Efficient User Authentication and Key Agreement With Privacy Protection

Wen-Shenq Juang
Department of Information Management
Shih Hsin University
Taipei, Taiwan, 116, R.O.C.
Email:wsjuang@cc.shu.edu.tw

Jing-Lin Wu
Department of Information Management
Shih Hsin University
Taipei, Taiwan, 116, R.O.C.
Email:m93662003@cc.shu.edu.tw

Abstract

Using smart cards, remote user authentication and key agreement can be simplified, flexible, and efficient for creating a secure distributed computers environment. Addition to user authentication and key distribution, it is very useful for providing identity privacy for users. In this paper, we propose novel user authentication and key agreement schemes with privacy protection. We first propose a single-server scheme and then apply this scheme to a multi-server environment. The main merits include: (1) the privacy of users can be ensured; (2) a user can freely choose his own password; (3) the computation and communication cost is very low; (4) servers and users can authenticate each other; (5) it generates a session key agreed by the server and the user; (6) our proposed schemes are nonce-based schemes which does not have a serious time-synchronization problem.

Keywords: *User Authentication, Session Key, Privacy Protection, Smart Card, Network Security.*

1 Introduction

For obtaining permitted services by service providers in a network environment, the user must legally login to the provider's server. In general, the user transmits a message of user authentication to the server, and then the server must be able to verify the identity of the user and give him the right of using permitted services. Typically, the user passes a password as a secret token to the server. The server first checks if the user's identity and the password are matching. The server rejects the user's request if his identity or the password is not matching. If the password is matching, the server give the user the right for using the permitted services.

In 1981, Lamport [8] first proposed a password authentication scheme at the both ends of the communication. Since then, many schemes have been proposed to point out its drawback and improve the security and efficiency of Lamport's scheme [8]. Only passing a password for authenticating between the user and the server is not enough, since it is less safety and is easily tapped by the adversary. Before two parties can do secure communication, a session key is needed for protecting subsequence communications [2, 6, 7, 19]. Also, using smart cards [6, 7, 19], remote user authentication and key agreement can be simplified, flexible and efficient for creating a secure distributed computers environment. It is also useful for providing identity privacy for the users [19]. In 2004, Juang proposed two efficient authentication and key agreement schemes [6, 7] for single server, and multi-server environments. But both Juang's schemes [6, 7] have no ability of anonymity for the user. Yang et. al. [19] proposed user identification and key distribution scheme with the ability of privacy protection but we point out it is less efficient because of using public-key cryptosystems.

For basically security and efficient requirements, the following criteria are important for remote user authentication and key agreement schemes with smart cards [6, 7, 19].

C1: Privacy protection: When the user authenticates successfully to the server, the adversary can not derive the user's identity.

C2: Freely chosen password: Users can freely chosen and change their passwords for protecting their smart cards.

C3: Low computation and communication cost: Since capacity and communication constrains of smart cards, they may not offer a powerful computation capability and high bandwidth.

C4: Mutual authentication: Servers and users can authenticate each other.

C5: Session key agreement: Servers and users must negotiate a session key for subsequent communications.

In this paper, we propose two efficient user authentication and key agreement schemes with the ability of privacy protection. One is only for a single server environment and the other is suitable for a multi-server environment. Compared with our proposed multi-server scheme and Yang et al.'s scheme [19], our scheme is more efficient since our scheme only uses the symmetric cryptosystems and hashing functions. Our proposed schemes satisfy all above five criteria. In addition, Yang et. al.'s scheme [19] has a serious time-synchronization problem, since their scheme is timestamp-based. Our proposed schemes have not this problem at all since our schemes are based on nonces.

The remainder of this paper is organized as follows: In Section 2, a brief review of related user authentication and key agreement schemes is given. In Section 3, we present our single server scheme with privacy protection. In Section 4, a multi-server scheme with privacy protection is given. In Section 5, we make a discussion. Finally, a concluding remark is given in Section 6.

2 Review

2.1 Notation

We first define the notation used in this paper. Let "$X \rightarrow Y : Z$" denote that a sender X sends a message Z to a receiver Y, $E_k(m)$ denote the ciphertext of m encrypted using the secret key k of some secure symmetric cryptosystem [13], $D_k(c)$ denote the plaintext of c decrypted using the secret key k of the corresponding symmetric cryptosystem [13], "$||$" denote the conventional string concatenation operator and $\oplus$ denote the bitwise exclusive-or operator. Let h be a public one-way function [14].

2.2 Juang's single server authentication scheme

In [6], Juang proposed a user authentication and key agreement scheme using smart cards with much less computational cost and more functionality. The major drawbacks of this scheme are that it does not provide the user anonymity functionality and it is not suitable for multi-server environments. Let S denote the server, U_i denote user i. Also, let x be the secret key kept secretly by the server S. Let ID_i be a unique identification of U_i.

The scheme is as following.

Registration Phase: Assume U_i submits his identity ID_i and his password PW_i to the server for registration. If the server accepts this request, he will perform the following steps:

Step 1: Compute $U_i's$ secret information $v_i = h(ID_i, x)$ and $w_i = v_i \oplus PW_i$.

Step 2: Store ID_i and w_i to the memory of a smart card and issue this smart card to U_i.

Login and Session Key Agreement Phase:

After getting the smart card from the server, U_i can use it when he logins in the server. If U_i wants to login to S, he must attach his smart card to a card reader. He then inputs his identity ID_i and his password PW_i to this device. Assume that N_1 is a nonce chosen by U_i and N_2 is a nonce chosen by S_j for freshness checking. Assume that ru_k is a random number chosen by U_i and rs_k is a random number chosen by S_j for generating the session key $k_i = h(rs_k, ru_k, v_i)$. The following protocol is the ith login with respect to this smart card.

