J Supercomput (2014) 70:133–148
DOI 10.1007/s11227-014-1135-8

An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures

Wen-Bin Hsieh · Jenq-Shiou Leu

Published online: 9 March 2014

Abstract As a smart phone becomes a daily necessity, mobile services are springing up. A mobile user should be authenticated and authorized before accessing these mobile services. Generally, mobile user authentication is a method which is used to validate the legitimacy of a mobile login user. As the rapid booming of computer networks, multi-server architecture has been pervasive in many network environments. Much recent research has been focused on proposing password-based remote user authentication protocols using smart cards for multi-server environments. To protect the privacy of users, many dynamic identity based remote user authentication protocols were proposed. In 2009, Hsiang and Shih claimed their protocol is efficient, secure, and suitable for the practical application environment. However, Sood et al. pointed out Hsiang et al.'s protocol is susceptible to replay attack, impersonation attack and stolen smart card attack. Moreover, the password change phase of Hsiang et al.'s protocol is incorrect. Thus, Sood et al. proposed an improved protocol claimed to be practical and computationally efficient. Nevertheless, Li et al. found that Sood et al.'s protocol is still vulnerable to leak-of-verifier attack, stolen smart card attack and impersonation attack and consequently proposed an improvement to remove the aforementioned weaknesses. In 2012, Liao et al. proposed a novel pairing-based remote user authentication protocol for multi-server environment, the scheme based on elliptic curve cryptosystem is more secure and efficient. However, through careful analyses, we find that Liao et al.'s protocol is still susceptible to the trace attack. Besides, Liao et al.'s protocol is inefficient since each service server has to update its

W.-B. Hsieh (✉) · J.-S. Leu
Department of Electronic Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan
e-mail: d9802106@mail.ntust.edu.tw

J.-S. Leu
e-mail: jsleu@mail.ntust.edu.tw

ID table periodically. In this paper, we propose an improved protocol to solve these weaknesses. By enhancing the security, the improved protocol is well suited for the practical environment.

Keywords User authentication · Pairing-based · Multi-server · Smart card

1 Introduction

Wireless communications have become a very attractive and interesting sector for the provision of electronic services. Mobile networks are available almost anytime and anywhere, and the popularity of wireless handheld devices is high. The services offered are strongly increasing because of the wide range of the users' needs [1]. With the rapid development of mobile networks, many mobile services are available online such as mobile banking, mobile government, mobile learning, mobile online game and so on. It is obvious that in future wireless protocols and communication environments (networks), security will play a key role in transmitted information operations [1]. As mobile devices bring a lot of convenience in our daily life, the security issues, how to authenticate mobile users in insecure communication networks, have become a hot research topic. To authenticate the identity of remote users in a public environment, the password-based authentication protocol was proposed in the first place. The first remote password-based authentication protocol for the insecure communication was proposed by Lamport [2] in 1981. In 2000, Hwang and Li [3] pointed out Lamport's protocol is vulnerable to the risks of interpolation attacks since the server must store the verifiers of users' passwords. Thus, Hwang and Li proposed a remote user authentication protocol using smart cards, which is based on ElGamal [4] public key cryptosystem. Since then, in order to lessen the communication and computation costs and remove the security issues, large number of smart card-based authentication protocols designed for single-server environment had been proposed [5–12]. However, it is difficult and bothersome for a user to remember numerous various identities and passwords when using single-server authentication protocol to login and access different remote servers. Therefore, Lee and Chang [13] proposed a user identification and key distribution protocol based on the difficulty of factorization and hash function, and agree with the multi-server environment. The user only needs to register at the registration center once and can access all the authorized services in remote servers. After that, much research devoted to the study of authentication of multi-server environments has been proposed [14–21].

From 2001, Li et al.'s [15] remote user authentication protocol based on the neural networks was found to spend too time and cost since users need large memory to store public parameters for authentication. After that, many protocols were continuously proposed to improve the previous ones. In 2008, Tsai [22] uses the nonce and one-way hash function to propose an efficient multi-server authentication protocol without a verification table. Because of low computation costs, Tsai's protocol is very suitable to be used in the distributed networks.

The aforementioned protocols are based on static ID which an adversary might intercept the login ID from the public network and use it to trace the legal user. In

2009, Liao and Wang [23] first proposed a dynamic ID-based remote user authentication protocol for multi-server environment. However, Hsiang and Shih [24] found that Liao et al.'s protocol is vulnerable to insider attack, masquerade attack, server spoofing attack, registration center spoofing attack, and is not reparable. Moreover, Liao et al.'s protocol cannot provide mutual authentication. Therefore, Hsiang et al. proposed an improved protocol to solve these problems. In 2011, Sood et al. [25] pointed out Hsiang et al.'s protocol is still susceptible to replay attack, impersonation attack and stolen smart card attack. Besides, the password change phase of their protocol is incorrect. Sood et al. proposed a secure dynamic identity based authentication protocol claimed to achieve user's anonymity and be against various attacks. Recently, Li et al. [26] discovered Sood et al.'s protocol still suffers leak-of-verifier attack, stolen smart card attack and impersonation attack. Furthermore, the authentication and session key agreement phase of Sood et al.'s protocol is not correct since the control server has no way to know the real identity of the user. In 2012, Liao et al. [27] present a novel pairing-based remote user authentication protocol for multi-server environment. The proposed scheme provided a more secure key distribution based on self-certified public keys (SCPKs) among the service servers, the protocol based on elliptic curve cryptosystem is more secure and efficient. However, through careful analyses, we find that Liao et al.'s protocol is still susceptible to the trace attack. Besides, Liao et al.'s protocol is inefficient since each service server has to update its ID table periodically. In this paper, we propose an improved protocol to solve these weaknesses. By enhancing the security, the improved protocol is well suited for the practical environment.

