Mutual Password Authentication with Key Agreement and User Anonymity for Cloud Storage

Linmei Jiang Xiaochao Li Donghui Guo*

School of Information Science and Engineering, Xiamen University,

Xiamen, 361005, Fujian, China

clough@hqu.edu.cn, {leexcjeffrey, dhguo}@xmu.edu.cn

Received 19 November 2014; Revised 2 February 2015; Accepted 19 March 2015

Abstract. A mutual password authentication scheme which makes use of discrete logarithm problem (DLP) in finite fields to encrypt messages and takes USB token and password as the two authentication factors to enhance security is proposed in this paper. Meanwhile, a secure cloud storage access model which applies and verifies the good performance of the proposed scheme is introduced. In addition, some criteria of the USB token-based password authentication is summarized from previous works and is used to check the security and functionality of the proposed scheme through detailed cryptanalysis. Furthermore, comparison from theoretical analysis and experimental results with other related authentication schemes shows that the proposed scheme is feasible to cloud storage with high efficiency.

Keywords: authentication, user anonymity, key agreement, USB token, cloud storage

1 Introduction

Nowadays, cloud computing has gained a considerable acceptance for its prominent features such as economic costs and access anywhere and any time. Nevertheless, in the cloud computing environment, the users' data are under the control of un-trusted cloud service provider and accessed over the Internet, which is widely reckoned an open and insecure scenario that may be subjected to various attacks such as password guessing, Man-in-the-Middle, client impersonation, and so on. Thus, it is the key point to guarantee the identity of a user is what he claimed to be through a secure and efficient identity authentication. In the mean time, a user may also want to keep his identity from being exposed to any adversary because it may cause the leakage of privacy. That is to say, identity authentication with user anonymity is preferred in the cloud computing. Nevertheless, although many authentication schemes has been proposed, very few of them supports user anonymity while resisting various attacks and keeping the merits such as key agreement, no verification table, low computation costs, and so on.

It is well known that authentication can be achieved through 3 basic factors, i.e. something known by user, something possessed by user and the unique biological characteristic of user [1]. Single-factor password authentication schemes which usually keep users' identifiers and passwords in a verification table in the server are widely implemented in real world applications due to its convenience, e.g., Huawei OceanStor, IBM Storwize, and Amazon S3. However, such schemes are vulnerable to various attacks such as stolen-verifier attack, modification attack and password guessing attack. A direct way to overcome the weakness of single-factor password authentication scheme is to remove the verification table from the remote server. Meanwhile, to make a system more secure, it is highly recommended to combine two or more of the factors mentioned above [2]. When it comes to selecting the other factor for password authentication, the second factor, e.g. smart card and USB token, usually takes precedence due to its low-computation cost and convenient portability. On the contrary, although the third factor such as iris and palmprint could provide higher authentication accuracy, it is too expensive to be applied for the cases with common confidentiality [1]. Therefore, many authentication schemes that take advantage of the first two factors have been proposed [2-8]. Unfortunately, although many good properties are achieved, there still exist various defects in these schemes. For example, Wang et al.'s scheme [3] is subjected to stolen-verifier attack and modification attack because a verification file is used to store the users' password verifier. Fan et al.'s scheme [4] and Lee et al.'s scheme [6] are vulnerable to insider attack because they directly encrypt the password verifier with the server's long secret key. Xu et al.'s scheme [5] may suffer from insider attack because it sends the plain password of the user to the server in the registration phase. Song's scheme [7] may suffer from off-line password guessing attack as analyzed by Cheng et al. [9]. Khan's scheme [8] needs the clock synchronization because a timestamp is employed to resist replay attack. In addition, these schemes are based on smart card, which makes them less feasible for cloud computing than USB

* Correspondence author

token based schemes such as Tao et al.'s scheme [1] and Liu et al.'s scheme [10], because a smart card always needs a card reader which is scarcely available for a common user whereas a USB token has a card reader built in.

Among all the existed password authentication schemes, "a password authentication scheme over insecure networks" proposed by Liao et al. [2] is widely referenced as the representative of DLP (discrete logarithm problem) based schemes. This scheme eliminates the requirement of verification table via using smart card to store users' privacy and is claimed to meet various security requirements by the authors. However, Liao et al's scheme was pointed out that it may be vulnerable to offline password guessing attack and server spoofing attack by Yang et al. [11] and Xiang et al. [12] respectively. Moreover, Xiang et al. [12] indicated that Liao et al's scheme may also be subjected to denial of service attack. Besides, there is still a clock synchronization problem in Liao et al's scheme, and it can not withstand insider attack as claimed by the authors themselves. In order to overcome all the flaws existed in Liao et al's scheme [2], we have proposed an improved authentication scheme in 'the 7th IEEE International Conference on Anti-counterfeiting, Security, and Identification (Shanghai)' [13], in which the main idea of our scheme is briefly introduced. In this paper, the scheme is further improved on the anonymity part and corresponding experiments are conducted to compare the performance between Liao et al's scheme and ours. Besides, more detailed analysis is made to illustrate that our scheme not only keeps all the merits and eliminates all the flaws existing in Liao et al's scheme, but also achieves some other good properties and be more suitable for cloud computing.

The rest of this paper is organized as follows: In Section 2, a secure cloud storage access model is introduced and the criteria for USB token or smart card based password authentication is listed and explained. In Section 3, the mutual password authentication scheme with key agreement and user anonymity using USB token is illustrated minutely. In Section 4, functionality and security analysis of the proposed scheme is implemented and comparison with some related schemes is made. In section 5, the experimental results are shown to compare the efficiency of the proposed scheme with Liao et al's scheme. Finally, the conclusion of the paper is drawn in Section 6.

2 Secure Cloud Storage Access Model and USB Token Based Authentication Criteria

In this section, a secure cloud storage access model which utilizes the proposed scheme is introduced, and some criteria for USB token or smart card based password authentication is summarized.

2.1 A Secure Cloud Storage Access Model

Today, as the rapid development of networking technologies, networked storages becomes more and more prevalent. For example, network file systems such as FTP server, SSH server and subversion server are widely used to store various documents or binary data, and online relational databases built in all kinds of portal sites keep large amount of users' info. Moreover, the networked storage servers are increasingly built in cloud, which means the servers are most likely to be maintained by a third-party called CSP (Cloud Service Provider) such as Amazon S3, IBM XIV, Windows Azure, and so on. Whereas, a dishonest CSP can deceive the clients [14], and it may manage any leakage of data even by helping the rival parties, so the CSP can not be trusted blindly [15]. When it comes to these untrusted servers, the privacy and integrity of data becomes the first concern, thus people always encrypt data before uploading them to the remote server and the server should build a well designed cloud storage access model to protect the data from being accessed by illegal person through well designed authentication scheme.