Step 1: $U_i \rightarrow S : N_1, ID_i, E_{v_i}(ru_i, h(ID_i|| N_1))$

Step 2: $S \rightarrow U_i : E_{v_i}(rs_{i,}N_1 + 1, N_2)$

Step 3: $U_i \rightarrow S : E_{k_i}(N_2 + 1)$

2.3 Juang's multi-server authentication scheme

In [7], Juang proposed a user authentication and key agreement scheme using smart cards for multi-server environments with much less computational cost and more functionality. The major drawback of this scheme is that it does not provide the user anonymity functionality. There are three kinds of participants in this scheme: users, servers and a registration centre. In this scheme, assume that the registration centre can be trusted. The registration centre examines the validity of login users and then issues a smart card to eligible users. The user only has to register at the registration center once and can use services provided by various servers. Let RC denote the registration centre, S_j denote server j, and U_i denote user i. Let UID_i be a unique identification of U_i and SID_j be a unique identification of S_j. Also, let x be the secret key kept secretly by RC, and $w_j = h(x, SID_j)$ be the secret key shared by S_j and RC. The shared secret key w_j can be computed by RC and sent to S_j after he registered at RC.

The proposed scheme is as follows.

Registration Phase: U_i submits his identity UID_i and his password PW_i to RC for registration. RC then performs the following steps:

Step 1: Compute U_i's secret information $v_i = h(x, UID_i)$ and $\mu_i = v_i \oplus PW_i$.

Step 2: Store UID_i and μ_i to the memory of a smart card and issue this smart card to U_i.

Step 3: Compute the shared secret key $v_{i,j} = h(v_i, SID_j)$ between U_i and S_j, and send the encrypted secret key $E_{w_j}(v_{i,j}, UID_i)$ to each S_j. Upon receiving $E_{w_j}(v_{i,j}, UID_i)$, S_j stored it in his encrypted keys table.

Login and Session Key Agreement Phase: After getting the smart card from RC, U_i can use it to login into S_j. Assume that N_1 is a nonce chosen by U_i and N_2 is a nonce chosen by S_j for freshness checking. Assume that ru_k is a random number chosen by U_i and rs_k is a random number chosen by S_j for generating the session key

$sk_k = h(rs_k, ru_k, v_{i,j})$. The following protocol is the *kth* login with respect to his smart card.
Step 1: $U_i \rightarrow S_j : N_1, UID_i, E_{v_{i,j}}(ru_k, h(UID_i||N_1))$
Step 2: $S_j \rightarrow U_i : E_{v_{i,j}}(rs_k, N_1 + 1, N_2)$
Step 3: $U_i \rightarrow S_j : E_{sk_k}(N_2 + 1)$

Shared Key Inquiry Phase: In Step 3 of the registration phase, RC will send the encrypted shared secret key $E_{w_j}(v_{i,j}, UID_i)$ to each S_j. Upon receiving the message, he will store it in his encrypted shared key table. If he do not want to manipulate this table, the shared key can be inquired from RC when it is needed. The following protocol can be inserted between Step 1 and Step 2 of the login and session key agreement phase when S_j needs the shared key.
Step 1': $S_j \rightarrow RC : N_3, UID_i, SID_j, E_{w_j}(h(UID_i||SID_j||N_3))$
Step 1": $E_{w_j}(v_{i,j}, N_3 + 1)$

2.4 Yang *et al.*'s user authentication and key distribution scheme

Yang *et al.* proposed a user authentication and key distribution with user anonymity [19] based on factoring, discrete logarithm and hash functions. The major drawbacks of this scheme are that it has a time-synchronization problem, and the computation and communication cost is still high. There are three kinds of participants in this scheme: a Smart Card Producing Center (SCPC), service providers (servers) and users. Let U_i denote user i, P_j denote service provider j. This scheme consists of two phases: (1) the key generation phase and (2) the anonymous user identification phase. Their proposed scheme is as follows:
The key generation phase:
The SCPC does the following to set up system parameters.

1. Chooses two large primes p and q, computes $n = pq$, randomly selects a number e and computes d, where $ed \equiv 1 \bmod \phi(n)$ and $\phi(n) = (p-1)(q-1)$.
2. Chooses an element $g \in Z_n^*$ which is a generator of both Z_p^* and Z_q^*.
3. Publishes (e, n, g) as public system parameters and keeps (d, p, q) secret.
4. Sends to each registered user U_i or service provider P_i a secret token $S_i \equiv (ID_i)^d \bmod n$, where ID_i is the identity of U_i or P_i.

The anonymous user identification phase:
If U_i wants to request a service from P_j, they then performs the following steps:
Step 1: U$_i$ Sends the service request to P_j for requesting services from P_j.
Step 2: Upon receiving the request, P_j chooses a random number k and computes $z \equiv g^k S_j^{-1} \bmod n$ and sends z to U_i.
Step 3: Upon receiving z, U_i chooses a random number t and does the following computations:

$$a = z^e ID_j \bmod n,$$
$$K_{ij} = a^t \bmod n,$$
$$x = g^{et} \bmod n,$$
$$s = g^t S_i^{h(x,T)} \bmod n,$$
$$y = E_{K_{ij}}(ID_i),$$

where T is the current timestamp and $K_{i,j}$ is the common session key. U_i then sends (x, s, y, T) to P_j.

Step 4: Upon receiving the message in Step 3, P_j checks the timestamp T. If it is old, he aborts the protocol. Otherwise, he then obtains the common session key $K_{ij} = x^k \bmod n$ and then decrypts y as $ID_i = D_{K_{ij}}(y)$ and verifies

$$xID_i^{h(x,T)} \overset{?}{=} s^e \bmod n.$$

If the verification passes, then the service request is granted.