The rest of the paper is organized as follows. In Sect. 2, a brief review of Liao et al.'s protocol is presented. The security flaws and weaknesses of Liao et al's protocol are shown in Sect. 3. Our architecture and the improved protocol is proposed in Sect. 4. Section 5 makes the security analysis of the proposed protocol and Sect. 6 compares the performance and functionality of the proposed protocol with the related protocols. The conclusion is given in Sect. 7.

2 Review of Liao et al.'s protocol

The notations used in this paper are listed in Table 1.

In this section, we review Liao et al.'s self-certified public key-based authentication protocol for multi-server environment. The details of mathematics of computation, including bilinear pairings, the related computational problems, BLS short signature protocols and self-certified public key (SCPK) cryptosystems can be referred to [27–30]. There are five phases in Liao et al.'s protocol. These phases are explained in the following paragraphs. First, all needed parameters are generated by a key generator center (KGC) which is impersonated by the registration server *RS*. Next, the registration server RS chooses a random number $s_{RS} \in Z_q^*$ keeping as the system private key and computes $Pub_{RS} = s_{RS} \in P$ as the system public key, where P is a generator of the group G_1. Finally, two hash functions $H(\cdot)$ and $h(\cdot)$ are selected by the registration server RS. The public parameters and functions, Params $= \{\hat{e}, G_1, G_2, q, P, Pub_{RS}, H(\cdot), h(\cdot)\}$, are published.

Table 1 The notations used in this paper

$\hat{e}$	A bilinear map, $\hat{e} : G_1 \times G_1 \rightarrow G_2$
U_i	The ith user with the mobile device
SS_j	The jth service providing server
ID_i	The user U_i's identity
PW_i	The userU_i's password
SID_j	The service server SS_j's identity
RS	The registration server
s_{RS}	The master secret key of the registration server RS, $\{S_{RS} \in Z_q^*\}$
Pub_{RS}	The public key of the registration server RS, $Pub_{RS} = S_{RS} \cdot P$
P	Agenerator (base point) of group G_1
$H(\cdot)$	A map-to-point function, $H : \{0, 1\}^* \rightarrow G_2$
$h(\cdot)$	A one-way hash function, $h : \{0, 1\}^* \rightarrow \{0, 1\}^k$, where k is the output length, $h(\cdot)$ allows the concatenation of some integer values and points on an elliptic curve
$\oplus$	A exclusive-OR operation in G_1. If $P_1, P_2 \in G_1$, P_1 and P_2 are points on an elliptic curve over a finite field. The XOR operations of the x-coordinates and y-coordinates of P_1 and P_2 can be presented as $P_1 \oplus P_2$
	The XOR operations of the x-coordinates and y-coordinates of P_1 and P_2 can be presented as $P_1 \oplus P_2$
$\|\|$	Message concatenation operation

2.1 Server registration phase

When a service server SS_j joined in the multi-server environment, the service server SS_j first generates a random number $v_j \in Z_q^*$ and submits $V_j = v_j \cdot P$ along with the identity SID_j to the registration server RS. Then the registration server RS generates a random number $w_j \in Z_q^*$ and performs the following computations

$$W_j = V_j + w_j \cdot P$$
$$\hat{S}_j = (h(SID_j || W_j) \cdot S_{RS} + w_j) \bmod q$$

After that, the registration server RS issues the values s_j and W_j to the service server SS_j. Finally, the private key $s_j = (s_j + v_j) \bmod q$ is computed by the service server SS_j, and the service server SS_j computes the following equation to check the validity of the received values

$$Pub_j = s_j \cdot P = h(SID_j || W_j) \cdot Pub_{RS} + W_j \quad (1)$$

Now the service server has his own private key $s_j$and public key Pub_j after the service registration phase.

2.2 User registration phase

When the user U_i wants to access the services provided by the multi-server system, he/she has to register the registration server *RS* by submitting the related information. Next, the registration server *RS* issues the smart card containing some secret information to the user U_i over a secure channel. The process of user registration phase is depicted as follows.

U1. $U_i \rightarrow RS : \{ID_i, HPW_i\}$
The user U_i chooses a password PW_i and a random value $b_i \in Z_q^*$. Next the user U_i uses the password salting mechanism to calculate $HPW_i = h(PW_i||b_i) \cdot P$. Then the user U_i sends $\{ID_i, HPW_i\}$ to the registration server *RS*.

U2. $RS \rightarrow U_i$: the smart card containing $\{ID_i, Reg_{IDi}, Pub_{RS}, h(\cdot), H(\cdot)\}$
After receiving $\{H(ID_i), HPW_i\}$, the registration server *RS* checks if ID format is valid and if ID_i has already existed in the database. If both of them are correct, the registration server RS calculates $QID_i = H(ID_i), DID_i = (T_i s_{RS}) \cdot QID_i$ and $Reg_{IDi} = DID_i \oplus s_{RS} \cdot HPW_i$, where $T_i \in Z_q^*$ denotes the current registration time generated by the registration server. Finally, the registration server *RS* issues the smart card containing $\{ID_i, Reg_{IDi}, Pub_{RS}, H(\cdot), h(\cdot)\}$ to the user U_i over a secure channel. At the same time, the registration server *RS* maintains an ID table which includes $\{ID_i, T_i\}$. After that, the service servers received the updated entries periodically over globaly a secure channel.

