A Lightweight Three-Party Privacy-preserving Authentication Key Exchange Protocol Using Smart Card

Xiaowei Li[1], Yuqing Zhang[1,2], Xuefeng Liu[1] and Jin Cao[1]
[1] State Key Laboratory of Integrated Services Networks, Xidian University
Shaanxi, Xi'an - China
[e-mail: lixw@nipc.org.cn]
[2] National Computer Network Intrusion Protection Center, University of the Chinese Academy of Sciences
Beijing - China
[e-mai :zhangyq@gucas.ac.cn]
*Corresponding author: Yuqing Zhang

Received December 3, 2012; revised April 3, 2013; accepted April 13, 2013; published May 31, 2013

Abstract

How to make people keep both the confidentiality of the sensitive data and the privacy of their real identity in communication networks has been a hot topic in recent years. Researchers proposed privacy-preserving authenticated key exchange protocols (PPAKE) to answer this question. However, lots of PPAKE protocols need users to remember long secrets which are inconvenient for them. In this paper we propose a lightweight three-party privacy-preserving authentication key exchange (3PPAKE) protocol using smart card to address the problem. The advantages of the new 3PPAKE protocol are: 1. The only secrets that the users need to remember in the authentication are their short passwords; 2. Both of the users can negotiate a common key and keep their identity privacy, i.e., providing anonymity for both users in the communication; 3. It enjoys better performance in terms of computation cost and security. The security of the scheme is given in the random oracle model. To the best of our knowledge, the new protocol is the first provably secure authentication protocol which provides anonymity for both users in the three-party setting.

Keywords: Authenticated key exchange, privacy-preserving, three-party protocol, smart card

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant (60970140) and Beijing Natural Science Foundation (4122089). The authors would like to thank the editor and all the anonymous reviewers for their useful suggestion..

http://dx.doi.org/10.3837/tiis.2013.05.021

1. Introduction

People focus on two things when communicating in the open networks. One thing is the sensitive data they sent. They do not want unauthorized people to read their secret. Authenticated key exchange (AKE) protocol plays an important role in modern communications to protect the sensitive data of the communication parties. Lots of AKE protocols which have strong security and good performance have been proposed in recent years, such as Internet Key Exchange (IKE) [1] and MQV [2], etc. The other thing that people focus on is their privacy. People pay more and more attention to the identity privacy in recent years. When communicating with others they do not want to leak their real identity to unauthorized people. In Global Mobility Network (GLOMONET) the mobile users do not want to leak their roaming line either since someone may locate them if the roaming line is leaked. Lots of AKE protocols which can provide privacy of the communication parties were proposed. The AKE protocols with such property are often called privacy-preserving authenticated key exchange (PPAKE) protocols. Aiello et al. [3] proposed two PPAKE protocols which can protect the privacy of the initiator and the responder respectively. They called the schemes JFKi and JFKr. Cheng et al. [4] proposed an efficient two-party AKE protocol with unilateral identity privacy in ID-based cryptosystem. Meanwhile a formal security analysis was given in a modified Bellare-Rogaway security model [5]. Considering the identity privacy in global mobility networks, Lee et al. [6] proposed an authentication scheme with anonymity.

Although these PPAKE protocols were proposed most of them need the user to remember a long private key. It is inconvenient for the user. So researchers use a human-memorable password to accomplish the authentication and key exchange which is called password authenticated key exchange (PAKE). Lots of PAKE protocols have been proposed [7-9] based on the fundamental work of Bellovin-Merritt [10]. However, there are two limitations in PAKE protocols. One is that people prefer to remember few passwords to complete communication but not many. When it comes to the scenario where a user wants to communicate with many other users, the passwords that the user should remember will be linear in the number of the communication partners. The other is how can privacy of the communication parties be protected in PAKE protocols. Lots of researchers were committed to slove these two limitations [11-21], however, these works either conider one of the problems [11-16] or need complicated computation cost [17].

In order to protect both the confidential data and privacy of the communication parties in the open networks, in this paper, we propose a lightweight three-party privacy-preserving authentication key exchange (3PPAKE) protocol using smart card. The advantage of the proposed protocol is that it cannot only protect the privacy of the initiator but also the responder. The security of the 3PPAKE protocol is given in the random oracle model. The high performance in communication cost and computation cost is shown compared with the related schemes. The trick used in the scheme may provide a new way in designing privacy-preserving authenticated key exchange protocols.

In Section 2, we review the previous work in solving the limitations mentioned above. In Section 3, we present a security model for the 3PPAKE protocol. In Section 4, we describe the 3PPAKE protocol we proposed. We then give the security analysis and the performance of our scheme in Section 5. Finally, in Section 6 we make a conclusion of this paper.

2. Related Work

Researchers pay much attention to address the two problems in PAKE protocols, i.e., reducing the number of the passwords that people need to remember and providing privacy-preserving property in PAKE protocols. In order to address the first problem, three-party password-based authenticated key exchange (3PAKE) protocols have been proposed. In 3PAKE protocols, each user only needs to share a password with a trusted server then two users can complete the communication with the help of the server. Lots of 3PAKE protocols have been proposed due to their convenience for the users [11-13]. The first formal treatment of 3PAKE protocols was provided by Abdalla et al. [13]. In [13], Abdalla et al. proposed a generic three-party password-based protocol. Actually, it works as a compiler which can transform any secure two-party password-based protocol into a secure three-party password-based protocol. However, Wang et al. [14] found Abdalla et al.'s scheme vulnerable to the undetectable on-line dictionary attack since the server did not authenticate the user during the protocol execution. Wang et al. [14] added a confirming message on the scheme of [13] to provide an explicit authentication between the server and the user and proposed a new generic 3PAKE protocol.