3 Single server authentication and key agreement with user anonymity

In this section, we propose an efficient single server user authentication and key agreement scheme with privacy protection. The concept used in this section will be used in the next section to construct an efficient multi-server user authentication and key agreement scheme with privacy protection. Let ID_i be a unique identification of user i. Also, let x be the master secret key kept secretly by the server S.

3.1 The proposed scheme

The proposed scheme is as follows.
Registration Phase: Assume U_i submits his identity ID_i and his password PW_i to the server S for registration. If S accepts this request, he will perform the following steps:
Step 1: Compute U_i's secret information $\alpha_i = h(x, ID_i)$ and $\beta_i = \alpha_i \oplus PW_i$. Compute the pseudo identification number $\lambda_{i,1} = h(\alpha_i||ID_i||1)$ and records $(k = 1, \lambda_{i,1}, ID_i)$ in an identification table.
Step 2: Store ID_i, $\lambda_{i,1}$, $k = 1$, and β_i to the memory of a smart card and issue this smart card to U_i or send them secretly to U_i.
User Authentication and Session Key Agreement Phase:
If U_i wants to log into S anonymously, he must attach his smart card to a card reader. He then inputs his identity ID_i and his password PW_i to this device. The following protocol is the kth login with respect to this smart card.
Step 1: $U_i \rightarrow S : N_1, \lambda_{i,k}, E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$
Step 2: $S \rightarrow U_i : N_2, E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2))$
Step 3: $U_i \rightarrow S : E_{sk_k}(N_2 + 1)$

In step 1, $U_i's$ smart card first computes $\alpha_i = \beta_i \oplus PW_i$ and sends his pseudo identification $\lambda_{i,k} = h(\alpha_i||ID_i||k)$, a nonce N_1 and the encrypted message $E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$ to S. The encrypted message includes the kth random value ru_k, which is used for generating the kth session key sk_k, and the authentication tag $h(N_1||ru_k||\lambda_{i,k})$, which is for verifying the identification of U_i.

Upon receiving the message in step 1, S first searches the pseudo identification $\lambda_{i,k}$ in the identification table to retrieve ID$_i$. He then computes $\alpha_i = h(x, ID_i)$ and decrypts the message $E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$ and verifies if the authentication tag $h(N_1||ru_k||\lambda_{i,k})$ is valid. If it is valid, S sends a nonce N_2 and the encrypted message $E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2))$ back to U_i. The encrypted message includes the random value rs_k chosen by S, which is used for generating the kth session key sk_k.

Upon receiving the message in step 2, U_i decrypts the message by computing $D_{\alpha_i}(E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2)))$. He then checks if the authentication tag $h(rs_k||N_1||N_2)$ is in it for freshness checking. If yes, U_i computes the next pseudo identification $\lambda_{i,k+1} = h(\alpha_i||ID_i||k+1)$, the kth session key $sk_k = h(rs_k, ru_k, \alpha_i)$, and updates $(\lambda_{i,k+1}, k + 1)$

and sends the encrypted message $E_{sk_k}(N_2 + 1)$ back to S.

After receiving the message in step 3, S decrypts the message by computing $D_{sk_k}(E_{sk_k}(N_2+1))$ and checks if the nonce $N_2 + 1$ is in it for freshness checking. He then computes $\lambda_{i,k+1} = h(\alpha_i||ID_i||k + 1)$ and updates $(k+1, \lambda_{i,k+1}, ID_i)$ in the identification table. Then U_i and S can use the session key sk_k in secure communication soon.

Table 1: Efficiency comparison between our single server scheme and other related scheme

	Our scheme	Juang's scheme [6]
D1	128 bits	128 bits
D2	384 bits	256 bits
D3	2 Hash	1 Hash
D4	6 Sym + 7 Hash	6 Sym + 3 Hash

D1: Password length
D2: Communication cost of authentication for cryptographic parameters
D3: Computation cost of registration
D4: Computation cost of authentication
Hash: Hashing operation Exp: Exponential operation
Sym: Symmetric encryption or decryption

3.2 Security Analysis

(1) Identity protection:

Compared with Juang's scheme [6], our proposed single-server scheme can achieve the ability of identity privacy. The adversary can not derive the user's identity ID_i or the secure key α_i from $\lambda_{i,k} = h(\alpha_i||ID_i||k)$. When the user wants to login, he first inputs his correct password. If the password is matching, then the smart card computes $\alpha_i = h(x, ID_i)$ and sends message $N_1, \lambda_{i,k}, E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$ to the server. The adversary can not know the user identity since this message does not include the plaintext about the user identity ID_i.

(2) Mutual authentication

The server and the user must achieve authentication each other. That means the server must verify the user's identity. Similarity, the user must also confirm whether the server is legal. The goal of mutual authentication is to create an agreement session key $sk_k = h(rs_k, ru_k, \alpha_i)$ between the user and the server [1, 6, 7].

Let A and B denote the user and the server, respectively. Let $A \xleftrightarrow{sk_k} B$ denote the player A shares a session key sk_k with the player B. Thus, the mutual authentication is between the player A and the player B if there exists a session key sk_k, and A believes $A \xleftrightarrow{sk_k} B$ and B believes $A \xleftrightarrow{sk_k} B$. A strong mutual authentication should include the statement:

A believes B believes $A \xleftrightarrow{sk_k} B$ and B believes A believes $A \xleftrightarrow{sk_k} B$

In step 1 of the user authentication and session key agreement phase, after B receives the message $E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$, he will compute $D_{\alpha_i}(E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k})))$ with using the shared key α_i of A and B. Then B can check if this message contains authenticator $h(N_1||ru_k||\lambda_{i,k})$. If yes, B chooses a random number rs_k and sends message $N_2, E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2)$ to A. B then computes the kth session key $sk_k = h(ru_k, rs_k, \alpha_i)$ and believe $A \xleftrightarrow{sk_k} B$.