To enhance data security, in figure 1, we proposed a cloud storage access model which consists of three entities: data owner, CSM(cloud storage mediator) and CSP. In this model, CSM is the center of storage security management which is responsible for authenticating data owners and forwarding the data storage request to CSP rather than stores the users' data itself. It is very convenient for the data owner to manage data through a WEB based front-end, and it is strong enough for a data owner to ensure the data privacy and integrity through dual-factor authentication and data encryption. To use the cloud storage service, a data owner should first open the data-processing page in a web browser which is responded by the web server of the CSM when an HTTP request is sent to. There are three components contained in the data-processing page, i.e. the authentication component for authenticating the legality of a user, the data processor for packaging (indexing, encrypting and archiving) and unpacking (extracting and decrypting) data, and the data transfer component for transferring the packages between data owner and the CSM over the Internet. After the data-processing page is opened and before data are processed or transferred, the data owner should plug in the USB token and enter the corresponding password to prove his own identity to the CSM through the authentication component in which the proposed scheme is implemented. Only after the authentication phase is completed successfully, could the data be transferred between the data owner and the CSP under the control of the CSM.

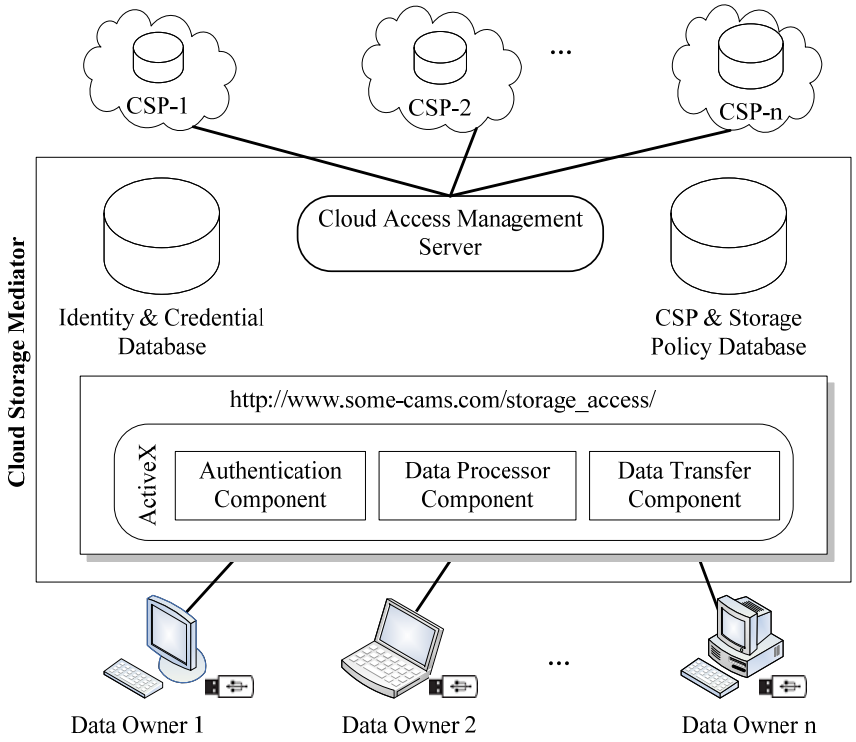


Fig. 1. A secure cloud storage access model

2.2 Password Authentication Criteria

Since many authentication schemes have been proposed and widely studied, researchers have put forward various criteria for password authentication with dual-factor based on password and USB-token or smart-card. According to previous researches [2, 16-19], a USB token or smart card based password authentication scheme should meet the following requirements in terms of functionality and security. The functionality requirements are: ① No verification table; ② Mutual authentication; ③ Freely choosing and updating password; ④ Session key agreement; ⑤ Prevention of clock synchronization problem; ⑥ User anonymity. While, the security requirements are: ① Resistance to password guessing attack; ② Resistance to insider attack; ③ Resistance to impersonation attack; ④ Resistance to server spoofing attack; ⑤ Resistance to replay attack; ⑥ Resistance to denial of service (DOS) attack; ⑦ Resistance to USB token loss attack. In addition, it is worth to know that as long as a scheme achieved the property of no verification table, it can resist two kinds of attacks. One is stolen-verifier attack which means the user's password verifier may be stolen from the verification table, the other one is modification attack which means an intruder can somehow break into the server and modify the password verifier in the verification table [2]. Obviously, once no password-verifier is saved in verification table, there is nowhere for an adversary to stole or to modify the password-verifier.

3 Proposed Scheme

In this section, our mutual password authentication scheme with key agreement and user anonymity using USB token for cloud storage is presented. The notations we used are shown in Table 1.

Our approach is composed of 5 phases, namely, initialization phase, registration phase, password authentication phase, password change phase and anonymity authentication phase. In the initialization phase, the server manager selects a long secret key and proper prime number for the storage server and starts the service program. And in the registration phase, the users apply for registration to the server and the server issues USB token to them in turn. Then in the password authentication phase, a user inserts his USB token into a terminal and input the corresponding password to start an login request, and the remote server verifies and determines whether the user should be accepted or not. And through the password change phase, a user can freely choose and change his password. Finally, the anonymity authentication phase does the same things as the password authentication phase but provides user anonymity when it is important to keep the identity of a user secret. By

the way, the initialization phase is none of the users' concern, and the registration phase is needed to do only once for every user that wants to use the resource of the server.

The detailed descriptions of these phases are given below.

Table 1. Notations used in the proposed scheme

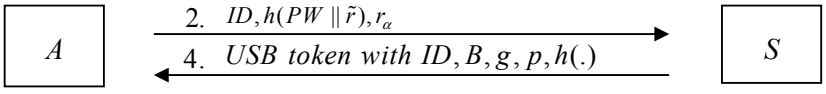
Notations	Description
A	A client.
S	The authentication server.
ID	The identifier of a client.
PW	Password.
x	The long secret key of the server S.
p	A large prime number.
g	The primitive element in Galois field GF(p)
$\tilde{r}, r, R$	Some large random numbers.
$\|\|$	Concatenation.
$\oplus$	Exclusive OR
$h(\cdot)$	One-way hash functions such as MD5 and SHA1.
$X \to Y : M$	Message M is sent from X to Y through an open channel.
$X \Rightarrow Y : M$	Message M is sent from X to Y through a secure channel.
SK	A session key.