Although 3PAKE protocols can address the first problem proposed above, it is nontrivial to address the second problem in password only scenario, i.e., providing privacy-preserving property. Meanwhile, there is also a drawback of 3PAKE protocols mentioned above that all the users' passwords must be stored in the server. So once the server is corrupted then all the users' passwords will be leaked. In addition, users often use the same password in multiple applications so if the password is leaked from one application the other applications will be corrupted either. It is very dangerous for the users. In such case, a human memorable password combined with a smart card is used in authentication and key exchange protocols to achieve secure communication. The password and the smart card mechanism can not only solve the problem of storing the password in the server but also provide privacy-preserving property. Juang et al. [15] proposed a robust and efficient anonymous two-party password authenticated key agreement scheme using smart card. It possesses many properties compared with the PAKE protocols using smart card, such as identity protection, low computation for smart card, no password table and it is secure even if the smart card was lost. However, Sun et al. [16] found the authentication scheme in [15] was vulnerable to the denial-of-service attack. An improvement of [16] was proposed by Sun et al. [16]. Recently, Li et al. [17] found the improved scheme in [16] was vulnerable to dictionary attack and key compromise impersonation attack when the smart card was lost. An effcient PAKE protocol was also proposed in [17]. Considering the privacy-preserving property in three-party PAKE protocol, Juang et al. [18] proposed the first 3PAKE protocol using smart card which can provide the privacy of both communication parties. It addresses the problem that how can make a PAKE protocol provide the security for the sensive data while achieving the privacy for both of the users in three-party setting. The 3PPAKE protocol in [18] enjoys all the security properties of smart card authentication. However, Juang et al. did not give a formal security proof of their PPAKE protocol and the pairing operations were used in [18] which leaded the computation cost a bit high. Recently, Lee et al. [19] proposed a three-party password-based authenticated key exchange protocol with user anonymity using extended chaotic maps which gave a new way to provide both the confidentiality of the sensitive data and the privacy of the users.

3. Security Model

Before introducing the security model, we need to introduce the *dictionary attack* in password authentcation first.

3.1 Dictionary Attack

The dictionary attack which is also called the password guessing attack is a special attack to password authentication key exchange protocol. The adversary can guess the user's password since the password space is small. It is usually divided into two kinds of dictionary attack.

- Online dictionary attack. The adversary attempts to impersonate the user to guess the user's password in an online interaction with the server. The correctness of his/her guess can be verified using the response from the server. A failed online password guess will be detected by the server or the user. This attack is inevitable since one can always guess other person's password even if the protocol is well designed. One solution to solve this problem is to limit the number of the password re-entered.
- Offline dictionary attack. The adversary can verify the correctness of his/her password guess offline by the transcript of the protocol. This attack is a great threat to the password authenticated key exchange protocols.

3.2 Formal Model

The security model is based on BPR model proposed by Bellare et al. [23]. Each participant in the protocol *P* is either a client *U* or the trusted server *S*. Two users $A, B \in U$ will authenticate each other and agree on a session key with the help of *S*. During the execution of *P*, each of *U* and *S* may have many *instances* and one time of the protocol execution is called an instance. We also call instance *i* of *U* as an *oracle*, and denote it $\prod_U^i$. At the beginning of the protocol each user shares a password *PW* with the server and holds a smart card. The password is chosen from a small dictionary *Password* whose distribution is D_{PW}. We assume there is an adversary $\mathcal{A}$ that has complete control over the network, and tries to break the privacy of the session key. $\mathcal{A}$ can get access to the oracles and interact with them via the following queries:

Execute ($\prod_A^i, \prod_B^j, \prod_S^k$): This query models passive attacks. The adversary $\mathcal{A}$ gets access to honest executions between the instances $\prod_A^i, \prod_B^j$ and $\prod_S^k$ by eavesdropping.

Send ($\prod_U^i / \prod_S^k, m$): This query models active attacks. Upon receiving the message *m*, oracle $\prod_U^i / \prod_S^k$ executes the protocol and responds with an outgoing message or a decision to indicate accepting or rejecting the session. If $\prod_U^i / \prod_S^k$ does not exist, it will be created as an initiator, or as a responder otherwise.

Reveal ($\prod_U^i$): If the oracle has not accepted, it returns $\perp$; otherwise, it reveals the session key as the answer.

Corrupt (*U*, *password*): Returns the password of *U* to $\mathcal{A}$.

Corrupt (*U*, *smart card*): Returns the secret information stored in *U*'s smart card to $\mathcal{A}$.

Test ($\prod_A^i / \prod_B^j$): When receiving this query, a *fresh* oracle(define later) $\prod_A^i / \prod_B^j$, as a challenger, randomly chooses b and responds to this query with the session key, if $b=1$, or a random value from the distribution of the session key if $b=0$. The *Test* query can only be asked once.

When the adversary receives the answer of the *Test* query from the challenger, he/she can continue querying the oracles except that it cannot reveal the test oracle $\prod_A^i / \prod_B^j$ or its partner $\prod_B^j / \prod_A^i$ (we define the partner of $\prod_A^i / \prod_B^j$ as the oracle which has matching conversations to $\prod_A^i / \prod_B^j$ and we assume $\prod_A^i$ and $\prod_B^j$ are partners to each other). Finally the adversary outputs a guess b'. If $b' = b$, we say that the adversary wins. We use the session ID which can be the concatenation of the messages in a session to define matching conversations, i.e., two oracles $\prod_A^i$ and $\prod_B^j$ have matching conversations to each other if they have the same session ID.

Definition 1 Fresh oracle An oracle $\prod_U^i$ is fresh if (1) $\prod_U^i$ has accepted; (2) $\prod_U^i$ is not being asked by the Reveal query; (3) there is no oracle $\prod_{U'}^j$, which has had a matching conversation to $\prod_U^i$, being asked by the Reveal query;(4) if *Corrupt* (*U*, *password*) is asked to *U* then *Corrupt* (*U*, *smart card*) is not allowed to be asked to *U* and vice versa.

Definition 2 AKE-Security A password-based authenticated key exchange protocol is said to be AKE secure if for any polynomial time adversary $\mathcal{A}$ the following equation holds:

$$Adv_P^{ake}(\mathcal{A}) = 2\left|\Pr[b' = b]\right| - 1 = \frac{O(q_s)}{N} + neg(k) \quad (1)$$

Where q_s is the number of the *Send* query, N is the the size of the password dictionary and $neg(l)$ is a negligible value.

4. A Lightweight Three-party Privacy-preserving Authentication Key Exchange Protocol Using Smart Card

In this section we give a detailed description of the proposed 3PPAKE protocol. The notations used in the protocol are shown in **Table 1**. Note that the hash functions used in this paper are a kind of functions other than a specific hash function. The protocol has three phases: the Registration Phase, the Authentication Phase and the Password-changing Phase.

