Weaknesses and improvement of three-party authenticated key exchange protocol using elliptic curve cryptography

S. Wu *and Q. Pu

Abstract

Quite recently, Yang et al. presented an efficient three-party authenticated key exchange protocol based upon elliptic curve cryptography for mobile-commerce environments. In this paper, we demonstrate that Yang et al's three-party authenticated protocol is potentially vulnerable to an unknown key-share attack and impersonation attack. Thereafter, we suggest a secure and efficient three-party authenticated key exchange protocol for mobile-commerce environments. Our improved protocol has the following advantages over Yang et al.'s protocol: (1) our scheme combines two factors to strengthen its authentication mechanism; (2) our scheme simply utilizes each user's unique identity to accomplish authentication, eliminating maintenance of a lot of users' keys. Furthermore, our scheme is more efficient than Yang et al's scheme. Therefore, the end result is more suited to be a candidate for implementation in mobile-commerce environments.

Key words: unknown key-share attack; impersonation attack; three-party; authenticated key exchange; mobile-commerce; elliptic curve cryptography.

1 Introduction

Authenticated key exchange(AKE) are protocols for mutual authentication of two parties and generation of a cryptographically strong shared key between them, which are fundamental for achieving secure communication over public, insecure networks. Due to the usefulness of authenticated key exchange protocols, numerous schemes have been proposed to improve security and performance during the last decades. In practices, most of these proposed AKE protocols are presented in the context that the two involved entities are client

* Corresponding author.
Email address: pqwsh@yahoo.com.cn (S. Wu).

and server respectively, e.g. [1–3], which are simply called 2PAKE protocols. However, there is a common problem in these 2PAKE protocols. That is, two communication parties need to previously share a password or a secret for the mutual authentication and a session key agreement. To apply 2PAKE protocols to a large scale peer-to-peer system, each pair of communication parties in a group needs to pre-share a secret. This restriction causes that each user has to keep a large number of secrets for communicating with a group of users. To solve this problem, various three-party authenticated key exchange (3PAKE) protocols were proposed [4–7], in which a trusted server exists to mediate between two communication parties to allow mutual authentication and each user only shares one secret with the server. The design of such protocols remains a hard problem despite years of research, evidenced by the number of protocols (including some well-studied and well-cited ones) being broken after publication[8–16]. As pointed out in [12], experience in the analysis and design of security protocols has shown that even seemingly sound designs may exhibit problems, so years of public scrutiny should still complement the process before a protocol is deemed secure.

In 2008, Chen et al. [17] proposed a round-efficient 3PAKE protocol to provide the computation and communication efficiency for user authentication and session key exchange. However, quite recently, Yang et al. [18]discovered that the computation costs and communication loads of their protocol were still high so that it could not be applied to mobile communications, and thus they proposed an efficient three-party authenticated key exchange protocol based upon elliptic curve cryptography[19,20] for mobile-commerce environments. They claimed that their protocol is superior to similar protocols with respect to security and efficiency. Unfortunately, we find that their protocol is still vulnerable to an unknown key-share attack and impersonation attack. Thereafter, we propose a secure and efficient three-party authenticated key exchange protocol for mobile-commerce environments. Our improved protocol not only defeats the attacks described by us but also has the following advantages over Yang et al.'s protocol:

(1) Firstly, our scheme is a two-factor mutual authentication scheme based on smart cards and passwords so that one must have the smart-card and know the password in order to agree on a session with another user. However, the authentication mechanism in Yang et al.' scheme depends solely on a long-term private key stored in the mobile device(or the card), which is risky because one can easily impersonate the user if he gets the device(this assumption is reasonable because the users may lose his mobile device sometimes).
(2) Secondly, our scheme simply utilizes each user's unique identity to accomplish authentication. Thus, the server does not need to maintain a large public-key table while the number of users becomes very large. Therefore, our scheme provides high scalability for the user addition in

mobile-commerce environment. However, Yang et al.'s scheme on elliptic curve cryptosystem (ECC) accomplished authentication using public-key and thus the server needs a large storage space to store users' public keys and certificates.