U3. Upon receiving the smart card containing $\{ID_i, Reg_{IDi}, Pub_{RS}, H(\cdot), h(\cdot)\}$, the user U_i stores them with b in the smart card.

2.3 Login phase

After completing the registration phase, the user U_i can use the smart card issued by the registration server *RS* to login to the service server SS_j.

L1. $U_i \rightarrow$ the smart card: $\{ID_i, PW_i\}$
The user U_i inserts the smart card into the card reader and then enters his/her identity ID_i and password PW_i. The smart card generates a random number $r_i \in Z_q^*$ and calculates $R_i = r_i \cdot P$. The smart card then computes $DID_i = Reg_{IDi} \oplus h(PW_i||b_i) \cdot Pub_{RS}$, $QID_i = H(ID_i)$, $Mi = r_i \cdot QID_i$, $d_{ij} = h(ID_i||SID_j||M_i||R_i)$ and $B_{ij} = (r_i + d_{ij}) \cdot DID_i$.

L2. $U_i \rightarrow SS_j : \{ID_i, M_i, B_{ij}, R_i\}$
Finally, the user U_i with the smart card sends the login request $\{ID_i, M_i, B_{ij}, R_i\}$ to the service server SS_j.

2.4 Verification phase

V1. Upon receiving the login request $\{ID_i, M_i, B_{ij}, R_i\}$, the service server SS_j checks if ID_i is valid. If it is not valid, terminate this session; otherwise, the

corresponding registration time $T_i$can be obtained from the corresponding entry of the registration table.

V2. After obtaining the registration time T_i, the service server SS_j computes $QID_i = H(ID_i)$ and $d_{ij} = h(ID_i||SID_j||M_i||R_i)$. Next, the service server SS_j checks if the following equation can hold:

$$\hat{e} <B_{ij}, P>= \hat{e} <M_i + d_{ij} \cdot QID_i, T_i \cdot Pub_{RS}> \quad (2)$$

If it holds, the service server SS_j accepts the login request; otherwise, terminate this session.

V3. $SS_j \rightarrow U_i : (Auth_{ji}, K_{ji}, R_j)$
Then the service server SS_j generates a random point $R_j = r_j \cdot P$, the temporary key $TK_{ji} = r_j \cdot R_i$, the shared secret key $K_{ji} = s_j \cdot R_i$ and $Auth_{ji} = h(ID_i||K_{ji}||R_j)$. Next, the service server SS_j sends $(Auth_{ji}, K_{ji}, R_j)$ to the user U_i.

V4. $U_i \rightarrow SS_j : Auth_{ij}$
After receiving the responses $(Auth_{ji}, K_{ji}, R_j)$, the user U_i computes $Pub_j = h(SID_j||W_j) \cdot Pub_{RS} + W_j$ based on SCPK as the public key of the service server SS_j. Then the temporary key $TK_{ij} = r_i \cdot R_j$ and the shared secret key $K_{ij} = r_i \cdot Pub_j$ are also computed by the user U_i. Next the user U_i checks if the computed $h(ID_i||K_{ij}||R_j)$ is equal to the received $Auth_{ji}$. If they are equal, the user U_i computes $Auth_{ij} = h(ID_i||K_{ij}||R_i||R_j)$ and sends it to the service server SS_j.

V5. Upon receiving $Auth_{ij}$, the service server SS_j checks if the computed $h(ID_i||K_{ji}||R_i||R_j)$ is equal to the received $Auth_{ij}$. If they are equal, the two parties can compute the session key $SK = h(ID_i||T_{ij})$.

2.5 Password change phase

If the user U_i wants to change his/her password, the user U_i has to input his/her identity ID_i, original password PW_i and new password PW_{new}. Then the following process will be performed by the user U_i and the registration server RS.

P1. The user U_i first generates a random number $n_i \in Z_q^*$ and calculates $N_i = n_i \cdot P$

P2. $U_i \rightarrow$ RS: $\{ID_i, CID_i, N_i\}$
Second, the user U_i computes $DID_i = Reg_{IDi} \oplus h(PW_i||b_i) \cdot Pub_{RS}$ and $CID_i = DID_i \oplus n_i \cdot Pub_{RS}$. Then the user send a password change request $\{ID_i, CID_i, N_i\}$ to the registration server RS.

P3. $RS \rightarrow U_i : \{V_1\}$
Upon receiving the password update request $\{ID_i, CID_i, N_i\}$, the registration server checks the ID table to verify the validity of the user's identity. If the validity of the user's identity is confirmed, the registration server RS computes $QID = H(ID_i)$ and $DID_i = CID_i \oplus s_{RS}N_i$ to obtain DID_i. Then the registration server RS checks if $\hat{e} < DID_i, P>= \hat{e} < QID_i, T_iPu_{RS}>$. If it holds, the registration server RS sends V_1 to U_i by computing $V_1 = h(i_i||s_{RS}N_i)$.

P4. $U_i \rightarrow RS : \{V_2, V_3\}$
After receiving V_1, the user U_i compares $V_1$with the computed $h(DID_i||n_i Pub_{RS})$ to confirm the legality of the registration server RS. If the legality of the registration server $RS$is confirmed, the user U_i sends V_2 and V_3 to RS by computing $HPW_{new} = h(PW_{new}||b_i)$, $V_2 = HPW_{new} \oplus (n_i Pub_{RS})$ and $V_3 = h(DID_i||n_i Pub_{RS}||HPW_{new})$.