3.1 Initialization Phase

Initially, the manager of storage server should select three proper parameters, namely g, p and $h(\cdot)$, to ensure the DLP secure enough. This could be done by executing the "dhparam" command provided by openssl. Then, a long secret key should be selected and kept secretly for the storage server. Once all the parameters are selected, they are fixed and couldn't be changed any more. And for safety reasons, it is recommended to split the secret key to multi-parts that are kept by different individuals respectively. When all the parameters are ready, the server manager starts the storage server to provide service for the end users.

3.2 Registration Phase

When a client A wants to use the resources of the system, he must first register to the server S with his own identifier ID and a corresponding password PW and acquire a USB token from the server provider. Figure 2 shows the registration phase of the proposed scheme and the detail is described in the following steps.



1. choose ID, PW and *generates* $\tilde{r}, r_\alpha$.
3. $B = g^{h(x\|ID)-h(PW\|\tilde{r})} \bmod p$.
   $ID_\alpha = r_\alpha \oplus ID$.
   Stores ID and ID_α *in the identity table.*
5. write $\tilde{r}, r_\alpha$ to USB token.

Fig. 2. Registration phase through secure channel

1) The client A chooses his ID and PW at will, and generates two random numbers $\tilde{r}$ and r_α.

2) $A \Rightarrow S : ID, h(PW \| \tilde{r}), r_\alpha$. The client A figures out $h(PW \| \tilde{r})$, then sends his ID, $h(PW \| \tilde{r})$ and r_α to the server S via a secure channel or by hand to start a registration request.

3) After receiving the registration request, the server S calculates $ID_\alpha = ID \oplus r_\alpha$ and $B = g^{h(x||ID)-h(PW||\tilde{r})} \ mod \ p$, where ID_α is only for anonymity password authentication. Then, the server S stores ID and ID_α in the identity table as shown in table 2.

4) $S \Rightarrow A$: USB token with $ID, B, g, p, h(\cdot)$. The server S writes $ID, B, g, p, h(\cdot)$ to a USB token and issues it to the client A through a secure channel.

5) Upon getting the USB token, the client A writes $\tilde{r}$ and r_α into it. Meanwhile it is the responsibility of the USB token to guarantee that B and r_α can only be changed under the control of the authorized application of the client and all the other data saved in it are read-only.

Table 2. Identity table

Identifier	Anonymous Identifier	Status
ID	ID_α	true / false
⋮	⋮	⋮

3.3 Password Authentication Phase

When a client A wants to access the server S, he must plug in his own USB token and enters the corresponding PW. Then, the following steps are performed during the password authentication phase like what is shown in Figure 3.

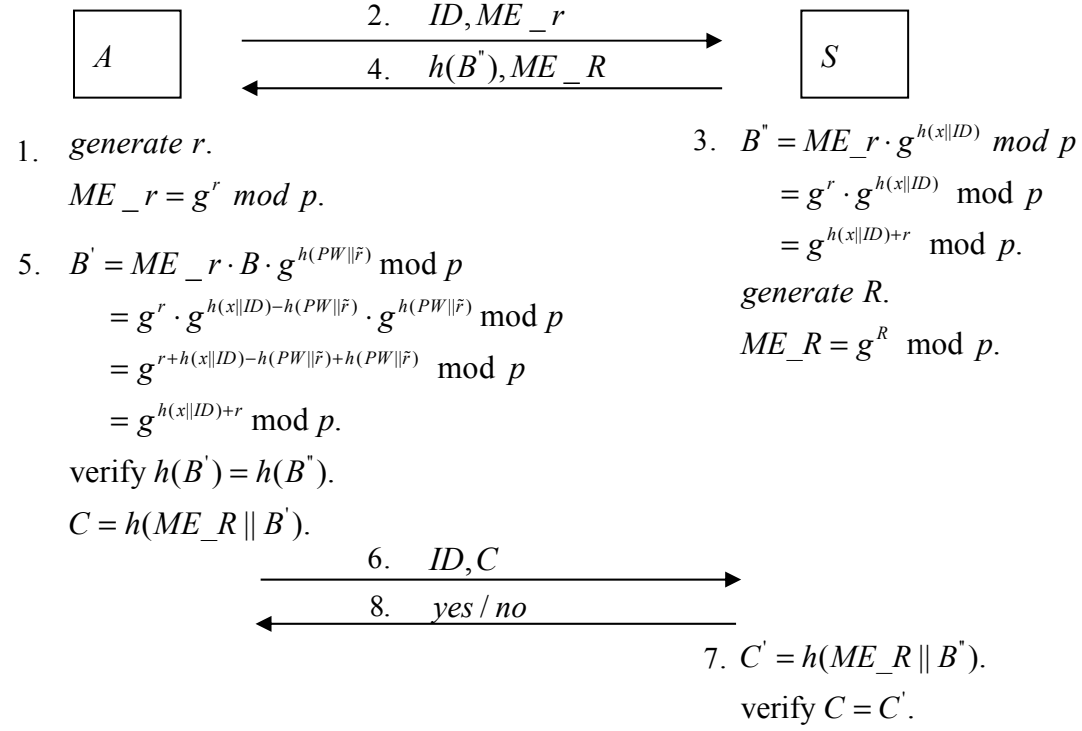


Fig. 3. Password authentication phase

1) The client A generates a random number r and calculates $ME\_r = g^r \ \text{mod} \ p$.

2) $A \rightarrow S: ID, ME\_r$. The client A starts a login request by sending ID and ME_r to the server S.

3) After receiving the login request, the server S calculates $B'' = ME\_r \cdot g^{h(x||ID)} \ mod \ p = g^r \cdot g^{h(x||ID)} \ \text{mod} \ p = g^{h(x||ID)+r} \ \text{mod} \ p$, where x is server's secret key only known and maintained by

the server S. Subsequently, it generates a random number R, then calculates $ME\_R = g^R \mod p$ and $h(B'')$.

4) $S \rightarrow A : h(B''), ME\_R$. The server S sends $h(B'')$ and ME_R to the client A.