Furthermore, our scheme is more efficient than Yang et al's scheme in [18]. Therefore, the end result is more suited to be a candidate for implementation in mobile-commerce environments.

The remainder of this paper is organized as follows. Section 2 briefly reviews Yang et al.'s three-party authenticated protocol and then shows its weaknesses and disadvantages. Section 3 provides an improved scheme along with some important discussions. Finally, conclusion is presented in Section 4.

2 Review of Yang et al.'s protocol

This section briefly describes the three party AKE protocols proposed by Yang et al. [18], starting with some definitions and notations. Finally, we will point out its weaknesses and advantages.

2.1 Preliminaries

The notations used in their protocol are described as in the following:

- $\mathcal{E}$: an elliptic curve defined over a finite field $\mathbb{F}_p$ with large group order[19,20], where p is a large odd prime $p > 2^{160}$;
- Q: a point in $\mathcal{E}$ with large order q, where q is a secure large prime;
- G: the cyclic addition group generated by Q;
- d_I/U_I: a private/public key pair of I(a protocol participant), where $U_I = d_I * Q$ ("*" denotes the point multiplication over $\mathcal{E}$).
- $E_k(\cdot)/D_k(\cdot)$: a secure symmetric encryption/decryption algorithm (e.g., AES (Advanced Encryption Standard)[21]), where k denotes the symmetric key;
- ID_I: the identities of I(a protocol participant).

2.2 Protocol description

There are three entities involved in the protocol: the authentication server S, and two users A(initiator) and B(responder) who wish to establish a session key between them. And the protocol is divided into two phases: the initialization phase and the authenticated key exchange phase. Here, we just follows the description in [18]. In the initialization phase, it assumes that the two users A and B must register to the server S to generate their private keys and public keys . Then, A, B, and S have their private/public key pairs d_A/U_A, d_B/U_B, and d_S/U_S respectively. In the authenticated key exchange phase, A and B authenticate each other with S's help, then A and B can share a common session key. This phase is divided into three rounds which are illustrated as in Fig. 1. And a more detailed description follows.

Fig. 1. A high-level description of Yang et al.'s protocol.

A B S

A:
$r_A \in Z_q^*$
$R_A = r_A * U_A$
$\hat{R}_A = r_A * U_S$
$K_A = d_A * \hat{R}_A = (k_{Ax}, k_{Ay})$
$w_A \in Z_q^*$
$W_A = w_A * Q$
$C_A = E_{k_{Ax}}(R_A, W_A)$

$A \rightarrow B$: $(ID_A, Request)$

$A \rightarrow S$: (ID_A, ID_B, C_A, R_A)

B:
$r_B \in Z_q^*$
$R_B = r_B * U_B$
$\hat{R}_B = r_B * U_S$
$K_B = d_B * \hat{R}_B = (k_{Bx}, k_{By})$
$w_B \in Z_q^*$
$W_B = w_B * Q$
$C_B = E_{k_{Bx}}(R_B, W_B)$

$B \rightarrow A$: $(ID_B, Response)$

$B \rightarrow S$: (ID_B, ID_A, C_B, R_B)

S:
$K_A = d_S * R_A = (k_{Ax}, k_{Ay})$
$K_B = d_S * R_B = (k_{Bx}, k_{By})$
$(R_A, W_A) = D_{k_{Ax}}(C_A)$
$(R_B, W_B) = D_{k_{Bx}}(C_B)$
$Check \;\; R_A$
$Check \;\; R_B$
$C_{SA} = E_{k_{Ax}}(R_A, W_B)$
$C_{SB} = E_{k_{Bx}}(R_B, W_A)$

$S \rightarrow B$: (ID_S, C_{SB})

$S \rightarrow A$: (ID_S, C_{SA})

A:
$(R_A, W_B) = D_{k_{Ax}}(C_{SA})$
$Check \;\; R_A$
$SK_A = w_A * W_B$

B:
$(R_B, W_A) = D_{k_{Bx}}(C_{SB})$
$Check \;\; R_B$
$SK_B = w_B * W_A$

Round 1:

Step 1. A randomly selects an integer $r_A \in Z_q^*$ to compute $R_A = r_A * U_A$ and $\hat{R}_A = r_A * U_S$.