P5. $RS \rightarrow U_i : \{V_4\}$
Upon receiving the message (V_2, V_3), the registration server RS computes $V_2 \oplus (s_{RS}N_i)$ to extract HPW_{new}. Then the user U_i compares the received V_3 with the computed $h(DID_i||s_{RS}N_i||HPW_{new})$. If they are equal, the legality of the user $U_i$is confirmed. Finally, the registration server $Reg_{ID_i}^{new} = DID_i \oplus (s_{RS}HPW_{new})$ and sends V_4 to the user U_i by computing $V_4 = Reg_{ID_i}^{new} \oplus (s_{RS}N_i)$.

P6. After receiving V_4, the user U_i computes $V_4 \oplus (n_i Pub_{RS})$ to extract $Reg_{ID_i}^{new}$. Finally, the smart card of the user U_i replaces the original Reg_{IDi} with $Reg_{ID_i}^{new}$.

3 Cryptanalysis of Liao et al.'s protocol

In this section, we will show that Liao et al's protocol is vulnerable to the trace attack and spoofing server attack. According to Liao et al.'s scheme mentioned above, they claimed that their scheme can resist to a variety of attacks, save the computation cost and be applied well to the user with mobile devices. However, Liao et al.'s scheme causes some weaknesses in the following discussion.

To evaluate the security of smart card-based user authentication scheme, we first define the threat model which an attacker may have the following capability [31]: an attacker has total control over the communication channel between the mobile user and the remote server. That is the attacker may intercept, insert, delete or modify any message in the channel.

3.1 Trace attack

In the login phase of Liao et al.'s protocol, the user U_i sends the login messages containing the user's identity ID_i to service server SS_j without any protection. Since the user's identity ID_i is sent over an open communication channel, an attacker may intercept the message using the assumed capability. With the user's identity ID_i, an attacker can trace it to know what kind of services the user accesses, how long the user logins into the system. Since the service server may have the system log recording what the user did, the user's privacy may be leaked. Furthermore, an attacker may trace the user's location according to the user's IP address. The trace attack seriously invades the user's privacy and can be utilized to commit real crimes such as kidnappings.

3.2 A burden to update ID table

To authenticate newly added users, the entries $\{ID_i, T_i\}$ added to the ID table should be sent to all the service servers periodically over a secure channel. However, if a

new user want to access the service, the update may not be completed, thus the authentication of the new user will fail. Besides, every time the update of the ID table needs to build secure channels between the services servers and the registration server, it is an inefficient and inconvenient way to maintain a verification mechanism.

3.3 Lack of a pre-authentication in the smart card

In login phase, when user inputs his/her identity ID_i and password PW_i into the smart card, there is no verification of identity ID_i and password PW_i. The smart card will submit the login request to the service server. However, the login message may be false because of an unexisting identity, an incorrect password or both. This will increase trivial burdens on the service servers having to verify the certainly failed login messages. Furthermore, it may be utilized by an attacker to launch a DoS attack.

4 The proposed protocol

In this section, we propose an improved efficient and secure protocol to avoid the security vulnerabilities and inefficiencies of Liao et al.'s protocol. Also there are three entities in our protocol, i.e., the user (U_i), the service provider server (SS_j) and the register server (RS). RS chooses the master secret key $s_{RS} \in Z_q^*$ and keeps it as the system private key and computes $Pub_{RS} = s_{RS} \in P$as the system public key, where P is a generator of the group G_1. Subsequently, RS generates a random value T as a registration token. Next, two hash functions $H(\cdot)$ and $h(\cdot)$ are selected by the registration server RS. The public parameters and functions, Params $= \{\hat{e}, G_1, G_2, q, P, T, Pub_{RS}, H(\cdot), h(\cdot)\}$, are published. The proposed protocol also has five phases. Figure 2 illustrates detailed steps of the login phase and the verification phase.

We first depicted the conceptual architecture implemented in the proposed protocol. The architecture is illustrated as Fig. 1.

The smart phone is equipped with a micro SD card which is issued by the registration server. The secret parameters are burned into the micro SD card and will be automatically cleared if the card is removed from the smart phone. The details of our protocol are as the following.

Fig. 1 The login and verification of the proposed protocol

4.1 Server registration phase

For each service provider server SS_j joined in the multi-server environment, the service provider server SS_j first generates a random value $v_j \in Z_q^*$. Then the service provider server SS_j computes $V_j = v_j \cdot P$ and submits V_j along with the identity SID_j to the registration server *RS*. Next, the registration server *RS* generates a random value $w_j \in Z_q^*$ and performs the following computations

$$W_j = V_j + w_j \cdot P$$
$$\hat{S}_j = (h(SID_j||W_j) \cdot s_{RS} + w_j) \text{mod} q$$

After that, the registration token T, $\check{s}_j$and W_j is issued to the service server SS_j by the registration server *RS*. Finally, the private key $s_j = (\check{s}_j + v_j) \bmod q$ is computed by the service server SS_j, and the service server SS_j checks the validity of the received values by computing the following equation

$$Pub_j = s_j \cdot P = h(SID_j||W_j) \cdot Pub_{RS} + W_j \tag{3}$$

Now the service server has his own private key s_j, public key Pub_j and the registration token $H(T)$ after the service registration phase.

4.2 User registration phase

If the user U_i wants to access the services provided by the multi-server system, he/she has to submit the related information to register the registration server *RS*. Next, the registration server *RS* issues the smart card containing some secret information to the user U_i over a secure channel. The process of user registration phase is depicted as follows.