Table 1. Notations

Notation	Description
S	the trusted server
A, B	the users
ID_A, ID_B	the identity of *A* and *B*
$E(F_p)$	an elliptic curve over a finite field F_p
P	a generator of a subgroup of F_p
PW_A, PW_B	*A*'s and *B*' human-memorizable password respectively
$H(\cdot)$	a kind of cryptographically secure hash functions from $(0,1)^* \rightarrow \{0,1\}^p$
u	the master secret key of the server
s, sP	the private and public key pair of the server
sk	the session key
$\|\|$	the concatenation operator
$\oplus$	the bitwise exclusive-OR operator

4.1 Registration Phase

When a user *U* wants to register in a server *S*, *U* and *S* will perform the following three steps. Then, a smart card generated from *S* is issued to *U*.

Step 1. *U* chooses his/her password PW_U and a random value b from Z_p^* and computes $H(PW_U \| b)$. Then, *U* sends his/her identifier ID_U and $H(PW_U \| b)$ to *S* in a secure channel.

Step 2. After receiving the message from the user, *S* first checks whether the identifier is valid. If it is not valid, *S* requests the user to send the registration message again. Otherwise, *S* chooses a random value r_U from Z_p^* and computes $V_U = H(u \| ID_U \| r_U) \oplus H(PW_U \| b)$ using its master secret key u. Then, *S* issues a smart card containing $\{ID_U, V_U\}$ to *U* by a secure channel. *S* stores $\{ID_U, r_U\}$ in its data center.

Step 3. After receiving the message from the server, *U* imbeds the random value b into the smart card. Now the smart card contains $\{ID_U, b, V_U\}$.

4.2 Authentication Phase

If both the user *A* and the user *B* have registered in *S*, then *A* and *B* can achieve mutual authentication and agree on a session key with the help of *S* in an anonymous way. Suppose *A* has a password PW_A and a smart card which contains $\{ID_A, b_A, V_A\}$ and *B* has a password PW_B and a smart card which contains $\{ID_B, b_B, V_B\}$. The process of the authentication is shown in **Fig. 1**.

Step 1. *A* inserts his/her smart card into a card reader and inputs his/her password PW_A. The smart card firstly computes $H(PW_A \| b_A)$ and $W_A = V_A \oplus H(PW_A \| b_A)$. Then, the smart card randomly selects three integers a, r_{A1} and r_{A2} from Z_p^* and computes $T_A = aP$ and $K_1 = H(a \cdot sP \| T_A \| sP)$ where sP is the server's public key. After that, the smart card continues to generate pseudo-IDs for *A* and *B*, i.e., $PID_A = H(K_1 \| r_{A1}) \oplus ID_A$ and $PID_B = H(K_1 \| r_{A2}) \oplus ID_B$. The authentication message of *A* is generated by $Auth_A = H(K_1 \| W_A)$. Then, *A* sends $\{T_A, r_{A1}, r_{A2}, PID_A, PID_B, Auth_A\}$ to the server *S*. On the other side, *B* proceeds the similar steps. *B* inserts his/her smart card into a card reader and inputs his/her password PW_B . The smart card firstly computes $H(PW_B \| b_B)$ and $W_B = V_B \oplus H(PW_B \| b_B)$. Then, the smart card randomly selects three integers b, r_{B1} and r_{B2} from Z_p^* and computes $T_b = bP$ and $K_2 = H(b \cdot sP \| T_B \| sP)$. After that, the smart card continues to generate pseudo-IDs for *B* and *A*, i.e., $PID_B = H(K_2 \| r_{B1}) \oplus ID_B$ and $PID_A = H(K_2 \| r_{B2}) \oplus ID_A$. The authentication message of *B* is generated by $Auth_B = H(K_2 \| W_B)$. Then, *B* sends $\{T_B, r_{B1}, r_{B2}, PID_B, PID_A, Auth_B\}$ to the server *S*.

Step 2. After receiving the messages from *A* and *B*, *S* first computes $K_1 = H(s \cdot aP \| T_A \| sP)$ and $K_2 = H(s \cdot bP \| T_B \| sP)$ with *S*'s master secret key s. Then, on the one side *S* reveals the identities of both parties by computing $ID_A = H(K_1 \| r_{A1}) \oplus PID_A$ and $ID_B = H(K_1 \| r_{A2}) \oplus PID_B$ from the messages sent by *A* and checks whether the authentication message $Auth_A = H(K_1 \| W_A)$ holds. If it does not hold, *S* requests the user to send a new one. On the other side *S* reveals the identities of both parties by computing $ID_B = H(K_2 \| r_{B1}) \oplus PID_B$ and $ID_A = H(K_2 \| r_{B2}) \oplus PID_A$ from the messages sent by *B* and checks whether the authentication message $Auth_B = H(K_2 \| W_B)$ holds. If it does not hold, *S* also requests the user to send a new one. If both of the authentication messages are correct, then *S* checks whether the identities are matched between *A* and *B*. If they are matched *S* computes $Auth_{SA} = H(K_1 \| ID_A \| ID_B \| T_A \| T_B)$ and $Auth_{SB} = H(K_2 \| ID_B \| ID_A \| T_B \| T_A)$. Then, *S* sends $\{T_A, T_B, Auth_{SA}\}$ to *A* and to *B* respectively.

Step 3. On one side when receiving the message $\{T_A, Auth_{SA}\}$, *A*'s smart card verifies the correctness of $Auth_{SA}$. If it is not equal to $H(K_1 \| ID_A \| ID_B \| T_A \| T_B)$, the smart card refuses it. Otherwise, the smart card computes the session key $sk_{AB} = H(a \cdot T_B \| ID_A \| ID_B \| T_A \| T_B)$ and $\sigma_A = H(sk_{AB} \| ID_A \| ID_B \| T_A \| T_B)$ as the comfirmation. Then, *A* sends σ_A to *B*. On the other side when receiving the message $\{T_B, Auth_{SB}\}$, *B*'s smart card proceeds the similar steps. *B*'s smart card verifies the correctness of $Auth_{SB}$. If it is not equal to $H(K_2 \| ID_B \| ID_A \| T_B \| T_A)$, the smart card refuses it. Otherwise, the smart card computes the session key $sk_{BA} = H(b \cdot T_A \| ID_A \| ID_B \| T_A \| T_B)$ and the confirmation $\sigma_B = H(sk_{BA} \| ID_B \| ID_A \| T_B \| T_A)$. Then, *B* sends σ_B to *A*.