Step 2. A computes $K_A = d_A * \hat{R}_A = (k_{Ax}, k_{Ay})$, where k_{Ax} and k_{Ay} are x and y coordinates of K_A over $\mathcal{E}$ respectively.

Step 3. A randomly selects an integer $w_A \in Z_q^*$ to compute $W_A = w_A * Q$ and uses k_{Ax} as the encryption key to compute $C_A = E_{k_{Ax}}(R_A, W_A)$.

Step 4. A sends $(ID_A, Request)$ and (ID_A, ID_B, C_A, R_A) to B and S, respectively. Here, the message "*Request*" denotes a request that A asks B to share a session key with him.

Round 2:

Step 1. After receiving $(ID_A, Request)$, B randomly selects an integer $r_B \in Z_q^*$ to compute $R_B = r_B * U_B$ and $\hat{R}_B = r_B * U_S$.

Step 2. B computes $K_B = d_B * \hat{R}_B = (k_{Bx}, k_{By})$, where k_{Bx} and k_{By} are x and y coordinates of K_B over $\mathcal{E}$ respectively.

Step 3. B randomly selects an integer $w_B \in Z_q^*$ to compute $W_B = w_B * Q$ and uses k_{Bx} as the encryption key to compute $C_B = E_{k_{Bx}}(R_B, W_B)$.

Step 4. B sends $(ID_B, Response)$ and (ID_B, ID_A, C_B, R_B) to A and S, respectively. Here, the message "*Response*" denotes a response that B accepts A's request.

Round 3:

Step 1. After receiving (ID_A, ID_B, C_A, R_A) and (ID_B, ID_A, C_B, R_B), S computes $K_A = d_S * R_A = (k_{Ax}, k_{Ay})$ and $K_B = d_S * R_B = (k_{Bx}, k_{By})$.

Step 2. S uses k_{Ax} and k_{Bx} as the decryption keys to compute $(R_A, W_A) = D_{k_{Ax}}(C_A)$ and $(R_B, W_B) = D_{k_{Bx}}(C_B)$ respectively.

Step 3. S checks if the decrypted R_A is the same as R_A that was sent from A in Round 1. If they are the same, then S confirms that A is a valid user. Otherwise, S stops the protocol and sends an authentication-failed message to B. At the same time, S checks if the decrypted R_B is the same as R_B that was sent from B in Round 2. If they are the same, then S confirms that B is a valid user. Otherwise, S stops the protocol and sends an authentication-failed message to A.

Step 4. If A and B are both valid users, then S uses k_{Ax} and k_{Bx} as the symmetric keys to compute $C_{SA} = E_{k_{Ax}}(R_A, W_B)$ and $C_{SB} = E_{k_{Bx}}(R_B, W_A)$ respectively.

Step 5. S sends C_{SA} and C_{SB} to A and B, respectively.

Step 6. After receiving C_{SA}, A uses k_{Ax} as the decryption key to compute $(R_A, W_B) = D_{k_{Ax}}(C_{SA})$. Then, A checks if the decrypted R_A is the same as R_A that he selected in Round 1. If they are the same, then A confirms that B has been authenticated by S and he can obtain the session key by computing $SK_A = w_A * W_B$. Otherwise, A rejects the transaction.

Step 7. After receiving C_{SB}, B uses k_{Bx} as the decryption key to compute $(R_B, W_A) = D_{k_{Bx}}(C_{SB})$. Then, B checks if the decrypted R_B is the same as R_B that he selected in Round 2. If they are the same, then B confirms that A has been authenticated by S and he can obtain the session key by computing $SK_B = w_B * W_A$. Otherwise, B rejects the transaction.