U1. $U_i \rightarrow RS : \{H(ID_i), HPW_i\}$ The user U_i chooses a password PW_i and a random value $b_i \in Z_q^*$. Next the user U_i uses the password salting mechanism to calculate $HPW_i = h(PW_i||b_i) \cdot P$. Then the user U_i sends $\{H(ID_i), HPW_i\}$ to the registration server *RS*.

U2. $RS \rightarrow U_i$: the smart card containing $\{Auth_i, T \cdot H(ID_i), Reg_{IDi}, Pub_{RS}, h(\cdot), H(\cdot)\}$
After receiving $\{H(ID_i), HPW_i\}$, the registration server *RS* checks if $H(ID_i)$ has already existed in the database. If it is not registered, the registration server *RS* calculates $DID_i = s_{RS}H(ID_i)$ and $Reg_{IDi} = DID_i \oplus s_{RS}HPW_i$ and $Auth_i = T \cdot H(ID_i)$. Finally, the registration server *RS* issues the smart card containing $\{Auth_i, Reg_{IDi}, Pub_{RS}, H(\cdot), h(\cdot)\}$ to the user U_i over a secure channel. At the same time, the registration server *RS* maintains the database which records $H(ID_i)$.

U3. Upon receiving the smart card containing$\{Auth_i, Reg_{IDi}, Pub_{RS}, H(\cdot), h(\cdot)\}$, the user U_i stores them with b_i and $CID_i = h(ID_i) \oplus h(PW_i||b_i)$ in the smart card.

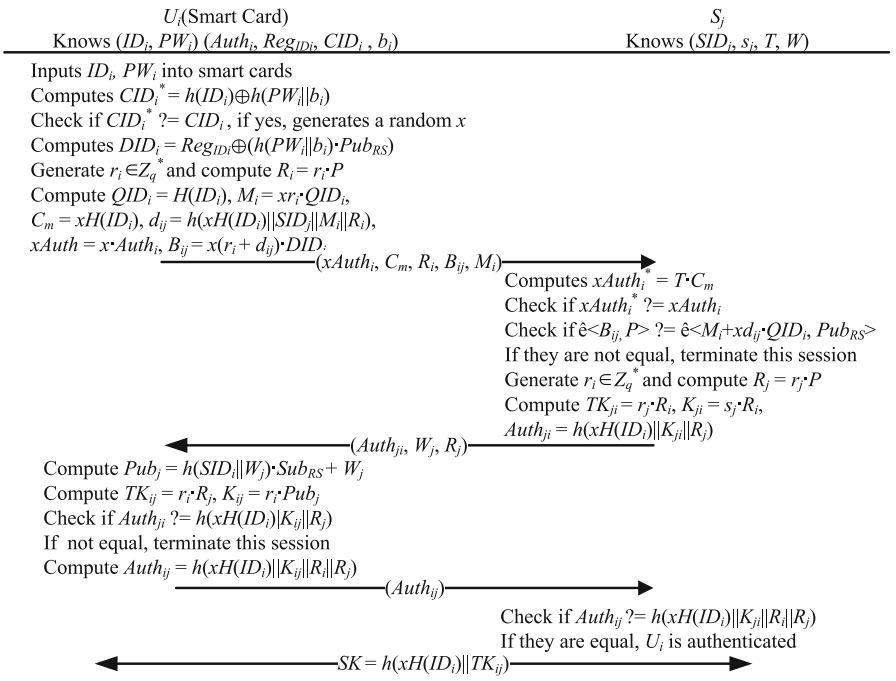

Fig. 2 The conceptual architecture of the proposed protocol

4.3 Login phase

After completing the registration phase, the user U_i can use the smart card issued by the registration server RS to login to the service server SS_j. The detailed processes of login and verification are shown in Fig. 2 and are depicted as follows.

L1. $U_i \rightarrow$ the smart card: $\{ID_i, PW_i\}$
The user U_i inserts the smart card into the card reader and then enters his/her identity ID_i and password PW_i. The smart card computes $sCID_i^* = h(ID_i) \oplus h(PW_i||b_i)$ and checks if $CID_i^* = CID_i$. If they are not equal, terminate this session.

L2. If $CID_i^* = CID_i$, the smart card generates two random numbers x and $r_i \ni Z_q^*$ and calculates $R_i = r_i \cdot P$. The smart card then computes $DID_i = Reg_{IDi} \oplus h(PW_i||b_i) \cdot Pub_{RS}$, $QID_i = H(ID_i)$, $C_m = x \cdot H(ID_i)$, $xAuth_i = x \cdot Auth_i$, $Mi = xr_i \cdot QID_i$, $d_{ij} = h(xH(ID_i)||SID_j||M_i||R_i)$ and $B_{ij} = x(r_i + d_{ij}) \cdot DID_i$.

L2. $U_i \rightarrow SS_j : \{xAuth_i, C_m, M_i, B_{ij}, R_i\}$
Finally, the user U_i with the smart card sends the login request $\{xAuth_i, C_m, M_i, B_{ij}, R_i\}$ to the service server SS_j.

4.4 Verification phase

V1. Upon receiving the login request $\{xAuth_i, C_m, M_i, B_{ij}, R_i\}$, the service server SS_j computes $xAuth_i^* = TC_m$ to check if $xAuth_i^* = xAuth_i$. If it is not valid, terminate this session.

V2. If $xAuth_i^* = xAuth_i$, then the service server SS_j computes $d_{ij} = h(C_m||SID_j||M_i||R_i)$. Next, the service server SS_j checks if the following equation can hold:

$$\hat{e} <B_{ij}, P >= \hat{e} < M_i + xd_{ij} \cdot QID_i, pub_{RS}> \tag{4}$$

If it holds, the service server SS_j accepts the login request; otherwise, terminate this session.