In step 2 of the user authentication and session key agreement phase, upon receiving the message $N_2, E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2))$, A decrypts the message $D_{\alpha_i}(E_{\alpha_i}(rs_k, h(rs_k||N_1||N_2))$ and confirms if this message contains the authenticator $h(rs_k||N_1||N_2)$. If yes, A generates a session key $sk_k = h(ru_k, rs_k, \alpha_i)$ and believe $A \xleftrightarrow{sk_k} B$. Since N_1 is chosen by A, A will believes B believes $A \xleftrightarrow{sk_k} B$.

In step 3 of user authentication and session key agreement phase, after B received

$E_{sk_k}(N_2+1)$, he will decrypt this message $E_{sk_k}(N_2+1)$ with the kth session key sk_k and get N_2+1. Then B checks if N_2 which is sent by him is correct. If yes, B believes A believes $A \overset{sk_k}{\longleftrightarrow} B$.

(3)Session key agreement

The session key $sk_k = h(ru_i, rs_i, \alpha_i)$ is not known to anybody but S and U_i since the random values ru_i, rs_i are encrypted by the secret key $\alpha_{i.}$

(4) Withstanding attacks

We prove our scheme can resist to following attack.

1. The man-in-middle attack [16]

   Since either ends of communicators can verify that the message is sent by the peer though the authenticators. The adversary has no way to forge a message, so this attack can be prevented on our scheme.

2. The dictionary attack [2]

   For computing the session key k_i, the adversary must know ru_i, rs_i and α_i, where the entropy of ru_i, rs_i or α_i is very large. The shared key α_i is only kept by the user and the server, so the session key are not be computed by the adversary.

3. The replay attack [17]

   Replay attack is simply replaying the message to the user or the server. For instance, the user just logins one time to server, but the adversary replays these authentication messages to the server for getting the permission of extra logins. To avoid these kind of attacks, our proposed scheme use nonces N_1, N_2, N_2+1 to resist them.

4. The modification attack [18]

   The modification attack is a disturbance attack. The purpose of this attack is that both the server and the user can not normal communicate each other. The server and the user consider they have the same session key k_i, but they have different session key k_i' in fact. Our proposed scheme can also resist this attack. Upon the message $N_1, \lambda_{i,k}, E_{\alpha_i}(ru_k, h(N_1||ru_k||\lambda_{i,k}))$ in step 1 of the user authentication and session key agreement phase, the adversary can not add or modify this message since the adversary does not has the share key α_i. If the adversary modify the message, the server will reject this message since the authentication tag is invalid. In the other hand, the user have the same process to prevent this attack.

5. The stolen-verifier attack [3]

   For achieving the ability of user anonymity, we use a pseudo identification $\lambda_{i,k} = h(\alpha_i||ID_i||k)$ to communicate with the server. If the $\lambda_{i,k}$ is known by the attacker, the attacker is still difficult to derive the user's real identification ID_i since the shared key α_i is protected by the secure one-way hash function $h()$ and the entropy of α_i is very large.

6. The insider attack [11]

   The weak password PW_i used in our scheme is only for protecting the corresponding smart card from being used by illegal users. If a user uses PW_i to access several servers for his convenience, the insider of the server can not impersonate the user to access other servers if this server do not have the corresponding smart card. We can replace $\beta_i = \alpha_i \oplus PW_i$ with $\alpha_i \oplus h(b \oplus PW_i)$ and use the checking method mentioned in [11] for protecting the weak password being known by the server. But this approach will need the user to remember the random number

b and input it after getting the smart card. The most important assumption for the server is protecting his master secret key x secretly. If this master secret key x is compromised, then the total system is insecure.

3.3 Performance considerations

We evaluate the efficiency of our scheme and Juang's scheme in Table 1. First, we assume the block size of secure symmetric cryptosystems is 128 bits and the output size of secure one way hashing functions is 128 bits. Because both our proposed single-server scheme and Juang's scheme are based on symmetric key cryptosystem, the performance is very well. In our scheme and [6], the password length only 128 bits is required. Our proposed scheme needs 384 bits for the user authentication. Both ours and Juang's scheme [6], the computation cost for registration is only needed one hash operation. The computation cost are aggregated operation numbers, including encryption operations, decryption operations or hashing operations. The encryption and encryption operations may be asymmetric or symmetric cryptosystem. In the login and session key agreement phase of our scheme, three symmetric key encryptions, three symmetric key decryptions and seven hash operations are required. In that of Juang's scheme [6], only three symmetric key encryptions, three symmetric key decryptions and three hash operation are required. The computation cost of the login and session key agreement is not including cost of generating session key. Although our proposed scheme has a little high communication and computation cost than Juang's scheme [6], but our scheme have more complete functionality.

The functionality comparison between our proposed scheme and related scheme is given in Table 2. Compared with Juang's scheme [6], our scheme can completely satisfy the listed properties but Juang's scheme [6] have no ability of privacy protection since it only transmits user identity to server for initial authentication.

4 Multi-server authentication and key agreement with user anonymity

There are three kinds of participants in our multi-server protocol: a key distribution centre, service providers (servers) and users. Let KDC denote the trusted key distribution centre, U_i denote user i, S_j denote service provider j . Let UID_i be a unique identification of U_i and SID_j be a unique identification of service provider j. Also, let x be the master secret key kept secretly by the key distribution centre KDC and $\delta_j = h(x, SID_j)$ be the secret key shared by S_j and KDC. The shared secret key δ_j can be computed by KDC and sent secretly to S_j after he registered at KDC.