Step 4. *A* and *B* check whether the equations $\sigma_B = H(sk_{BA} \| ID_B \| ID_A \| T_B \| T_A)$ and $\sigma_A = H(sk_{AB} \| ID_A \| ID_B \| T_A \| T_B)$ hold rsespectively. If they hold, *A* and *B* can assure that the communication peer has computed the correct session key. Otherwise, they require the communication peer send the confirmation again.

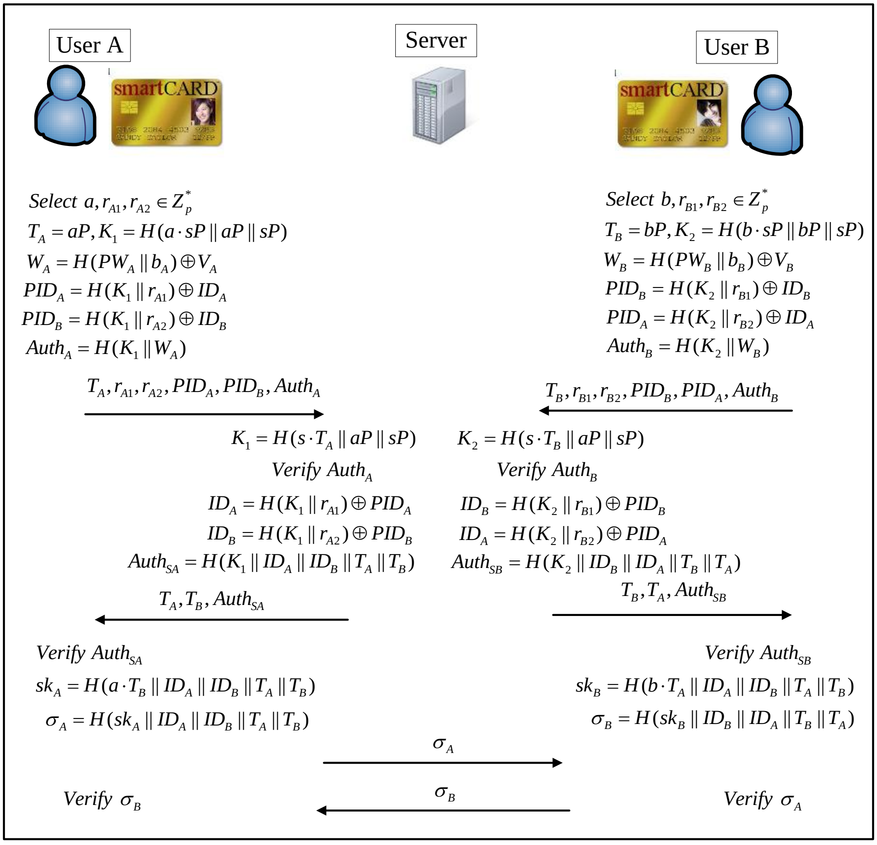

User A
Server
User B
smartCARD
smartCARD
Select $a, r_{A1}, r_{A2} \in Z_p^*$
$T_A = aP, K_1 = H(a \cdot sP \| aP \| sP)$
$W_A = H(PW_A \| b_A) \oplus V_A$
$PID_A = H(K_1 \| r_{A1}) \oplus ID_A$
$PID_B = H(K_1 \| r_{A2}) \oplus ID_B$
$Auth_A = H(K_1 \| W_A)$
Select $b, r_{B1}, r_{B2} \in Z_p^*$
$T_B = bP, K_2 = H(b \cdot sP \| bP \| sP)$
$W_B = H(PW_B \| b_B) \oplus V_B$
$PID_B = H(K_2 \| r_{B1}) \oplus ID_B$
$PID_A = H(K_2 \| r_{B2}) \oplus ID_A$
$Auth_B = H(K_2 \| W_B)$
$T_A, r_{A1}, r_{A2}, PID_A, PID_B, Auth_A$
$T_B, r_{B1}, r_{B2}, PID_B, PID_A, Auth_B$
$K_1 = H(s \cdot T_A \| aP \| sP)$
Verify $Auth_A$
$ID_A = H(K_1 \| r_{A1}) \oplus PID_A$
$ID_B = H(K_1 \| r_{A2}) \oplus PID_B$
$Auth_{SA} = H(K_1 \| ID_A \| ID_B \| T_A \| T_B)$
$K_2 = H(s \cdot T_B \| aP \| sP)$
Verify $Auth_B$
$ID_B = H(K_2 \| r_{B1}) \oplus PID_B$
$ID_A = H(K_2 \| r_{B2}) \oplus PID_A$
$Auth_{SB} = H(K_2 \| ID_B \| ID_A \| T_B \| T_A)$
$T_A, T_B, Auth_{SA}$
$T_B, T_A, Auth_{SB}$
Verify $Auth_{SA}$
$sk_A = H(a \cdot T_B \| ID_A \| ID_B \| T_A \| T_B)$
$\sigma_A = H(sk_A \| ID_A \| ID_B \| T_A \| T_B)$
Verify $Auth_{SB}$
$sk_B = H(b \cdot T_A \| ID_A \| ID_B \| T_A \| T_B)$
$\sigma_B = H(sk_B \| ID_B \| ID_A \| T_B \| T_A)$
σ_A
σ_B
Verify σ_B
Verify σ_A

Fig. 1. The authentication phase of the proposed protocol

4.3 Password-changing Phase

When a user U wants to changes his/her password, U first inserts the smart card into a card reader, then keys the old password PW_{old} and a new password PW_{new} . U's smart card computes $V_{new} = V_{old} \oplus H(b \| PW_{old}) \oplus H(b \| PW_{new})$ and then it updates V_{old} by V_{new} .

5. Security Analysis and Performance

5.1 Security Analysis

We analyze the security of the proposed protocol in the random oracle model, i.e., suppose the Hash function is a random oracle. Some computational assumptions need to be given before starting the proof.

Elliptic Curve Computational Diffie-Hellman (ECDH) Assumption: Let $E(F_p)$ denote an elliptic curve over the prime finite field F_p. Let P be a generator on $E(F_p)$ and aP and bP be two elements of $E(F_p)$. Given P, aP, bP, cP, no efficient algorithm can compute abP with non-negligible probability where $a, b \in Z_p^*$.