5) While receiving the message, the client A calculates $B' = ME\_r \cdot B \cdot g^{h(PW||\tilde{r})} \mod p$ $= g^r \cdot g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \mod p = g^{r+h(x||ID)-h(PW||\tilde{r})+h(PW||\tilde{r})} \mod p = g^{h(x||ID)+r} \mod p$. Then the client A verifies the validity of the server S by checking whether the received message $h(B'')$ is equal to $h(B')$. If not, the server is rejected and the login phase is repeated; otherwise, the client A calculates $C = h(ME\_R \,||\, B')$.

6) $A \rightarrow S : ID, C$. The client A sends ID and C to the server S.

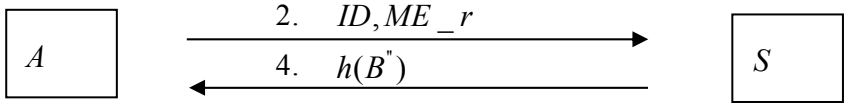
7) The server S computes $C' = h(ME\_R \,||\, B'')$, and then verifies whether C is equal to C'. If true, S accepts the login request; otherwise, S rejects the login request because the client A has no proper USB token or doesn't know the corresponding password.

8) $S \Rightarrow A : yes / no$. The server S sends yes or no to the client A to notify whether the login request is accepted or denied.

In this phase, two random numbers r and R are chosen by the client A and the server S respectively and they are sent to each other in the form of $g^r \mod p$ and $g^R \mod p$. Therefore, the final session key $SK = (g^R \mod p)^r \mod p = g^{r \cdot R} \mod p = (g^r \mod p)^R \mod p$ can be computed by both of the client and the server.

3.4 Password Change Phase

To change the password of A, the USB token of the client A must be inserted; the old PW and a newly chosen password $\overline{PW}$ must be entered. Then, the following steps are performed. Figure 4 shows the password change phase.



1. $generate\ r.$
$ME\_r = g^r \ mod\ p.$

3. $B'' = ME\_r \cdot g^{h(x||ID)} \ mod\ p$
$= g^r \cdot g^{h(x||ID)} \mod p$
$= g^{h(x||ID)+r} \mod p.$

5. $B' = ME\_r \cdot B \cdot g^{h(PW||\tilde{r})} \mod p$
$= g^r \cdot g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \mod p$
$= g^{h(x||ID)+r} \mod p.$
verify $h(B') = h(B'')$.

6. $\overline{B} = B \cdot g^{h(PW||\tilde{r})} \cdot g^{-h(\overline{PW}||\tilde{r})} \mod p$
$= g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \cdot g^{-h(\overline{PW}||\tilde{r})} \mod p$
$= g^{h(x||ID)-h(\overline{PW}||\tilde{r})} \mod p.$

Fig. 4. Password change phase

1) The client A generates a random number r and calculates $ME\_r = g^r \ mod\ p$.

2) $A \rightarrow S : ID, ME\_r$. The client A sends ID and ME_r to the server S.

3) The server S calculates $B^{''} = ME\_r \cdot g^{h(x||ID)} \ mod \ p = g^r \cdot g^{h(x||ID)} \ \mathrm{mod} \ p = g^{h(x||ID)+r} \ mod \ p$, then calculates $h(B^{''})$.

4) $S \rightarrow A : h(B'')$. The server S sends $h(B^{''})$ to the client A.

5) The client A figures out $B^{'} = ME\_r \cdot B \cdot g^{h(PW||\tilde{r})} \ \mathrm{mod} \ p = g^r \cdot g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \ \mathrm{mod} \ p = g^{h(x||ID)+r} \ \mathrm{mod} \ p$, then calculates $h(B')$. If $h(B') \neq h(B'')$, the password change request is rejected.

6) The client A calculates $\overline{B} = B \cdot g^{h(PW||\tilde{r})} \cdot g^{-h(\overline{PW}||\tilde{r})} \ \mathrm{mod} \ p = g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \cdot g^{-h(\overline{PW}||\tilde{r})} \ \mathrm{mod} \ p = g^{h(x||ID)-h(\overline{PW}||\tilde{r})} \ \mathrm{mod} \ p$. In the end, B is replaced by $\overline{B}$ in the USB token.

3.5 Anonymity Authentication Phase

Anonymity, namely the secrecy of the identities of communicating agents, is becoming a major concern in many multi-user electronic commerce and industrial engineering applications; and there are dozens of different flavors of anonymity [20]. As far as an authentication scheme is concerned, there are two levels of anonymity being suggested to achieve. One is sender anonymity which says that attackers may not know who the senders are; the other is sender untraceability which says that attackers may neither know the sender's identity nor tell whether two conversations are originated from the same agent [21]. In this section, we improve the anonymity phase we proposed in literature [13], which achieves sender anonymity, to support sender untraceability. Figure 5 shows the anonymity authentication phase and the detailed steps are described as below.