4.1 The proposed scheme

The proposed scheme is as follows.

Registration Phase: Assume U_i submits his identity UID_i and his password PW_i to KDC for registration. If KDC accepts this request, he will perform the following steps:

Step 1: Compute U_i's secret information $\alpha_i = h(x, UID_i)$ and $\beta_i = \alpha_i \oplus PW_i$.

Step 2: Store UID_i, and β_i to the memory of a smart card and issue this smart card to U_i or send them secretly to U_i.

Shared Key Inquiring Phase: If U_i wants to use the services provided by S_j, he must inform S_j to query the shared key

Table 2: Functionality comparison between our single server scheme and other related scheme

	Our scheme	Juang's scheme [6]
C1	Yes	No
C2	Yes	Yes
C3	Very low	Very low
C4	Yes	Yes
C5	Yes	Yes
C6	Yes	Yes

C1: Privacy protection
C2: Freely chosen password
C3: Communication and computation cost
C4: Mutual authentication
C5: Session key agreement
C6: No serious time-synchronization problem

$\gamma_{i,j}$ from KDC in advance. KDC will compute $\gamma_{i,j} = h(\alpha_i \oplus SID_j)$,where α_i is shared key with U_i, and then sends $\gamma_{i,j}$ to S_j.They will perform the following steps:

Step 1: $U_i \rightarrow S_j : N_1, UID_i$
Step 2: $S_j \rightarrow KDC : N_1', SID_j, E_{\delta_j}(UID_i, h(UID_i||SID_j||N_1'))$
Step 3: $KDC \rightarrow S_j : E_{\delta_j}(\gamma_{i,j}, h(UID_i|| SID_j||N_1'||\gamma_{i,j}))$
Step 4: $S_j \rightarrow U_i : E_{\gamma_{i,j}}(N_1 + 1)$

In Step 1, U_i sends a nonce N_1, his identification UID_i to S_j for informing S_j to query the shared key $\gamma_{i,j}$ from KDC.

Upon receiving the message in Step 1, S_j first checks if U_i had logined before. If not, he sends a nonce N_1', his identification SID_j and the encrypted message $E_{\delta_j}(UID_i, h(UID_i\ ||SID_j||N_1'))$ to KDC.

Upon receiving the message in Step 2, KDC decrypts the message $E_{\delta_j}(UID_i, h(UID_i||\ SID_j||N_1'))$, and checks if the verification tag $h(UID_i||SID_j||N_1')$ is valid and the nonce N_1' is fresh. For checking the freshness of the nonce N_1', KDC can keep a recently used nonces table. If yes, he then sends the encrypted message $E_{\delta_j}(\gamma_{i,j}, h(UID_i||SID_j||N_1'||\gamma_{i,j}))$ back to S_j.

Upon receiving the message in Step 3, S_j decrypts the message $E_{\delta_j}(\gamma_{i,j}, h(UID_i\ ||UID_j||\ N_1'||\gamma_{i,j}))$ and checks if the verification tag $h(UID_i||UID_j||N_1'||\gamma_{i,j})$ is valid. If yes, he records $(UID_i, 1, \lambda_{i,j,1} = h(\gamma_{i,j}||UID_i||SID_j||1),\ \gamma_{i,j})$ in a key table and then sends the encrypted message $E_{\gamma_{i,j}}(N_1 + 1)$ back to U_i.

Upon receiving the message in Step 4, U_i decrypts the message $E_{\gamma_{i,j}}(N_1 + 1)$ and checks if $N_1 + 1$ is in it for freshness checking. If yes, then the pseudo identification registration in S_j has been finished.

User Authentication and Session Key Agreement Phase:

If U_i wants to logs into S_j anonymously, he must attach his smart card to a card reader. He then inputs his identity UID_i and his password PW_i to this device. The following protocol is the kth login for U_i with respect to S_j.

Step 1: $U_i \rightarrow S_j : N_2, \lambda_{i,j,k}, E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,j,k}))$
Step 2: $S_j \rightarrow U_i : N_3, E_{\gamma_{i,j}}(rs_k, h(rs_k|| N_2||N_3))$
Step 3: $U_i \rightarrow S_j : E_{sk_k}(N_3 + 1)$

In step 1, $U_i's$ smart card first computes $\alpha_i = \beta_i \oplus PW_i$ and $\gamma_{i,j} = h(\alpha_i, SID_j)$ and sends his pseudo identification $\lambda_{i,j,k}$, a nonce N_2 and the encrypted message

$E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k\ ||\lambda_{i,j,k}))$ to S_j. The encrypted message includes the kth random value ru_k, which is used for generating the kth session key sk_k, and the authentication tag $h(N_2||ru_k||\lambda_{i,j,k})$, which is for verifying the identification of U_i.

Upon receiving the message in step 1, S_j first searches the pseudo identification $\lambda_{i,j,k}$ in the key table. He then decrypts the message $E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,j,k}))$ and verifies if the authentication tag $h(N_2||ru_k||\lambda_{i,j,k})$ is valid using the shared key $\gamma_{i,j}$ in the matched entries. If yes in some entry, the corresponding valid user identification UID_i is found. If it is valid and the nonce N_2 is fresh, S_j sends a nonce N_3 and the encrypted message $E_{\gamma_{i,j}}(rs_k,h(rs_k||N_2||N_3))$ back to U_i. The encrypted message includes the random value rs_k chosen by S_j, which is used for generating the kth session key sk_k, and the nonce N_3, which is for freshness checking.