Elliptic Curve Decisional Diffie-Hellman (ECDDH) Assumption: Let aP, bP and cP be three elements of $E(F_p)$. Given P, aP, bP, cP, no efficient algorithm can decide whether $cP = abP$ with non-negligible probability where $a, b, c \in Z_p^*$.

Elliptic Curve Gap Diffie-Hellman (Gap-ECDH) Assumption: Let aP and bP be two elements of $E(F_p)$. Given P, aP, bP and an ECDDH oracle that correctly solves the decisional Diffie-Hellman problem, no efficient algorithm can compute abP with non-negligible probability where $a, b \in Z_p^*$.

Theorem 1 (AKE-Security) *Let $E(F_p)$ denote an elliptic curve over the prime finite field F_p and G is the group in $E(F_p)$. Let D_{PW} be the distribution of the password and N be the size of the distribution. Let 3PPAKE be the protocol we proposed described in Fig.1. Let $\mathcal{A}$ be an adversary against the AKE security within a time bound t, making less than q_s Send queries, q_e Execute queries and less than q_H random oracle queries, we have:*

$$Adv_{3PPAKE}^{ake}(\mathcal{A}) < \frac{(q_e + q_s)^2}{p} + \frac{{q_H}^2}{p} + \frac{2q_s}{N} + (6q_e q_H + 4q_s q_H) Adv_G^{Gap-ECDH}(t) \tag{2}$$

Proof. For an easier analysis in the proof, we first exclude the events that some collisions appear on the transcripts $\{T_A, r_{A1}, r_{A2}, PID_A, PID_B, Auth_A\}$, $\{T_B, r_{B1}, r_{B2}, PID_B, PID_A, Auth_B\}$ and some collisions appear on Hash functions. We also exclude the event that $\mathcal{A}$ luckily guesses bit b of the test session without interacting with the 3PPAKE protocol. Let *excluded* denote these events, then the probabilities of these events are bounded by:

$$\Pr[excluded] \le \frac{(q_e + q_s)^2}{2p} + \frac{{q_H}^2}{2p} + \frac{1}{2} \tag{3}$$

Now we can analyze the security of the protocol. Let $\mathcal{S}$ be the event that $\mathcal{A}$ breaks the AKE security of the 3PPAKE protocol. Let Pr[$\mathcal{S}$] be the probability of this event. Then we can further divide the event $\mathcal{S}$ into two independent events. Let $\mathcal{S}_1$ be the event that $\mathcal{A}$ breaks the AKE security of the 3PPAKE protocol by breaking the password security. Let $\mathcal{S}_2$ be the event that $\mathcal{A}$ breaks the AKE security of the 3PPAKE protocol without breaking the password security. Pr[$\mathcal{S}_1$] and Pr[$\mathcal{S}2$] are probabilities of event $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively. Then, we have:

$$\Pr[\mathcal{S}] = \Pr[\mathcal{S}_1] + \Pr[\mathcal{S}_2] \tag{3}$$

The probability of the event $\mathcal{S}_1$ Actually, event $\mathcal{S}_1$ can also be divided into three parts according to the different ways of breaking the password, i.e., corrupting the password directly, online dictionary attack and offline dictionary attack. We denote the probability of these three cases by $\Pr[\mathcal{S}_{1cor}]$, $\Pr[\mathcal{S}_{1on}]$ and $\Pr[\mathcal{S}_{1off}]$ respectively. Now we have:

$$\Pr[\mathcal{S}_1] = \Pr[\mathcal{S}_{1cor}] + \Pr[\mathcal{S}_{1on}] + \Pr[\mathcal{S}_{1off}] \tag{4}$$

The probability of the event $\mathcal{S}_{1cor}$ From **Fig. 1** we can see, although $\mathcal{A}$ has corrupted password of the user *A* or *B*, he/she cannot get any further information. Since $\mathcal{A}$ does not have the secret value V_A or V_B stored in the smart card, he/she cannot compute the authentication message $Auth_A$ or $Auth_B$. It means the authentication message cannot pass by *S*'s verification. So holding the password of user but no smart card gives no help to the adversary in breaking the AKE security of the 3PPAKE protocol. Note if the adversary has corrupted some user's password then he/she cannot corrupt the user's smart card and vice versa. Then, we have:

$$\Pr[\mathcal{S}_{1cor}] \le \Pr[\mathcal{S}_2] \tag{5}$$

The probability of the event $\mathcal{S}_{1on}$ Since we allow the adversary to obtain the smart card and get all the information in the smart card, when $\mathcal{A}$ gets the smart card he/she can choose a potential password of A or B and uses this password to communicate with the server *S*. If *S* does not refuse $\mathcal{A}$ it means $\mathcal{A}$ correctly guesses the password of *A* or *B*. Then $\mathcal{A}$ can further get the session key between *A* and *B* since the random number a or b is chosen by the adversary himself/herself. Let N be the size of the password space and q_s be the numbers of *Send* queries $\mathcal{A}$ can make, then $\Pr[\mathcal{S}_{1on}]$ can be bounded as follows:

$$\Pr[\mathcal{S}_{1on}] \le \frac{q_s}{N} \tag{6}$$

The probability of the event $\mathcal{S}_{1off}$ If $\mathcal{A}$ can break the AKE security of the 3PPAKE protocol by offline dictionary attack then we can construct an algorithm to break Gap-ECDH assumption. When we get a Gap-ECDH tuple (mP, nP), we substitute the server *S*'s public key sP by nP and simulate the protocol for $\mathcal{A}$. We answer the queries of $\mathcal{A}$ as in the real protocol in the random oracle model. Here we have to note that we do not know the value n of

