$ back to $U_i$ and wait for a response from $U_i$. If no response is received in time or the response is incorrect, reject $U_i$’s login and stop the session.

Note that, in the above steps, in message $(T_u, T_s, MAC_u)$, $T_u$ is a response to $U_i$ while $T_s$ is a challenge to $U_i$. In addition, a further replaying of the message $(ID_i, T_u, MAC_u)$ to the server before the end of this session will be discovered in the above step 1 because the paired time stamps $(T_u, T_s)$ and $ID_i$ have been stored in the above step 3.

On receiving the message $(T_u, T_s, MAC_u)$ from the server, the smart card performs the following steps to authenticate the server, achieve session key agreement, and respond to the server:

1. Check if the received $T_u$ is equal to the stored $T_u$ to assure the freshness of the received message. If it is not, report login failure to the user and stop the session.
2. Compute $MAC'_u = h(T_u \parallel T_s \parallel a_i)$ and check whether it is equal to the received $MAC_u$. If not, report login failure to...
Fig. 5 – Messages transmitted in our new scheme using synchronized clock.

- The user and stop. Otherwise, conclude that the responding party is the real server.
- Compute \(\text{MAC}_u'' = h(T_u \| (a + 1))\) and session key \(K_a = h((T_u \| T_s) \oplus a)\), then send the message \((T_u, \text{MAC}_u'')\) back to the server. Note that, in the message \((T_u, \text{MAC}_u'')\), \(T_s\) is a response to the server.

When the message \((T_u, \text{MAC}_u'')\) from \(U_i\) is received, the server performs the following steps to authenticate \(U_i\) and achieve key agreement:

1. Check if the received \(T_s\) is equal to the stored \(T_s\). If it fails, reject \(U_i\)’s login request and stop the session.
2. Compute \(\text{MAC}_s'' = h(T_s \| (a' + 1))\) and check whether it is equal to \(\text{MAC}_u''\). If it is not, reject \(U_i\)’s login request and stop the session. Otherwise, conclude that \(U_i\) is a legal user and permit the user \(U_i\)’s login. At this moment, mutual authentication and session key agreement between \(U_i\) and the server are achieved. From now on, the user \(U_i\) and the server can use the session key \(K_a\) in their further secure communication until the end of the access session.

If synchronized clocks are used in our scheme, the checking of time stamp, \(T_u\) or \(T_s\), in a message alone by the receiving party can assure the freshness of the message. Therefore, using synchronized clocks, we do not have to create challenge and wait for response to withstand replay attacks. In such case, we can create a new scheme where only two messages are required to achieve the mutual authentication and session key agreement. Fig. 5 illustrates the messages transmitted in such new scheme. Instead of challenges or responses, \(T_u\) and \(T_s\) in the new scheme become playing just the role of true time stamps. Therefore, the third message in Fig. 4 is deleted. In addition, \(T_u\) in the second message is not required and is deleted. Besides, there is no need to store \(T_u\) or \(T_s\) during an access session. However, the message authentication codes \(\text{MAC}_u\) and \(\text{MAC}_s\) in Fig. 5 remain the same as that of Fig. 4. The session key computations in the new scheme also remain unchanged.

4. Security analysis

In this section, we will point out the weakness of Juang’s (2004) scheme, in section “Weakness of Juang’s scheme”, and analyze the security of our scheme in section “Security analysis of our scheme”.

4.1. Weakness of Juang’s scheme

In Juang’s (2004) scheme, first, it does not check the integrity of transmitted messages. Instead of time stamps, Juang adopts challenge and response approach using nonce \(N_1\) and \(N_2\) to protect his scheme against replay attacks. Unfortunately, to save computation cost, messages are not signed in his scheme. It implies that an attacker may modify the transmitted messages, modifying \(r_u\), for example, without being discovered as long as the challenge nonce and the response nonce remain correlated. In addition, in the first message in Fig. 3, the second block of the ciphertext \(E_{K_1}(r_u, C_1)\) and its corresponding plain-text \(C_1 = h(\text{ID} || N_1)\) are known to an intruder. Therefore, the shared secret key \(V_j\) may face off-line known plain-text attack. Besides, in Juang’s scheme, a late enough replaying of the first message in Fig. 3 will not be determined immediately as a replay attack by the server. For example, replay the first message after one day. The server must send a challenge nonce in the second message to the user and wait for a response in the third message to detect if the first message is a replayed one. Attackers can launch a masquerade attack using the replayed first message to login if the third message is not a requirement to permit a login request in Juang’s scheme. In other words, Juang claimed that the third message in his scheme can be delayed and is not correct. That is, the server in his scheme should not permit a user’s login until the correct third message from the user is received.