Upon receiving the message in step 2, U_i decrypts the message by computing $D_{\gamma_{i,j}}(E_{\gamma_{i,j}}\ (rs_k,h(rs_k||N_2||N_3)))$. He then checks if the authentication tag $h(rs_k||N_2||N_3)$ is in it for freshness checking. If yes, U_i computes the next pseudo identification $\lambda_{i,j,k+1} = h(\gamma_{i,j}||UID_i||SID_j||k+1)$, the kth session key $sk_k = h(rs_k, ru_k, \gamma_{i,j})$, and records $SID_j, \lambda_{i,j,k}$ in a table and sends the encrypted message $E_{sk_k}(N_3+1)$ back to S_j.

After receiving the message in step 3, S_j decrypts the message by computing $D_{sk_k}(E_{sk_k}(\ N_3+1))$ and checks if the nonce N_3+1 is in it for freshness checking. He then computes $\lambda_{i,j,k+1} = h(\gamma_{i,j}||UID_i||SID_j||k+1)$ and updates $(UID_i, k+1, \lambda_{i,j,k+1} = h(\gamma_{i,j}||UID_i||SID_j\ ||k+1),\ \gamma_{i,j})$ in the key table. Then U_i and S_j can use the session key sk_k in secure communication soon

4.2 Security Analysis

(1) Identity protection:

Similarity, our proposed multi-server scheme can offer user identity protection. So the adversary can not know the user identification. In the user authentication and session key agreement phase, the user first sends a message $N_2, \lambda_{i,j,k}, E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,i,k}))$ to the server. Because this message does not include user identification UID_i , the adversary can not know the user identification.

(2) Mutual authentication

In step 1 of the user authentication and session key agreement phase, after S_j receives the message $E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,j,k}))$, S_j will compute $D_{\gamma_{i,j}}(E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,j,k})))$ using the share key $\gamma_{i,j}$ of U_i and S_j. Then S_j can check if this authenticator $h(N_2||ru_k||\lambda_{i,j,k})$ is valid. If yes, S_j chooses a random number rs_k and can computes the kth session key $sk_k = h(ru_k, rs_k, \gamma_{i,j})$ and believes $U_i \xleftrightarrow{sk_k} S_j$.

In step 2 of the user authentication and session key agreement phase, upon receiving the message $N_3, E_{\gamma_{i,j}}(rs_k,h(rs_k||N_2||N_3))$, U_i decrypts the message $D_{\gamma_{i,j}}(E_{\gamma_{i,j}}(rs_k,h(rs_k||N_2||N_3))$ and confirms if this message contains the authenticator $h(rs_k||N_2||N_3)$. If yes, U_i generates a session key $sk_k = h(ru_k, rs_k, \gamma_{i,j})$ and believe $U_i \xleftrightarrow{sk_k} S_j$. Since N$_2$ is chosen by U$_i$, U_i will believes S$_j$ believes $U_i \xleftrightarrow{sk_k} S_j$.

In step 3 of the user authentication and session key agreement phase, after S_j receiving $E_{sk_k}(N_3+1)$, he will decrypt this message $E_{sk_k}(N_3+1)$ with the kth session key sk_k and get N_3+1. Then S_j checks if N_3 which is sent by him is correct. If yes, S_j believes U_i believes $U_i \xleftrightarrow{sk_k} S_j$.

(3)Session key agreement.

The session key $sk_k = h(ru_k, rs_k, \gamma_{i,j})$ is known to nobody but S_i and U_j, since the random values ru_k, rs_k are randomly chosen by the user and the server and are encrypted by the shared key $\gamma_{i,j}$.

(4) Withstanding attacks

We prove our scheme can resist to following attack

1. The man-in-middle attack [16]

   Our proposed multi-server scheme also can resist to the man-in-the-middle attack. If the message is modified by the adversary, either ends of the communication will find out and reject this message. Since our proposed scheme can accomplish strong mutual authentication, our scheme can resist this attack.

2. The dictionary attack [2]

   For deriving the session key sk_k, the adversary must know ru_k, rs_k and $\gamma_{i,j}$ but the shared key $\gamma_{i,j}$ is only kept secretly by the user, the server and KDC. The adversary can not get the session key sk_k, since ru_i and rs_i are randomly chosen and protected by the shared key $\gamma_{i,j}$ and the entropy of ru_k, rs_k or $\gamma_{i,j}$ is very large.

3. The replay attack [17]

   The replay attack is simply replaying the message to the user or the server. Our multi-server scheme also provide an ability to avoid this attack. Our proposed scheme uses nonces $N_2, N_3, N_3 + 1$ to resist the replay attack.

4. The modification attack [18]

   Upon the message $N_2, \lambda_{i,i,k}, E_{\gamma_{i,j}}(ru_k, h(N_2||ru_k||\lambda_{i,j,k}))$ in step 1 of the user authentication and session key agreement phase, the adversary can not alter this message since the adversary does not has the share key $\gamma_{i,j}$. If the adversary modify the message, the server will reject this message. In the other hand, the user also can observe the original message whether is changed by the adversary. So this attack on our scheme can be prevented.

5. The stolen-verifier attack [3]

   In our proposed multi-server scheme, we use a pseudo identification $\lambda_{i,j,k} = h(\gamma_{i,j}||UID_i||SID_j||k)$ for user anonymity. Without knowing $\gamma_{i,j} = h(\alpha_i, SID_j)$, the attacker can not get the user's real identification UID_i since the entropy of $\gamma_{i,j}$ is very large. Our proposed multi-server scheme can withstand the stolen-verifier attack.