nP, so when $\mathcal{A}$ asks a send query to S by $\{T_A = aP, r_{A1}, r_{A2}, PID_A, PID_B, Auth_A\}$ we cannot answer it directly. In such case we check whether there is a record $< h, anP, T_A, nP >$ in the hash list (the hash list records the queries to the hash oracle and the corresponding answers) where $h = H(anP \| T_A \| nP)$. If there is the record we can easily answer this query using the record. Otherwise, we will choose a random value h and let $h = H(* \| T_A \| nP)$, i.e., asking the hash oracle by ourselves. Then we store $< h, (*, T_A, nP) >$ in the hash list and answer the corresponding query using this record. From now on when $\mathcal{A}$ asks a hash query by x, T_A, nP, we firstly check whether $x = \mathrm{ECDDH}(T_A, nP)$ by an ECDDH oracle. If the equation does not hold we refuse it. If the equation holds, it means the adversary provides the correct ECDH tuple. In this way, we further check whether there is a record $< h, (anP, T_A, nP) >$ or $< h, (*, T_A, nP) >$ in the hash list. If $< h, (anP, T_A, nP) >$ is in the hash list, it means $\mathcal{A}$ asked this query before and we answer h to $\mathcal{A}$. If $< h, (*, T_A, nP) >$ is in hash list, it means we asked the hash query instead of $\mathcal{A}$. In such case, we answer h to $\mathcal{A}$ and update $*$ with x in the hash list. If there is no record of $< h, (anP, T_A, nP) >$ and $< h, (*, T_A, nP) >$, we choose a random value h' and let $h' = H(x \| T_A \| nP)$. Then, we answer $\mathcal{A}$ by h' and store the $< h', (x, T_A, nP) >$ in the hash list.

Now the simulation is perfect then we can consider the offline dictionary attack. We select *ith Execute* query from $[1, q_s]$ and substitute T_A (or T_B) which is the random value chosen by A(or B) in this session by mP (the Gap-ECDH tuple we chose at the beginning). Suppose $\mathcal{A}$ can guess the password of A(or B) from the *ith Execute* query then we can compute $ECDDH(mP, nP)$ by calling $\mathcal{A}$ as a subroutine. Here we take A's session as an example. $\mathcal{A}$ can get $\{T_A, r_{A1}, r_{A2}, PID_A, PID_B, Auth_A\}$ from an *Execute* query. Then $\mathcal{A}$ continues to corrupt the smart card of A and gets V_A. Now $\mathcal{A}$ can choose PW_A' as a potential password of A and compute $W_A' = V_A \oplus PW_A'$. If $\mathcal{A}$ can guess the password offline then $\mathcal{A}$ must have computed K_1 and verified the equation $Auth_A = H(K_1 \| W_A')$. Since all the queries are answered in the random oracle model, if $\mathcal{A}$ knows K_1 it means he/she has asked a hash query by ($mnP, T_A = mP, nP$), i.e., $\mathcal{A}$ has computed mnP. Then we can use $\mathcal{A}$ to solve the ECDH(mP, nP) problem. Suppose the probability that $\mathcal{A}$ chooses the transcript from the ith *Execute* query to guess the correct password is $1/q_e$ and the probability that we get ECDH value by picking randomly in the hash list is $1/q_h$. Let $Adv_G^{Gap-ECDH}(t)$ be the advantage that $\mathcal{A}$ breaks the Gap-ECDH assumption and t be the running time of $Adv_G^{Gap-ECDH}(t)$. Then we can bound the probability that $\mathcal{A}$ chooses the transcript from the ith *Execute* query and successfully guesses the password of A(or B).

$$\Pr[\mathcal{S}_{1off}] \le q_e q_H Adv_G^{Gap-ECDH}(t) \qquad (7)$$

From the equations (2)-(5) we can conclude:

$$\Pr[\mathcal{S}_1] = \Pr[\mathcal{S}_{1cor}] + \Pr[\mathcal{S}_{1o}] + \Pr[\mathcal{S}_{1off}] \le \Pr[\mathcal{S}_2] + \frac{q_s}{N} + q_e q_H Adv_G^{Gap-ECDH}(t) \qquad (8)$$

The probability of the event $\mathcal{S}_2$ If $\mathcal{A}$ breaks the AKE security of the 3PPAKE protocol without breaking the password then we can construct an algorithm to break Gap-ECDH assumption. Since $\mathcal{A}$ does not guess the real password of *A* or *B*, $\mathcal{A}$ cannot compute the correct authentication message $Auth_A$ or $Auth_B$. So if $\mathcal{A}$ wants to pass the verification of *S* he/she can only ask the *Execute* query or replay the messages that *A* or *B* sents. In such case, we imbed a Gap-ECDH problem Gap-ECDH(mP, nP) into a session to replace T_A and T_B. If $\mathcal{A}$ chooses this session as the test session and gets the session key of this session it means that $\mathcal{A}$ has computed $\text{ECDH}(T_A, T_B)$. Since all the hash oracles are answered in the random oracle model, it means $\mathcal{A}$ must have asked a hash oracle by ($\text{ECDH}(T_A, T_B), ID_A, ID_B, T_A, T_B$). Then we can check the hash list and find the value of $\text{ECDH}(T_A, T_B)$, i.e., mnP. Note here we still need to ask an ECDDH oracle to make the simulation perfect for $\mathcal{A}$. Suppose the probability that $\mathcal{A}$ chooses the right session we chose as the test session is $1/(q_e + q_s)$. Suppose the probability that we get ECDH value by picking randomly in the hash list is $1/q_H$. Then we can bound the probability that $\mathcal{A}$ breaks the AKE security of the 3PPAKE without breaking the password:

$$\Pr[\mathcal{S}_2] \le (q_e + q_s) q_H Adv_G^{Gap-ECDH}(t) \tag{9}$$

From equations (2), (6), (7) we can obtain the probability that $\mathcal{A}$ breaks the AKE security of the 3PPAKE as follows:

$$\begin{aligned} Adv_{3PPAKE}^{ake}(\mathcal{A}) &= 2\left|\Pr[b = b']\right| - 1 \\ &= 2\left|\Pr[excluded + \Pr[\mathcal{S}]]\right| - 1 \\ &= 2\left|\Pr[excluded] + \Pr[\mathcal{S}_1] + \Pr[\mathcal{S}_2]\right| - 1 \\ &\le \frac{(q_e + q_s)^2}{p} + \frac{{q_H}^2}{p} + \frac{2q_s}{N} + (6q_e q_H + 4q_s q_H) Adv_G^{Gap-ECDH}(t) \end{aligned} \tag{10}$$

Theorem 2 (Anonymity). *The proposed 3PPAKE protocol can provide the identity protection for both the initiator and the responder under the Gap-ECDH assumption.*