V3. $SS_j \rightarrow U_i : \{Auth_{ji}, K_{ji}, R_j\}$
Then the service server SS_j generates a random point $R_j = r_j P$, the temporary key $TK_{ji} = r_j R_i$, the shared secret key $K_{ji} = s_j R_i$and $Auth_{ji} = h(C_m||K_{ji}||R_j)$. Next, the service server SS_j sends $(Auth_{ji}, K_{ji}, R_j)$ to the user U_i.

V4. $U_i \rightarrow SS_j : Auth_{ij}$
After receiving the responses $(Auth_{ji}, K_{ji}, R_j)$, the user U_i computes $Pub_j = h(SID_j||W_j)Pub_{RS} + W_j$ based on SCPK as the public key of the service server SS_j. Then the temporary key $TK_{ij} = r_i R_j$ and the shared secret key $K_{ij} = r_i Pub_j$ are also computed by the user U_i. Next the user U_i checks if the computed $h(xH(ID_i)||K_{ij}||R_j)$ is equal to the received $Auth_{ji}$. If they are equal, the user U_i computes $Auth_{ij} = h(ID_i||K_{ij}||R_i||R_j)$ and sends it to the service server SS_j.

V5. Upon receiving $Auth_{ij}$, the service server SS_j checks if the computed $h(ID_i||K_{ji}||R_i||R_j)$ is equal to the received $Auth_{ij}$. If they are equal, the two parties can compute the session key $SK = h(xH(ID_i)||T_{ij})$.

4.5 Password change phase

If the user U_i wants to change his/her password, the user U_i has to input his/her identity ID_i, original password PW_i and new password PW_{new}. Then the following process will be performed by the U_i and the registration server RS.

P1. $U_i \rightarrow$ the smart card: $\{ID_i, PW_i\}$
The user U_i inserts the smart card into the card reader and then enters his/her identity ID_i and password PW_i. The smart card computes $CID_i^* = h(ID_i) \oplus h(PW_i||b_i)$ and checks if $CID_i^* = CID_i$. If they are not equal, terminate this session.

P2. If they are equal, the smart card generates a random number $n_i \in Z_q^*$ and calculates $N_i = n_i \cdot P$.

P2. $U_i \rightarrow RS : \{H(ID_i), NID_i, N_i\}$

Second, the user U_i computes $DID_i = Reg_{IDi} \oplus h(PW_i||b_i)Pub_{RS}$ and $NID_i = DID_i \oplus n_i Pub_{RS}$. Then the user send a password change request $\{H(ID_i),\ NID_i,\ N_i\}$ to the registration server *RS*.

P3. $RS \rightarrow \quad U_i : \{V_1\}$
Upon receiving the password change request $\{H(ID_i),\ NID_i,\ N_i\}$, the registration server *RS* checks the database to verify the validity of the user's identity $H(ID_i)$. If the validity of the user's identity is confirmed, the registration server *RS* computes $QID_i = H(ID_i)$ and $DID_i = NID_i \oplus s_{RS}N_i$ to obtain DID_i. Then the registration server *RS* checks if $\hat{e} < DID_i,\ P >= \hat{e} < QID_i,\ Pub_{RS} >$. If it holds, the registration server *RS* sends V_1 to U_i by computing $V_1 = h(DID_i||s_{RS}N_i)$.

P4. $U_i \quad \rightarrow RS : \{V_2,\ V_3\}$
After receiving V_1, the user $U_i$compares $V_1$with the computed $h(DID_i||n_i\ Pub_{RS})$ to confirm the legality of the registration server *RS*. If the legality of the registration server *RS* is confirmed, the user $U_i$sends $V_2$and $V_3$to *RS* by computing $HPW_{new} = h(PW_{new}||b_i)$, $V_2 = HPW_{new} \oplus (n_i Pub_{RS})$ and $V_3 = h(DID_i||n_i Pub_{RS}||HPW_{new})$.

P5. $RS \rightarrow \quad U_i : \{V_4\}$
Upon receiving the message $(V_2,\ V_3)$, the registration server *RS* computes $V_2 \oplus (s_{RS}N_i)$to extract HPW_{new}. Then the user U_i compares the received V_3 with the computed $h(DID_i||s_{RS}N_i||HPW_{new})$. If they are equal, the legality of the user $U_i$is confirmed. Finally, the registration server $Reg^{new}_{ID_i} = DID_i \oplus (s_{RS}HPW_{new})$ and sends V_4 to the user U_i by computing $V_4 = Reg^{new}_{ID_i} \oplus (s_{RS}N_i)$.

P6. After receiving V_4, the user U_i computes $V_4 \oplus (n_i Pub_{RS})$ to extract $Reg^{new}_{ID_i}$ and $CID_{new} = h(ID_i) \oplus h(PW_{new}||b_i)$. Finally, the smart card of the user U_i replaces the original Reg_{IDi} and CID_i with $Reg^{new}_{ID_i}$ and CID_{new}, respectively.

5 Security analysis

The proposed protocol is based on Liao et al.'s protocol, however, we add some countermeasures to resist the described attack and improve the efficiency and security of the original protocol. In the following paragraphs, we will explain how the weaknesses are eliminated.