6. The insider attack [11]

   The function of the weak password PW_i in our multi-server scheme is the same with that in our single server scheme. The most important assumption for KDC is protecting his master secret key x secretly. If this master secret key x is compromised, then this multi-server system is insecure. The most important assumption for the server S_j is protecting his shared key table $\gamma_{i,j}$ secretly. If his shared key table is compromised, then this server is insecure.

4.3 Performance considerations

In this subsection, we present a efficiency comparison among our proposed scheme, Yang et al.'s scheme [19] and Juang's scheme [7]. The comparison is given in Table 3. We also assume that n in Yang et al.'s scheme [19] that has the same assumption with Lin et al.'s scheme [10] is of 1024 bits in order to make the discrete logarithm

Table 3: Efficiency comparison between our multi-server scheme and other related schemes

	Our scheme	Yang et al.'s scheme [19]	Juang's scheme [7]
E1	256 bits	1024 bits	256 bits
E2	384 bits	5 $\times$ 1024 bits	256 bits
E3	1 Hash	2 Exp	1 Hash
E4	6 Sym + 5 Hash	None	4 Sym +2 Hash
E5	6 Sym + 7 Hash	9 Exp + 2 Sym + 2 Hash	7 Sym + 3 Hash

E1: Memory needed in the smart card
E2: Communication cost of the authentication for cryptographic parameters
E3: Computation cost of the registration
E4: Computation cost of the shared key inquiring
E5: Computation cost of the user authentication and key agreement
Hash: Hashing operation Exp: Exponential operation
Sym: Symmetric encryption or decryption

problem infeasible. Moreover, we also assume both the output size of secure one-way hashing functions and the block size of secure symmetric cryptosystems are 128 bits. In our scheme and Juang's scheme [7], the memory needed in the smart card is 256 bits. In [19], However, the memory needed in the smart card is 1024 bits since their scheme based on the intractability of the discrete logarithm problem. The communication cost of the user authentication of our scheme and Juang's scheme [7] is 384 and 256 bits respectively. In [19], the communication cost for the authentication is 5 $\times$ 1024 bits. In our scheme and Juang's scheme [7], the computation cost of registration is one hash operation. In that phase, that is two exponentiation operations in Yang et. al.'s scheme. In our scheme, the computation cost of the shared key inquiring phase is needed three symmetric key encryptions, three symmetric key decryptions, five hash operations and one exclusive-or operation. In Juang's scheme [7], that is needed two symmetric key encryptions, two symmetric key decryptions, two hash operations. That phase of Yang et al.'s scheme [19] is not required. The computation cost of anonymous user identification in our scheme is three symmetric key encryptions, three symmetric key decryptions and seven hash operations. The computation cost of user identification in Juang's scheme [7] is three symmetric key encryptions, four symmetric key decryptions and three hash operations. The computation cost of anonymous user identification in Yang et al.'s scheme [19], nine exponential operations, one symmetric key encryptions, one symmetric key encryptions, and two hash operations are required. Note that the computation cost of our scheme, Juang et al.'s scheme[7] and Yang et al.'s scheme[19] do not accounted cost of generating session key.

We summarize the functionality and complexity of related scheme in Table 4. Our scheme can satisfy all listed functions and has low communication and computation cost. In comparison with Yang et al.'s, our proposed scheme have low communication and no time synchronization problems since using symmetric key cryptosystems and nonces to prevent replay attack, respectively. In comparison with Juang's scheme [7], our scheme provides an ability of privacy protection which is not provided by Juang [7].

Table 4: Functionality comparison between our multi-server scheme and other related schemes

	Our scheme	Yang et al.'s scheme [19]	Juang's scheme [7]
C1	Yes	Yes	No
C2	Yes	Yes	Yes
C3	Very low	High	Very low
C4	Yes	Yes	Yes
C5	Yes	Yes	Yes
C6	Yes	No	Yes

C1: Privacy protection
C2: Freely chosen password
C3: Communication and computation cost
C4: Mutual authentication
C5: Session key agreement
C6: No serious time synchronization problem

5 Discussions

For practical implementation, the smart cards used in our schemes can be issued by the trusted key distribution center and assumed to be tamperproof devices. For protecting $U_i's$ smart card from being used by an illegal user, a weak password PW_i can be chosen and used to protect it. Its role is like the personal identification number (PIN) used in the current banking system. If some illegal user uses the smart card by wrong passwords exceeding some fixed times, the operating system of the smart card will block the login procedure.

Using the factoring method proposed in [9], factoring a 512-bit moduli can be done in less than ten minutes on a US$10K device and factoring a 1024-bit moduli can be done in a year on a US$10M device in 2003. Differently from the schemes [19] using public-key cryptosystems, only symmetric cryptosystems and one-way hashing functions are used in our proposed schemes. Our approach provides another choice for better efficiency and no need to base on any assumed hard number theoretical problem, e.g., the factoring problem or the discrete logarithm problem. In practical considerations, one-way hash functions can be easily constructed by symmetric cryptosystems [12]. This approach can reduce the needed memory in smart cards for storing cryptographic programs.

In step 1 of our proposed schemes, the pseudo identification $\lambda_{i,k} = h(\alpha_i||ID_i||k)$ in section 3 or $\lambda_{i,j,k} = h(\gamma_{i,j}||UID_i||SID_j||k)$ in section 4 for the kth transaction is used for protecting the privacy of user i. After the server receiving the pseudo identification $\lambda_{i,k}$ or $\lambda_{i,j,k}$, he will search this entry in the key table and find the corresponding real identification. By sending the transaction value k in step 1 of our proposed schemes, all possible pseudo identification $\lambda'_{i,k}$ or $\lambda'_{i,j,k}$ can be easily computed online and then compared with the received pseudo identification $\lambda_{i,k}$ or $\lambda_{i,j,k}$ by the server for saving the storage.