Proof. The security proof of the anonymity is similar to the proof of the AKE security of the 3PPAKE protocol so we just give a brief explanation. If $\mathcal{A}$ can break the anonymity of the 3PPAKE protocol we can construct an algorithm to solve the Gap-ECDH problem by calling $\mathcal{A}$ as a subroutine. When we get a Gap-ECDH tuple Gap-ECDH(mP, nP), we embed this tuple into the 3PPAKE protocol to replace the value T_A (or T_B) and the public key of the trusted server. From the description of the 3PPAKE protocol we can see if $\mathcal{A}$ can obtain the identity of the communication parties, he/she must know the value $H(K_1 \| r_{A1})$ and $H(K_1 \| r_{A2})$ from *A* or $H(K_2 \| r_{B1})$ and $H(K_2 \| r_{B2})$ from B. It means that $\mathcal{A}$ must have asked the hash query by (K_1, r_{A1}) and (K_1, r_{A2}) or (K_2, r_{B1}) and (K_2, r_{B2}). As the proof in the Theorem 1 we know $\mathcal{A}$ has computed K_1 or K_2 then we can compute mnP by checking the hash list we established. So if the Gap-ECDH problem is hard then the proposed 3PPAKE protocol can provide the identity protection for both users.

5.1 Performance

In this section we show the performance of the proposed 3PPAKE protocol by comparing with some related three-party authenticated key exchange protocols. To the best of our knowledge the 3AKE protocols which can provide the privacy-preserving property for both communication parties are only Juang et al. [18] and ours. We also make a comparison with an efficient password-based 3AKE protocol [12] which does not use the smart card. Table 2 shows the comparison between the proposed 3PPAKE protocol and some related 3AKE protocols in terms of computation costs and some other properties.

In order to present an objective and detailed comparison, we make the computation costs analysis on the basis of the implementation results in [24]. The experiment is implemented on an ellipse curve which is over a finite field with 512 bits prime p and a large prime order q =160 bits. Since the device on the user side is often constrained by the processing speed, the computation cost of the user is evaluated by Philips HiPersmart card. The Philips HiPersmart card provides a 32-bit RISC MIPS-based processor and equips a five-stage pipeline 2 KB instruction cache, 256 KB flash memory and 16 KB RAM, as well as offers a maximum clock speed of 36 MHz. The computational costs on the server side are evaluated using a Pentium IV processor with 512 MB RAM. Both the user and the server sides make use of the publicly available library MIRACL [25]. Table 3 and Table 4 show the experimental data for some related cryptographic operations and related protocols on the Philips HiPersmart card and on the Pentium IV processor, respectively. From Table 2 and Table 4 we can see, compared with Juang et al.'s scheme [18] our scheme has better performance in computation cost and communication cost. The computation cost of users in our scheme is 0.397s while 0.647s in [18]. It reduces about 61% computation cost of [18]. It is attractive for the user since the communication device on the user side is often constrained by processing speed. Compared with Chang et al.'s scheme [12] although a smart card is used in the proposed scheme, it has better performance on computation cost of the server. Meanwhile, [12] cannot provide privacy-preserving property. So based on an overall consideration of efficiency and security, our scheme performs better in terms of communication costs, computational costs as well as security.

Table 2. Comparison between the proposed 3PPAKE protocol and some related 3AKE protocols

Protocol	**Communication costs**		**Communication rounds**	**Privacy preserving**	**Two factor**	**Provable security**
	User	**Server**				
Chang *et al.*'s [12]	$3T_e + 5T_H$	$4T_e + 4T_H$	6	No	No	Yes
Juang *et al.*'s [18]	$1T_p + 2T_m$ $+5T_H + 2T_s$	$1T_p + 2T_m$ $+5T_H + 2T_s$	6	Yes	Yes	No
Ours	$3T_m + 7T_H$	$2T_m + 8T_H$	4	Yes	Yes	Yes

Table 3. Computational costs of related cryptographic operations on different devices

Devices	**Cryptographic operations**				
	T_p	T_e	T_m	T_s	T_H
HiPerSmart™ 36 MHz	0.38s	0.14s	0.13s	<0.001s	<0.001s
Pentium IV 3GHz	3.16ms	1.32ms	1.17ms	<0.01ms	<0.01ms

Table 4. Computational costs of related protocols on different devices

Devices	**Protocols**					
	Chang *et al.*'s [12]		Juang *et al.*'s [18]		Ours	
	User	Server	User	Server	User	Server
HiPerSmart™ 36 MHz	0.425s	-	0.647s	-	0.397s	-

Pentium IV 3GHz	-	5.32ms	-	5.57ms	-	2.42ms

5. Conclusion

In this paper we propose a lightweight 3PPAKE protocol using smart card which can protect both of the communication parties' privacy. The proposed 3PPAKE protocol is provably secure in the random oracle model. The performance comparison shows the advantage of the 3PPAKE protocol. It reduces lots of computation costs on the user side so the new scheme is more suitable for practical applications especially in the computation constrained environment. Our further work will be on the PAKE protocol which provides privacy for the user not only against the attacker but also the server. It means the user can login to the server in an anonymous way.

References

[1] D. Harkins and D. Carreal, "The Internet Key-Exchange (IKE)," *RFC 2409*, 1998. http://www.ietf.org/rfc/rfc2409.txt

[2] L. Law, A. Menezes, M. Qu, J. Solinas and S. Vanstone, "An efficient protocol for authenticated key agreement," *Designs, Codes and Cryptography*, vol. 28, no. 2, pp.119-134, March, 2003. Article (CrossRef Link)

[3] W. Aiello, S. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. Keromytis and O. Reingold, "Just fast keying: key agreement in a hostile Internet," *ACM Transactions on Information and System Security*, vol. 7, no. 2, pp. 1-30, May, 2004. Article (CrossRef Link)

[4] Z. Cheng, L. Chen, R. Comley and Q. Tang, "Identity-based key agreement with unilateral identity privacy using pairings," in *Proc. of Information Security Practice and Experience*, pp.202-213,April 11-14, 2006. Article (CrossRef Link)

[5] M. Bellare, P. Rogaway, Entity authentication and key distribution, in *Proc. of* CRYPTO, pp. 232-249, August 22-26, 1993. Article (CrossRef Link)