5.1 Resist to the trace attack

The improved protocol no longer sends the identity directly and clearly. Nevertheless, the identity is mingled with a secret and then is encrypted by the service server's public key. For example, the login message $C_m = \{x \cdot H(ID_i) \oplus H(T)\}$ is sent to the service server SS_j. Therefore, each time the service server will receive the different login messages of the same user *Ui*. Moreover, only the service server corresponding to the public key being used to encrypt can decrypt the login messages using the corresponding private key.

5.2 Release from updating ID table

Instead of using ID table to authenticate the registered user, the registration server issues a smart card containing a specific token $H(T)$ to the registered user. Also the service servers registered the registration server will have the same tokens. Therefore, the service server can use the token to identify if the user can access the system. If the token is stolen by an attacker ID_e, he/she still cannot forge a login request to pass the authentication. Since he/she has no knowledge of the system secret key s_{RS}, he/she cannot compute $DID_e = s_{RS} \cdot H(ID_e)$. Hence, he/she cannot compute a correct B_{ej} to pass the authentication.

5.3 Pre-authentication in the smart card

In our improved protocol, the smart card issued by the registration server contains the authentication code $Auth_i$ which is composed of registered user's identity and password. Since there is no way to get any information of the user's identity and password, an attacker has to guess these two unknown values simultaneously. Therefore, it is infeasible to launch an off-line password attack for an attacker. Having an authentication code in the smart card, the user has to pass the pre-authentication before sending a login request to the service server.

6 Performance analysis of proposed scheme

Since the mobile devices have the limited computing capability and energy resources, the proposed protocol should be taken wireless communication, computation cost and communication cost into consideration. Moreover, evaluation focuses on the client rather than the server, since the server is regarded as a powerful device. For convenience to evaluate the computational cost, we define computation notations as follows.

- TG_e: Time to execute a bilinear map operation, $\hat{e} : G_1 \times G_1 \rightarrow G_2$.
- TG_{mul}: Time to execute point scalar multiplication on the group G_1.
- TG_H: Time to execute a map-to-point hash function $H(\cdot)$.
- TG_{add}: Time to execute point addition on the group $G1$.
- T_h: Time to execute a one-way hash function $h(\cdot)$.

Since the time of executing the XOR operation and the modular multiplication operation is relatively less, their computation cost is neglected here. Tables 2 and 3 list the experimental data for related pairing-based operations. The computation cost on

Table 2 The computation cost of the user's smart card

Processor clock speeds	Operation				
	TG_e	TG_{mul}	TG_H	TG_{add}	T_h
36 MHz	0.38 s	0.13 s	$<$0.1 s	$<$0.01 s	$<$0.001 s

Table 3 The computation cost of the server

Processor clock speeds	Operation				
	TG_e	TG_{mul}	TG_H	TG_{add}	T_h
3 GHz	3.16 ms	1.17 ms	<1 ms	<0.1 ms	<0.01 ms

Table 4 Cryptanalysis of multi-server related protocols

Functionalities	Liao et al. [28]	Hsiang and Shih [25]	Liao and Wang [24]	Geng and Zhang [17]	Proposed scheme
User's anonymity	No	Yes	Yes	No	Yes
Single registration	Yes	Yes	Yes	Yes	Yes
Mutual authentication	Yes	Yes	Yes	Yes	Yes
Session key agreement	Yes	Yes	Yes	Yes	Yes
Two factor security	Yes	No	No	No	Yes
Correct password update	Yes	No	Yes	Yes	Yes
Malicious user attack resistance	Yes	No	No	No	Yes
Malicious server attack resistance	Yes	Yes	No	No	Yes
Perfect forward secrecy	Yes	No	No	Yes	Yes
Pre-authentication	No	Yes	Yes	No	Yes

the user side is presented as Table 2. In [32], the processor on the user side (i.e., smart card) has the maximum clock speeds of 36 MHz and equips a five-stage pipeline 2 KB instruction cache, 256 KB flash memory and 16 KB RAM.

The computation cost on the server side is presented as Table 3. The processor of the server has the maximum clock speed of 3 GHz with 512 MB RAM. The operating system runs Windows XP.

We focus on the computation cost of the login and authentication phase. The proposed protocol needs $7TG_{mul} + TG_H + TG_{add} + 7T_h$ online computations on the user side, and requires $2TG_e + 5TG_{mul} + TG_{add} + 2T_h$ computations on the server side. The computation time of the login and authentication phase of the user is 1.027 seconds which is almost close to the time needed by Liao et al.'s protocol. However, the improved protocol can resist to the trace attack which will invade the user's privacy seriously. Besides, it removes the burden of updating ID tables of the service server. The cryptanalysis of multi-server related protocols is presented as Table 4. The performance analysis in this section is based on software; software approaches could be a good choice since they have low cost and require a short development time. The low values of performance are a forbidden factor for possible software implementation. On the other hand, hardware alternatives could be selected for implementing crypto-processors architectures. Both Application Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs) can support high data rates, although such designs are more time consuming and expensive compared with the software alternative. The details can be referred in [33].

7 Conclusion

In recent years, many studies try to propose a secure and efficient remote user authentication scheme. Nevertheless, most of the proposed schemes were found to be insecure, and the improved scheme was proposed later. After studying the recent improved schemes, we proposed the more secure and efficient scheme in this paper. Our proposed scheme can satisfy all the security features needed for achieving secure password authentication in multi-server environments, as compared with the previously proposed schemes. We present a cryptanalysis of a recently proposed Liao et al.'s scheme and showed that their scheme is vulnerable to the trace attack and inefficient in maintaining an ID table. We have specified and analyzed the proposed anonymous remote user authentication protocol for multi-server architecture using smart cards which is very effective to thwart various attacks. In addition, in comparison with the previously proposed schemes, our improved scheme uses nearly the same operations in its implementation. Security and performance analysis proveS that the proposed scheme is more secure and practical. In the future work, we will implement the proposed protocol using a secure combination of a smart phone and a micro SD card.