4.2. Security analysis of our scheme

In this section, we analyze the security of our scheme as following:

1. The server’s secret key \(x\) is protected by the secure one-way hash function \(h()\). It is computationally infeasible to derive \(x\) from the value \(h(\text{ID} \oplus X)\). In the same way, the shared secret \(a\) between \(U_i\) and the server cannot be derived from the message authentication code \(\text{MAC}_u\), \(\text{MAC}_s\), or \(\text{MAC}_{u''}\). Therefore, \(a\) is safely shared only between \(U_i\) and the server.
2. The server and the smart card will not generate repeat nonce, or the same message authentication codes, because we use time stamps as nonce in our challenge and response approach. Time stamps will never repeat due to the different combination of year, month, day, hour, minute, and second in each time stamp.
3. Mutual authentication between \(U_i\) and the server is achieved, because \(U_i\) and the server authenticate each other with the message authentication codes \(\text{MAC}_u\) and \(\text{MAC}_{u''}\), respectively. Since nobody can create the correct message authentication codes without knowing the shared secret value \(a\) between \(U_i\) and the server, \(a\) is used to confirm the legitimacy of each party. In other words, it is infeasible for an intruder or a pretended server to masquerade as a legal party.
4. The replay attacks fail due to our challenge and response scheme. Based on the scheme, a replay attack cannot pass the subsequent challenges. When the server receives the message \((\text{ID}_i, T_u, \text{MAC}_u)\), it includes a challenge nonce \(T_u\) from \(U_i\). Therefore, the server must send back the received \(T_u\) to \(U_i\) as the response nonce. When \(U_i\) receives the message \((T_u, T_s, \text{MAC}_s)\), it includes not only the response nonce \(T_s\) but also the challenge nonce \(T_u\) from the server.
Therefore, \( U_i \) must send back the message \((T_s, \text{MAC}_u)\) including the response nonce \( T_s \) to the server. Note that \( T_u \) and \( T_s \) as challenge nonce are fresh. Besides, \( \text{MAC}_u \) and \( \text{MAC}_s \) guarantee their integrity and source, respectively. In addition, it is impossible to create corresponding responses and their message authentication codes, \( \text{MAC}_u \) and \( \text{MAC}_s \), without knowing the shared secret value \( a_i \) between \( U_i \) and the server. Therefore, except for \( U_i \) and the server, no one can pass the challenges.

5. The parallel session attacks fail because of the asymmetric structure of the message authentication codes \( \text{MAC}_u \) and \( \text{MAC}_s \). Note that \( \text{MAC}_u = h(T_u \| a_i) \) and \( \text{MAC}_s = h(T_s \| (a_i + 1)) \).

### 4.3. Note of our scheme

In our scheme, it is essential to check response nonce \( T_u \) and \( T_s \) in the verification phase as mentioned in Stallings (2003). If the corresponding challenging party does not check \( T_u \) and \( T_s \) as responses, an intruder can use the parallel session attack to login to the server successfully. Fig. 6 is an illustration of messages transmitted in the parallel session attack. In Fig. 6, if \( U_i \) does not check response \( T_u \) in message 3, \( U_i \) will not discover that \( T_u \) has been changed to \( T_{u1} \) in the message \((T_{u1}, T_{s1}, \text{MAC}_{s1})\). \( U_i \) will then be used to create response message \((T_{s1}, \text{MAC}_{u1})\) for the intruder.

### 5. Efficiency

In this section, we evaluate some performance issues of our scheme, and compare the result with Juang’s (2004) scheme. At the end of this section, the comparisons of the related schemes are given in Table 1.

Both our scheme and Juang’s (2004) scheme are based on secure one-way hash function (Merkle, 1989; NIST FIPS PUB 180, 1993; Rivest, 1992). In Juang’s (2004) scheme, besides the secure one-way function, symmetric encryption NIST FIPS PUB 197 (2001) is also used. In the registration phase, the computation of our scheme and Juang’s (2004) scheme, both perform one hashing operation. In verification phase, our scheme performs seven hashing operations to achieve mutual authentication. In Juang’s (2004) scheme, it requires three symmetric encryptions, three symmetric decryptions and three hashing operations. Since the cost of key generation is almost the same and the computation cost of hashing operation is much less than that in symmetric encryption, we can see that our scheme is more efficient.

Besides the efficiency of the smart card, there are also some other significant issues for user authentication mentioned in Juang’s (2004) paper, including (1) no verification table; (2) freely chosen password; (3) mutual authentication; (4) no synchronization requirement; and (5) session key exchange. We add our scheme into the comparison table given in Juang’s paper as shown in Table 1.

![Table 1 – Comparisons of the smart card based schemes](image-url)
6. Conclusions

In this paper, we point out the weakness of Juang’s scheme and further propose a novel remote user authentication scheme using smart card. Compared with Chien et al.’s (2002) scheme, our scheme has following merits: (1) there is no parallel session attack problem; (2) does not need the synchronized clock; (3) generates a session key. Compared with Juang’s (2004) scheme, our scheme can achieve the same purpose but have less computational cost and is safer.

REFERENCES


Wen-Gong Shieh received his B.E. degree in industrial design from National Cheng-Kung University in 1981, and both his M.E. degree and Ph.D. degree in computer science and engineering from the University of Texas at Arlington in 1988 and 1992 respectively. He joined the Department of Information Management, Chinese Culture University, in 1992. Currently, he is an associate professor and the chairman of the Department of Information Management. His research interests include electronic commerce, information security, computer virus, and software development.

Jian-Min Wang received his B.E. degree in computer science from Chinese Culture University in 2003, and his M.B.A. degree in information management from Chinese Culture University in 2005. Currently, he is performing his military service. His research interests include information security, electronic commerce, and Web management.