In our scheme, for improving the repairability mentioned in [5, 11], the secret value $\alpha_i = h(x, UID_i)$ stored in each U_i's smart card can be replaced with the new formula $\alpha_i = h(x, UID_i, j)$, where j is the number of times that U_i has revoked his used secret key α_i. But this approach will need the key distribution center to record the number j in his database or U_i needs to send the number j to the server in the authentication phase. The password changing procedure proposed in [5, 11] can be directly used in our proposed schemes for changing users' passwords.

Like the schemes in [6, 7], we do not provide the perfect forward secrecy in our proposed schemes, since it may cause a result of lower performance and increased communication and computation cost. If this property is required, the Diffie-Hellman algorithm [4] can be directly applied to our schemes as in the schemes [6, 7]. Yang et al.'s scheme [19] has a serious time-synchronization problem, since their scheme is based on time-stamps. For example, when receiving the message (x, s, y, T) from the user, the server would believe the user is legal if $T' - T < \triangle T$ where T' is the receiving time of the server and T is the sending time of the user. Our proposed schemes solve this serious problem, because we use nonces to prevent the replay attacks.

6 Conclusions

In this paper, we have proposed two user authentication and key agreement schemes with privacy protection for single server and multi-server environments. Regarding the single-server scheme, it is more simple and efficient. Regarding the multi-server scheme, users only need to register one time and can use all provided services by service providers. Both our proposed schemes have the ability of privacy protection. Our schemes also have low communication and computation cost for user authentication by only using symmetric cryptosystems and one-way functions. Also, our schemes successfully solve the serious time-synchronization problem in a distributed computers environment since our proposed schemes are nonce-based.

Acknowledgments. This work was supported in part by the National Science Council of Republic of China under contract NSC 94-2213-E-128-001.

References

[1] M. Burrows, M. Abadi and R. Needham, "A Logic of Authentication," *ACM Trans. on Computer Systems*, Vol. 8, No. 1, pp. 18-36, 1990.

[2] S. Bellovin and M. Merritt, "Encrypted key Exchange: Password-Based Protocols Secure Against Dictionary Attacks," Research in Security and Privacy, Proceedings IEEE Computer Society Symposium, pp. 72-84, 1992.

[3] Y. Chang and C. Chang, "Authentication Schemes with no Verification Table," Applied Mathematics and Computation, Vol. 167, pp.820-832, 2005.

[4] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, Vol. IT-22, No. 6, pp. 644-654, 1976.

[5] T. Hwang and W. Ku, "Repairable Key Distribution Protocols for Internet Environments," IEEE Trans. on Communications, Vol. 43, No. 5, pp. 1947-1950, 1995.

[6] W. Juang, "Efficient Password Authenticated Key Agreement Using Smart Cards," *Computers & Security*, Vol. 23, No. 2, pp. 167-173, 2004.

[7] W. Juang, "Efficient Multi-server Password Authenticated Key Agreement Using Smart Cards," *IEEE Trans. on Consumer Electronics*, Vol. 50, No. 1, pp. 251-255, 2004.

[8] L. Lamport, "Password Authentication With Insecure Communication," *Communications of the ACM*, Vol. 24, pp. 770-772, 1981.

[9] A. Lenstra, E. Tromer, A. Shamir, W. Kortsmit, B. Dodson, J. Hughes and P. Leyland, "Factoring Estimates for a 1024-bit RSA Modulus," In Laih, C. (ed.), Advances in Cryptology-AsiaCrypt'03, Lecture Notes in Computer Science, 2894, pp. 55-74, Springer, New York, 2003.

[10] I. Lin, M. Hwang and L. Li, "A New Remote User Authentication Scheme for Multi-server Architecture," *Future Generation Computer Systems*, Vol. 19, pp. 13-22, 2003.

[11] W. Ku and S. Chen, "Weaknesses and Improvements of an Efficient Password Based Remote User Authentication Scheme Using Smart Cards," *IEEE Trans on Consumer Electronics*, Vol. 50, No. 1, pp. 204-207, 2004.

[12] R. Merkle, "One Way Hash Functions and DES," In Brassard, G. (ed.), Advances in Cryptology-Crypt'89, Lecture Notes in Computer Science, 435, pp. 428-446, Springer, New York, 1989.

[13] NIST FIPS PUB 197, "Announcing the ADVANCED ENCRYPTION STANDARD(AES)," National Institute of Standards and Technology, U. S. Department of Commerce, Nov., 2001.

[14] NIST FIPS PUB 180-2, "Secure Hash Standard," National Institute of Standards and Technology, U. S. Department of Commerce, DRAFT, 2004.

[15] R. Rivest, "The MD5 Message-digest Algorithm," RFC 1321, Internet Activities Board, Internet Privacy Task Force, 1992.

[16] D. Seo and P. Sweeney," Simple Authenticated Key Agreement Algorithm,". *Electronics Letters*, Vol. 35, pp. 1073 - 1074, 1999.

[17] P. Syverson, "A Taxonomy of Replay Attacks," Computer Security Foundations Workshop VII,. CSFW 7. Proceedings 14-16, pp. 187-191, 1994.

[18] C. Yang, T. Chang and M. Hwang, "Cryptanalysis of Simple Authenticated Key Agreement Protocols," *IEICE Trans. Fundamentals*, Vol. E87-A, No. 8, pp. 2174-2176, 2004.

[19] Y. Yang, S. Wang, F. Bao, J. Wang, R. Deng, "New Efficient User Identification and Key Distribution Scheme Providing Enhanced Security," *Computers and Security*, Vol. 23, No. 8, pp. 697-704, 2004.