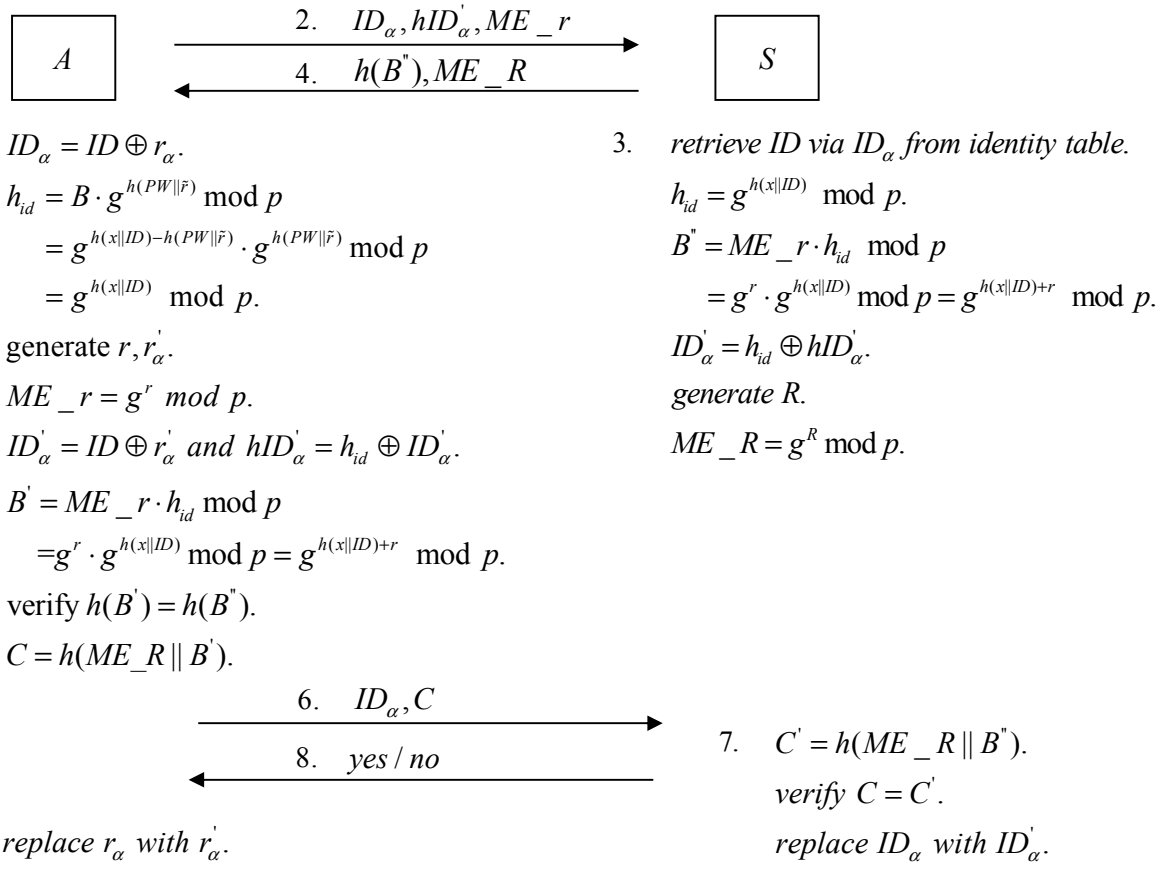


Fig. 5. Anonymity authentication phase

1) The client A first calculates the anonymous identifier $ID_\alpha = ID \oplus r_\alpha$, then calculates $h_{id} = B \cdot g^{h(PW||\tilde{r})} \ \mathrm{mod} \ p = g^{h(x||ID)-h(PW||\tilde{r})} \cdot g^{h(PW||\tilde{r})} \ \mathrm{mod} \ p = g^{h(x||ID)} \ mod \ p$. Thereafter, the client A generates two random numbers r and $r_\alpha^{'}$, then calculates $ME\_r = g^r \ \mathrm{mod} \ p$. Lastly, the client A

calculates $ID_\alpha^{'} = ID \oplus r_\alpha^{'}$ and $hID_\alpha^{'} = h_{id} \oplus ID_\alpha^{'}$, where $ID_\alpha^{'}$ is the new anonymous identifier for the next anonymity authentication phase and $hID_\alpha^{'}$ is aimed at transferring $ID_\alpha^{'}$ to the server secretly.

2) $A \rightarrow S : ID_\alpha, hID_\alpha^{'}, ME\_r$. The client A starts an anonymity login request by sending ID_α, $hID_\alpha^{'}$ and ME_r to server S.

3) On receiving the login request, the server S first retrieve the true ID by searching ID_α in the identity table and calculates $h_{id} = g^{h(x||ID)} \bmod p$, then calculates $B^{''} = ME\_r \cdot h_{id} \ mod \ p$ $= g^r \cdot g^{h(x||ID)} \bmod p = g^{h(x||ID)+r} \bmod p$ and $h(B^{''})$. Subsequently, the server S retrieves $ID_\alpha^{'}$ by calculating $ID_\alpha^{'} = h_{id} \oplus hID_\alpha^{'}$. Thereafter, the server S generates a new random number R and calculates $ME\_R = g^R \bmod p$.

4) $S \rightarrow A : h(B''), ME\_R$. The server S sends $h(B^{''})$ and ME_R to the client A.

5) The client A calculates $B^{'} = ME\_r \cdot h_{id} \bmod p = g^r \cdot g^{h(x||ID)} \bmod p = g^{h(x||ID)+r} \bmod p$. If $h(B') \neq h(B'')$, the server is rejected and the login phase is repeated; otherwise, the client A continues to calculate $C = h(ME\_R \,||\, B^{'})$.

6) $A \rightarrow S : ID_\alpha, C$. The client A sends ID_α and C to the server S.

7) The server S calculates $C' = h(ME\_R \,||\, \mathrm{B}'')$ and verifies whether C is equal to C'. If false, the server S rejects the login request; otherwise, the server S accepts the login request and replaces the anonymous identifier ID_α with the new anonymous identifier $ID_\alpha^{'}$ in the identity table as shown in table 3, where $ID_\alpha^{'}$ is obtained in the step 3.

8) $S \Rightarrow A : yes / no$. The server S sends yes or no to the client A to notify whether the login request is accepted or denied.

9) If the login request is accepted by the server S, the client A replace r_α in the USB token with $r_\alpha^{'}$.

Table 3. Identity table after finishing anonymity authentication phase

Identifier	Anonymous Identifier	Status
ID	$ID_\alpha^{'}$	true / false
⋮	⋮	⋮

4 Functionality and Security Analysis

In this section, the functionality and security features of our scheme is analyzed by checking all the authentication requirements listed in section 2.

4.1 Functionality Features

Table 4. Functionality comparisons

Schemes/Functionalities	F1	F2	F3	F4	F5	F6
Liao et al's scheme [2]	Y	Y	Y	Y	N	N
Fan et al's scheme [4]	Y	Y	N*	N	Y	N
Xu et al's scheme [5]	Y	Y	Y	Y	N	N
Lee et al's scheme [6]	Y	Y	N	Y	N	Y
Song's scheme [7]	Y	Y	Y	Y	N	N
Proposed scheme	Y	Y	Y	Y	Y	Y

▪ *Y: supported; N: not supported; N*: Fan et al's scheme doesn't support freely updating password.*

To check all the functionality requirements listed in section 2, we give a brief comparison of the functionality features of the proposed scheme and some related schemes in table 4, which shows the proposed scheme achieves most functionality features.

F1: No verification table

In the proposed scheme, no verification table that keeps the password verifiers is used. Instead, an identity table is adopted to keep the mapping of the identifier and the anonymous identifier. Actually, an identity table is very useful instead of compromising system security, and is widely adopted by plenty of applications to identify the ownerships of system resources.