References

1. Sklavos N, Zhang X (2007) Wireless security and cryptography: specifications and implementations. CRC-Press, A Taylor and Francis Group, ISBN: 084938771X
2. Lamport L (1981) Password authentication with insecure communication. Commun ACM 24(11):770–772
3. Hwang M-S, Li L-H (2000) A new remote user authentication scheme using smart cards. IEEE Trans Consum Electron 46(1):28–30
4. ElGamal T (1985) A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inf Theory 32(4):469–72
5. Hwang T, Ku WC (1995) Reparable key distribution protocols for Internet environments. IEEE Trans Consum Electron 43(5):1947–1949
6. Sun HM (2000) An efficient remote user authentication scheme using smart cards. IEEE Trans Consum Electron 46(4):958–961
7. Shen JJ, Lin CW, Hwang MS (2003) A modified remote user authentication scheme using smart cards. IEEE Trans Consum Electron 49(2):414–416
8. Amit K, Awashti S (2004) An enhanced remote user authentication scheme using smart cards. IEEE Trans Consum Electron 50(2):583–586
9. Chang C, Hwang KF (2003) Some forgery attacks on a remote user authentication scheme using smart cards. Informatics 14(3):289–294
10. Das ML, Saxena A, Gulati VP (2004) A dynamic ID-based remote user authentication scheme. IEEE Trans Consum Electron 50(2):629–631
11. Ku WC, Chang ST (2005) Impersonation attack on a dynamic ID-based remote user authentication scheme using smart cards. IEICE Trans Commun 5:2165–2167
12. Hwang MS, Lee CC, Tang YL (2002) A simple remote user authentication scheme. Math Comput Model 36(1–2):103–107
13. Lee WB, Chang CC (2000) User identification and key distribution maintaining anonymity for distributed computer network. Comput Syst Sci 15(4):211–214
14. Tsuar WJ, Wu CC, Lee WB (2001) A flexible user authentication for multi-server internet services. Networking-JCN2001LNCS, vol. 2093, Springer, Berlin, pp 174–183
15. Li L, Lin I, Hwang M (2001) A remote password authentication scheme for multi-server architecture using neural networks. IEEE Trans Neural Netw 12(6):1498–1504
16. Lin C, Hwang MS, Li LH (2003) A new remote user authentication scheme for multiserver architecture. Future Gener Comput Syst 1(19):13–22

17. Tsuar WJ (2005) An enhanced user authentication scheme for multi-server internet services. Appl Math Comput 170:258–266
18. Wu TS, Hsu CL (2004) Efficient user identification scheme with key distribution preserving anonymity for distributed computer networks. Comput Secur 23:120–125
19. Yang Y, Wang S, Bao F, Wang J, Deng R (2004) New efficient user identification and key distribution scheme providing enhanced security. Comput Secur 23(8):697–704
20. Juang WS (2004) Efficient multi-server password authenticated key agreement using smart cards. IEEE Trans Consum Electron 50(1):251–255
21. Chang C, Lee JS (2004) An efficient and secure multi-server password authentication scheme using smart cards. In: IEEE proceeding of the international conference on cyberworlds
22. Tsai J (2008) Efficient multi-server authentication scheme based on one-way hash function without verification table. Comput Secur 27(4):115–121
23. Liao Y-P, Wang S-S (2009) A secure dynamic ID based remote user authentication scheme for multi-server environment. Comput Stand Interf 31(1):24–29
24. Hsiang H-C, Shih W-K (2009) Improvement of the secure dynamic ID based remote user authentication scheme for rmulti-server environment. Comput Stand Interf 31(6):1118–1123
25. Sood S-K, Sarje A-K, Singh K (2011) A secure dynamic identity based authentication protocol for multi-server architecture. J Netw Comput Appl 34(2):609–618
26. Li Xiong, Xiong Yongping, Ma Jian, Wang Wendong (2012) An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards. J Netw Comput Appl 35(2):763–769
27. Yi-Pin L, Chih-Ming H (2012) A novel multi-server remote user authentication scheme using self-certified public keys for mobile clients. Future Gener Comput Syst. Available online 11 April 2012, ISSN 0167–739X. doi:10.1016/j.future.2012.03.017
28. Girault M (1991) Self-certified public keys. In: Advances in cryptology, Eurocrypt'91. Springer, Berlin, pp 491–497
29. Petersen H, Horster P (1997) Self-certified keys concepts and applications. In: Proceedings of the 3rd conference of communications and multimedia security, Athens, September, pp 22–23
30. Miller V (2004) The Weil pairing and its efficient calculation. J Cryptol 17:235–261
31. Daojing H, Maode M, Yan Z, Chun C, Jiajun B (2011) A strong user authentication scheme with smart cards for wireless communications, Comput Commun, vol 34, Issue 3, pp 367–374, 15 March 2011
32. Scott M, Costigan N, Abdulwahab W (2006) Implementing cryptographic pairings on smartcards. In: Cryptographic hardware and embedded systems—CHES 2006, LNCS, vol 4249. Springer, Berlin, pp 134–147
33. Sklavos N (2010) On the hardware implementation cost of crypto-processors architectures. information systems security. Off J (ISC)2. A Taylor & Francis Group Publication 19(2):53–60