[6] C.C. Lee, M.S. Hwang and I.E. Liao, "Security enhancement on a new authentication scheme with anonymity for wireless environments," *IEEE Transactions on Industrial Electronics*, vol. 53, no.5, pp.1683-1687, October, 2006. Article (CrossRef Link)

[7] J. Katz, R. Ostrovsky and M. Yung, "Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords," in *Proc. of EUROCRYPT*, pp.475-494, May 6-10, 2001. Article (CrossRef Link)

[8] E. Bresson, O. Chevassut and D. Pointcheval, "Group Diffie-Hellman Key Exchange Secure against Dictionary Attacks," in *Proc. of ASIACRYPT*, pp.497-514, December 1-5, 2002. Article (CrossRef Link)

[9] M. Abdalla and D. Pointcheval, "A Scalable Password-Based Group Key Exchange Protocol in the Standard Model," in *Proc. of ASIACRYPT*, pp.332-347, December 3-7, 2006. Article (CrossRef Link)

[10] S.M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Proc. of 13th IEEE Symposium on Security and Privacy*, pp.72-84, May 4-6, 1992. Article (CrossRef Link)

[11] R. Lu and Z. Cao, "Simple three-party key exchange protocol," *Computers and Security*, vol.26, no.1, pp. 94-97, February, 2007. Article (CrossRef Link)

[12] T.Y. Chang, M.S. Hwang and W.P. Yang, "A communication-efficient three-party password authenticated key exchange protocol," *Information Sciences*, vol.181, no.1, pp.17-226, January , 2011. Article (CrossRef Link)

[13] M. Abdalla, P.A. Fouque and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. of Public Key Cryptography*, pp. 65-84, January, 2005. Article (CrossRef Link)

[14] W. Wang and L. Hu, "Efficient and provably secure generic construction of three-party password-based authenticated key exchange protocols," in *Proc. of INDOCRYPT*, pp. 118-132, December 11-13, 2006. Article (CrossRef Link)

[15] W.S. Juang, S.T. Chen and H.T. Liaw, "Robust and efficient passwordauthenticated key agreement using smart cards," *IEEE Trans. Ind. Electron*., vol. 55, no. 6, pp. 2551-2556, May 2008. Article (CrossRef Link)

[16] D.Z. Sun, J.P. Huai, J.Z. Sun, J.X. Li, J.W. Zhang and Z.Y. Feng, "Improvements of Juang et al.'s password-authenticated key agreement scheme using smart cards," *IEEE Transaction on Industrial Electronics*, vol 56, no.6, pp. 2284-2291, June, 2009. Article (CrossRef Link)

[17] X. Li and Y. Zhang, "A simple and robust anonymous two-factor authenticated key exchange protocol," *Security and Communication Networks*, published online, http://onlinelibrary.wiley.com/doi/10.1002/sec.605/abstract, August, 2012.

[18] W.S. Juang, C.L. Lei, H.T. Liaw and W.K. Nien, "Robust and efficient three-party user authentication and key agreement using bilinear pairings", *Int. J. Innovative Computing, Information and Control*, vol. 6, no. 2, pp. 763-772, February, 2010. http://www.ijicic.org/08-312-1.pdf

[19] C.C. Lee, C.T. Li and C.W. Hsu, "A Three-party Password-based Authenticated Key Exchange Protocol with User Anonymity using Extended Chaotic Maps," *Nonlinear Dynamics*, Article (CrossRef Link)

[20] C.C. Lee, S.D. Chen and C.L. Chen, "A Computation-Efficient Three-Party Encrypted Key Exchange Protocol," *Applied Mathematics & Information Sciences*, vol. 6, no. 3 pp. 573-579, September, 2012. http://naturalspublishing.com/ArtcIss.asp?ArtcID=710

[21] C.C. Lee, R.X. Chang and H.J. Ko, "Improving Two Novel Three-party Encrypted Key Exchange Protocols with Perfect Forward Secrecy," *International Journal of Foundations of Computer Science*, vol. 21, no. 6, pp. 979-991, December. 2010. Article (CrossRef Link)

[22] C.C. Lee and Y.F. Chang, "On Security of a Practical Three-party Key Exchange Protocol with Round Efficiency," *Information Technology and Control*, vol. 37, no. 4, pp.333-335, December. 2008. http://itc.ktu.lt/itc374/Lee374.pdf

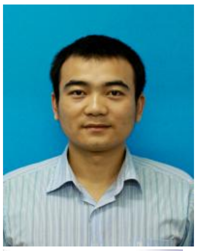
[23] M. Bellare, D. Pointcheval and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Proc. of* EUROCRYPT, pp. 139-155, August 14-18 2000. Article (CrossRef Link)

[24] T.Y. Wu and Y.M. Tseng, "An efficient user authentication and key exchange protocol for mobile client-server environment," *Computer Networks*, vol. 54, no. 9, pp. 1520-1530, June, 2010. Article (CrossRef Link)

[25] Shamus Software, http://certivox.com/solutions/miracl-crypto-sdk/

Xiaowei Li received his B.S. in Department of Applied Mathematics from Xidian University, China, in 2008. He is currently a Ph.D. candidate in Department of Communication Engineering, Xidian University, China. He has joined in State Key Laboratory of Integrated Services Networks, in Xidian University. He has published several papers in International Journals and conferences including Security and Communication Networks and Globecom. His research interests include provable security, network protocol security and wireless network security.

Yuqing Zhang is a professor and supervisor of Ph.D. candidates of Graduate University of Chinese Academy of Sciences. He received his B.S. and M.S. degree in computer science from Xidian University, China, in 1987 and 1990 respectively. He received his Ph.D. degree in Cryptography from Xidian University in 2000. He is a member of IEEE Communications Society and IEICE Transactions on Communications. He has published lots of papers in International Journals and conferences including IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Wireless Communications, IEEE Communications Letters, Globecom, RAID and so on. His research interests include cryptography, information security and network protocol security.



Xuefeng Liu received his B.Sc in information security from Xidian University, China, 2007. He has joined in 2007 for his M.Sc and Ph.D in Xidian University. His research interests include wireless network security, cloud computing, mobile computing and applied cryptography.

Jin Cao received the B.Sc. degree from Xidian University, China, in 2008. He is currently working toward the Ph.D. degree in Cryptography, Xidian University, China. His interests are in wireless network security and LTE networks.