F2: Mutual authentication

In the step 5 of the password authentication phase, a client can verify the legitimacy of the remote server by checking whether $h(B') = h(B'')$ holds or not, where $B'' = g^{h(x||ID)} \cdot g^{r} \mod p$. Since only the remote server possesses the long secret key x, without doubt, only the remote server can calculate $h(B'')$. Hence, if the equation $h(B') = h(B'')$ holds, the client can make sure that he is communicating with the legal remote server.

On the other hand, in the step 7 of the password authentication phase, the server can verify the legitimacy of the client by checking whether $C = C'$ holds or not, where $C = h(ME\_R \,||\, B') = h(g^{R} \bmod p \,||\, B')$ and $B' = ME\_r \cdot h_{id} \bmod p = g^{r} \cdot B \cdot g^{h(PW||\tilde{r})} \bmod p$. Since only the legal client knows PW and possesses the USB token that contains B and $\tilde{r}$, only the legal client can calculate $h(B')$. Hence, if the equation $C = C'$ holds, the remote server can assure itself that the client is legal. Therefore, our scheme achieves mutual authentication.

F3: Freely choosing and updating password

In the proposed scheme, the client can freely choose his ID and password in the registration phase. In the mean time, the proposed scheme allows every user to change his password at will through the password change phase.

F4: Session key agreement

As described in section 3, through a successful password authentication phase, the client and the server can agree on a common session key $SK = (g^{R} \bmod p)^{r} \bmod p = (g^{r} \bmod p)^{R} \bmod p = g^{r \cdot R} \bmod p$, where r and R are chosen by A and S respectively, and only the results of $g^{r} \bmod p$ and $g^{R} \bmod p$ are transferred in the communication channel. It is infeasible for an adversary to derive r and R to compute the session key SK, because it is equivalent to solve the DLP.

F5: Prevention of clock synchronization problem

To prevent replay attack, many previous password authentication schemes use the time stamp, e.g., Liao et al's scheme [2] and Khan's scheme [8]. It is exactly the time stamp that causes the clock synchronization problem. Thus, our scheme uses random numbers instead of the time stamp to withstand replay attack.

F6: User anonymity

In the anonymity authentication phase of the proposed scheme, the identifier of client A is encrypted as $ID_{\alpha} = ID \oplus r_{\alpha}$; the true ID and the random number r_{α}' which is used to form the next anonymous identifier $ID_{\alpha}' = ID \oplus r_{\alpha}'$, is encrypted in $hID_{\alpha}' = h_{id} \oplus ID_{\alpha}'$. Here, $h_{id} = g^{h(x||ID)} \mod p$ can only be figured out by the server S because x is the long secret key of the server. Therefore, only the server S can retrieve ID_{α}' and the sender anonymity property is achieved. Moreover, because r_{α} and r_{α}' are random numbers varied in every login session, ID_{α} would not be repeated twice, so the sender untraceability property is guaranteed. Therefore, in the whole login session, the client's identity is hidden and the user anonymity is achieved.

Table 5. Security comparisons

Schemes/Security characteristics	S1	S2	S3	S4	S5	S6	S7
Liao et al's scheme [2]	N	N	Y	N	Y	N	Y
Fan et al's scheme [4]	Y	N	Y	Y	Y	Y	N
Xu et al's scheme [5]	Y	N	Y	Y	Y	Y	Y
Lee et al's scheme [6]	Y	N	Y	N	Y	Y	Y
Song's scheme [7]	Y	N	Y	Y	Y	Y	Y
Proposed scheme	Y	Y	Y	Y	Y	Y	Y

▪ *Y: prevent the attack; N: unable to prevent the attack.*

4.2 Security Features

In Table 5, a brief comparison of the security features of the proposed scheme and other related schemes is made. Obviously, it shows our scheme prevents all related cryptographic attacks.

S1: Resistance to password guessing attack

In the proposed scheme, if an adversary intercepts the ID, $g^r \bmod p$, $h(B'')$, $g^R \bmod p$ and C, he cannot break the password PW by playing guessing attacks because none of them contains any part of PW. Furthermore, even if the USB token is lost, off-line guessing attacks are still impossible. Although an adversary can retrieve B, g, p, $\tilde{r}$, r_α, ID and $h(\cdot)$ from the lost USB token, he can never derive PW from $B = g^{h(x||ID)-h(PW||\tilde{r})} \bmod p$ without knowing x, where x is the long secret key maintained by the server S.

S2: Resistance to insider attack

In the proposed scheme, information that contains the password PW appears only once in the server, namely, in the registration phase and in the form of $h(PW \,||\, \tilde{r})$. It is computationally impossible to derive PW from $h(PW \,||\, \tilde{r})$. And without knowing $\tilde{r}$ which is saved in the USB token, a privileged-insider can not derive PW by playing off-line guessing attacks too.

S3: Resistance to impersonation attack

In this attack, an adversary may attempt to modify intercepted messages or forge a valid login message to masquerade the legal client and login to the server. However, the login request message C is calculated from equation $C = h(ME\_R \,||\, B^{'})$, where $B^{'} = ME\_r \cdot B \cdot g^{h(PW||\tilde{r})} \bmod p$ and $ME\_R = \mathrm{g}^R \bmod p$. It is impossible for the adversary to figure out C because he doesn't know any of B, $\tilde{r}$ and PW. Even if an adversary gets the lost USB token, he can not figure out C without knowing PW. Therefore, impersonation attack is infeasible to the proposed scheme.

S4: Resistance to server spoofing attack

In a server spoofing attack, an adversary may try to cheat the client who requests for service. To do so, in the proposed scheme, the adversary has to forge a valid response message $h(B'')$ in Step 2 of password authentication phase. That is infeasible because the adversary cannot figure out $B^{''} = g^{h(x||ID)} \cdot g^r \bmod p$ without knowing the long secret key x. Moreover, the adversary cannot spoof the client by replaying a previous $h(B'')$ because the random number r is different and only known by the client in each login request. Therefore the proposed scheme can resist server spoofing attack.

S5: Resistance to replay attack

In the proposed scheme, the password pattern C is used only once in the password authentication phase and it cannot be intercepted for reuse because the random number R is different and only known by the server S in each login request.

S6: Resistance to denial of service (DOS) attack

As depicted in [12], for scheme of Liao et al.[2], an adversary may tamper the data B even if he doesn't know the correct password through a password changing phase, where B contains the client's password and is saved in the smart card. These will cause the valid client can never be accepted by the server again. For another example, Liaw et al.'s scheme [22] has almost the same problem because an unauthorized user can easily create a new password for the smart card through its password change phase [23]. However, in the proposed scheme, the data B can only be changed in the password change phase in the condition of the user A knows the previous password PW, and the data r_α can only be changed in the anonymity authentication phase when a login request has been accepted by the server, and all the other data are read-only. Thus, there is no chance for an adversary to tamper the data saved in the USB token to make a denial of service attack in the proposed scheme.

S7: Resistance to USB token loss attack

In the proposed scheme, an adversary who gets a lost USB token cannot log-in the server S without knowing the password PW. Moreover, as described in **S1** of this section, even if the USB token is lost, off-line guessing attacks are still impossible. Thus, the proposed scheme can absolutely resist the USB token loss attack.

5 Performance Comparison

In order to evaluate the performance of the proposed scheme, we compare it with some other DLP based password authentication schemes in this section.

5.1 Computational Primitives Comparison

In this sub-section, we make a computational primitive comparison among the proposed scheme, Liao et al.'s scheme [2], Xu et al.'s scheme [5] and Lee et al.'s scheme [6]. Table 6 gives a brief review of their performance expressed with two key computations, one is the hash computation t_h, the other one is modulus exponentiation computation t_M. The computational time complexity of t_M is much higher than t_h. It's easy to see from the table that the proposed password authentication phase and anonymity password authentication phase have almost the same computational cost with Lee et al.'s scheme, and need one t_h less than Xu et al.'s scheme. In addition, our scheme needs one t_M more than Liao et al's no session key agreement scheme, but needs one t_M less than Liao et al's scheme with session key agreement. In view of the level of security and functionality the proposed scheme offers as explained in section 4, the performance of our scheme is quite satisfying.

Table 6. Computational primitives' comparison

Primitives	Proposed	Proposed, with Anonymity	Liao et al. [2]	Liao et al., with Key Agreement [2]	Xu et al. [5]	Lee et al. [6]
Client	$3t_h+2t_M$	$3t_h+2t_M$	$3t_h+2t_M$	$3t_h+3t_M$	$4t_h+2t_M$	$2t_h+2t_M$
Server	$3t_h+2t_M$	$3t_h+2t_M$	$3t_h+t_M$	$3t_h+2t_M$	$3t_h+2t_M$	$4t_h+2t_M$
Total	$6t_h+4t_M$	$6t_h+4t_M$	$6t_h+3t_M$	$6t_h+5t_M$	$7t_h+4t_M$	$6t_h+4t_M$

- *t_h is hash computation.*
- *t_M is modulus exponentiation computation.*

5.2 Time Cost Comparison in Implementation

To evaluate the actual computational efficiency of the proposed scheme, we give the results of the experiments that conducted in Visual Studio C++ 2010 with cryptographic library of openssl 0.9.8e in this sub-section. The specs of the computers we used in the experiments are list in table 7.

Table 7. Computer specs for experiments

C/S	CPU	Memory	OS
Client	Pentium (R) Dual Core CPU E5200 @ 2.5GHz	1.99 GB	Windows XP Professional with Service Pack 3
Server	Inter (R) Xeon (R) CPU E5620 @ 2.4GHz	12.0GB	windows 7 x64 professional

We implement the password authentication phase to measure the computation cost in two kinds of network environments. One is that the client and server are located in the same host which makes the main cost is the primitive computation, the other is that the client and server are located in different hosts in a LAN (Local Area Network) with bandwidth of 1.0Gbps which makes the main cost involving the communication time. On the other hand, in both network environments, we test the computation cost in prime number with difference size, where the primitive element is 2, the hash function type is MD5 or SHA1, the random numbers generated are 1024 bits, the user ID length are 4 to 10 characters, and the long secret key is 1024 characters. To make a fair comparison, we select and treat the parameters equally without discrimination to the related schemes by following the standard of the FIPS 140-2 [24].

Figure 6 and Figure 7 give the intuitive comparisons of the implementation results in the password authentication phase (the login phase and authentication phase in Liao et al's scheme [2]) , when client and server (C/S) are in the same host and in different hosts of the same LAN respectively. In both figures, the x axis is the combination of the hash function and the bits of the long secret key, and the y axis is the average time cost of 1000 times of authentications. Obviously, we can see from the figures that the cost increases when the bits of the long secret key increases and the cost varies little when the hash function changes. This is because modulus exponentiation computation contributes the main time cost. On the other hand, by comparing the two figures, we can see that the time cost in figure 7 is much more than that in figure 6. This is because when the client and server are located in different hosts, more than half of the time cost comes from the communication. Finally, we

can find that the implementation result truly reflects the computational primitives' comparison result, which shows the cost of our scheme is more than that of Liao et al's schemes, but is less than that of Liao et al's scheme with session key agreement. Nevertheless, as what has been explained in section 4, our scheme achieves more good features and resists more security attacks than any of the two schemes.

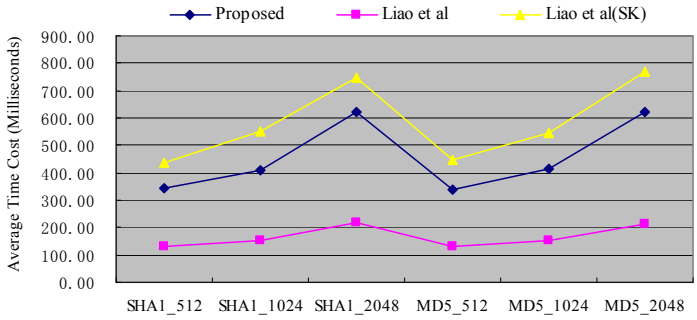


Fig. 6. Computation cost (C/S in the same host)

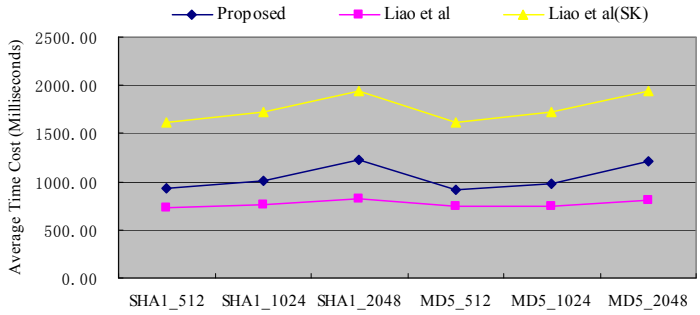


Fig. 7. Computation and Communication cost (C/S in a LAN)

6 Conclusion and Future Work

In this paper, a mutual password authentication scheme which makes great improvement to Liao et al's scheme [2] is illustrated and the prominent security properties and functionality properties achieved in the proposed scheme are demonstrated fully. In terms of security properties, our scheme can resist most of the well-known attacks. In terms of functionality properties, our scheme not only provides outstanding capabilities such as mutual authentication, freely changing password, no need for clock synchronization, low computation costs and session key agreement, but also supports the anonymity property of sender untraceability that makes the identities of all users be secret and untraceable. That is to say, our scheme not only overcomes all the defects existed in Liao et al's scheme, but also provides more merits to make it suitable for cloud computing environment.

A general defect of our scheme and the other USB token or smart card based schemes is that the server's security is based on a long secret key x, which can not be changed once it has been adopted. In order to enhance the security of the password authentication scheme, we are making further research on how to enable the server manager to change the long secret key.

7 Acknowledgement

This research was supported by Research Fund for the Doctoral Program of Higher Education of China under Grants No.20090121110019 and National Natural Science Foundation of China (General Program) under Grants No.61274133.

References

[1] Y. Tao, D. Tang, Gaoshan, Shenhao, "Design and implementation of USB key-based JavaEE dual-factor authentication system," in *Proceedings of 2009 International Conference on Information Management, Innovation Management and Industrial Engineering*, pp.443-446, 2009

[2] I.E. Liao, C.C. Lee, M.S. Hwang, "A password authentication scheme over insecure networks," *Journal of Computer and System Sciences*, Vol. 72, pp. 727-740, 2006.

[3] S.H. Wang, S.Q. Chang, Z.W. Wang, G.Z. Sun, "A password authentication and update scheme based on elliptic curve cryptography," *International Journal of Advancements in Computing Technology*, Vol. 4, pp. 84-90, 2012.

[4] C.I. Fan, Y.C. Chan, Z.K. Zhang, "Robust remote authentication scheme with smart cards," *Computers & Security*, Vol. 24, pp. 619-628, 2005.

[5] J. Xu, W. Zhu, D. Feng, "An improved smart card based password authentication scheme with provable security," *Computer Standards & Interfaces*, Vol. 31, pp. 723-728, 2009.

[6] C.C. Lee, C.T. Li, K.Y. Huang, S.Y. Huang, "An improvement of remote authentication and key agreement schemes," *Journal of Circuits Systems and Computers*, Vol. 20, pp. 697-707, 2011.

[7] R.G. Song, "Advanced smart card based password authentication protocol," *Computer Standards & Interfaces*, Vol. 32, pp. 321-325, 2010.

[8] M.K. Khan, S. Kim, K. Alghathbar, "Cryptanalysis and security enhancement of a `more efficient & secure dynamic ID-based remote user authentication scheme'," *Computer Communications*, Vol. 34, pp. 305-309, 2011.

[9] Z. Cheng, Y. Liu, C. Chang, S. Chang, "An improved protocol for password authentication using smart cards," *Journal of Computers*, Vol. 22, pp. 29-37, 2012.

[10] Z. Liu, L. Gu, Y. Yang, G. Xing, "An identity authentication scheme based on USB key for trusted network connect," in *Proceedings of 2010 IEEE International Conference on Information Theory and Information Security*, pp.203-207, 2010

[11] G.M. Yang, D.S. Wong, H.X. Wang, X.T. Deng, "Two-factor mutual authentication based on smart cards and passwords," *Journal of Computer and System Sciences*, Vol. 74, pp. 1160-1172, 2008.

[12] T. Xiang, K.W. Wong, X.F. Liao, "Cryptanalysis of a password authentication scheme over insecure networks," *Journal of Computer and System Sciences*, Vol. 74, pp. 657-661, 2008.

[13] L. Jiang, X. Li, L.L. Cheng, D. Guo, "Identity authentication scheme of cloud storage for user anonymity via USB token," in *Proceedings of the 7th IEEE International Conference on Anti-counterfeiting, Security, and Identification*, pp.1-6, 2013

[14] Y. Zhu, H.X. Hu, G.J. Ahn, M.Y. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 23, pp. 2231-2244, 2012.

[15] S.K. Sood, "A combined approach to ensure data security in cloud computing," *Journal of Network and Computer Applications*, Vol. 35, pp. 1831-1838, 2012.

[16] C.T. Li, "A new password authentication and user anonymity scheme based on elliptic curve cryptography and smart card," *IET Information Security*, Vol. 7, pp. 3-10, 2013.

[17] R.C. Wang, W.S. Juang, C.L. Lei, "Robust authentication and key agreement scheme preserving the privacy of secret key," *Computer Communications*, Vol. 34, pp. 274-280, 2011.

[18] W.S. Juang, "Efficient password authenticated key agreement using smart cards," *Computers & Security*, Vol. 23, pp. 167-173, 2004.

[19] B. Vaidya, J.H. Park, S. Yeo, J.J.P.C. Rodrigues, "Robust one-time password authentication scheme using smart card for home network environment," *Computer Communications*, Vol. 34, pp. 326-336, 2011.

[20] X. Li, W. Qiu, D. Zheng, K. Chen, J. Li, "Anonymity Enhancement on Robust and Efficient Password-Authenticated Key Agreement Using Smart Cards," *IEEE Transactions On Industrial Electronics*, Vol. 57, pp. 793-800, 2010.

[21] D. Hughes, V. Shmatikov, "Information hiding, anonymity and privacy: A modular approach," *Journal of Computer Security*, Vol. 12, pp. 3-36, 2004.

[22] H. Liaw, J. Lin, W. Wu, "An efficient and complete remote user authentication scheme using smart cards," *Mathematical and Computer Modelling*, Vol. 44, pp. 223-228, 2006.

[23] E.J. Yoon, S.H. Kim, K.Y. Yoo, "A Security Enhanced Remote User Authentication Scheme Using Smart Cards," *International Journal of Innovative Computing Information and Control*, Vol. 8, pp. 3661-3675, 2012.

[24] NIST, Security Requirements for Cryptographic Modules, FIPS PUB 140-2, 2001.