The correctness of the protocol follows from the fact that, in an honest execution of the protocol, $SK_A = SK_B = w_A w_B * Q$.

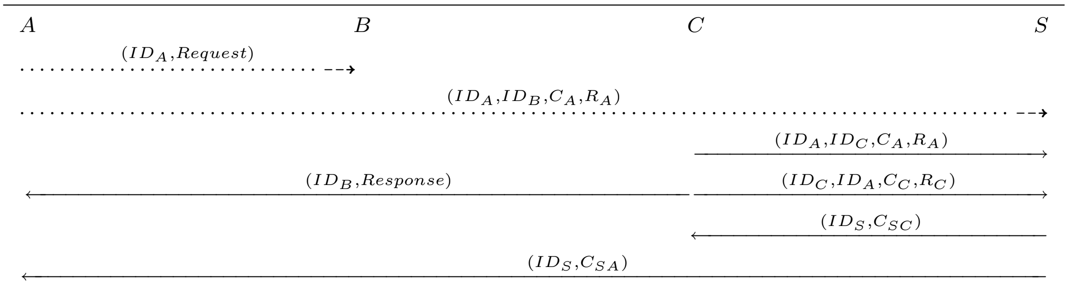
2.3 Weaknesses of Yang et al.'s protocol

Unfortunately, Yang et al.'s protocol[18] described above is completely insecure in the presence of an active adversary. In addition, we still find some disadvantages in their scheme.

2.3.1 unknown key-share attack.

Firstly, we shows that it is potentially vulnerable to an unknown key-share attack, by which an adversary can deceive the protocol principals about the identity of the peer entity[22]. In particular, any legitimate user not supposedly involved in a protocol run, say C, who has his private/public key pair d_C/U_C, can end up sharing a session key with user A but with A thinking it is sharing with user B who is not sharing any key with A or C. The attack scenario is outlined in Fig. 2, where a dashed line indicates that the corresponding message is intercepted by C enroute to its destination. A more detailed description of the attack is as follows:

Fig. 2. An unknown key-share attack on Yang et al.'s protocol.



(1) The protocol steps proceed as normal with A sending sending $(ID_A, Request)$ and (ID_A, ID_B, C_A, R_A) to B and S respectively notifying them that it wishes to initiate a session.

(2) Another user C intercepts the message (ID_A, ID_B, C_A, R_A) and instead sends (ID_A, ID_C, C_A, R_A) to S as if it originated from A at first, causing S to believe that A and C wish to establish a protocol session. Afterward, it randomly chooses $r_C, w_C \in Z_q^*$ and computes $R_C = r_C * U_C$, $\hat{R}_C = r_C * U_S$, $K_C = d_C * \hat{R}_C = (k_{Cx}, k_{Cy})$, $W_C = w_C * Q$, $C_C = E_{k_{Cx}}(R_C, W_C)$, where k_{Cx} and k_{Cy} are x and y coordinates of K_C over $\mathcal{E}$ respectively. C then sends

$(ID_B, Response)$ and (ID_C, ID_A, C_C, R_C) to A and S, respectively, as if it originated from B, causing them to believe that B accepts A's request.

(3) The rest of the steps proceed in a straightforward manner, but where C_C and R_C are used instead of C_B and R_B, respectively.

(4) S then outputs (ID_S, C_{SA}) and (ID_S, C_{SC}) to A and C, respectively.

(5) After receiving C_{SA}, A will use k_{Ax} as the decryption key to compute $(R_A, W_B) = D_{k_{Ax}}(C_{SA})$ and then obtain the secret key by computing $SK_A = w_A * W_C$.

(6) After receiving C_{SC}, C will use k_{Cx} as the decryption key to compute $(R_C, W_A) = D_{k_{Cx}}(C_{SC})$ and then obtain the secret key by computing $SK_B = w_C * W_A$.

In the above attack, a malicious user C impersonates B to respond to A when A requests to initiate an instance of the protocol with B. And A ends up thinking it is sharing a key with B when it is actually sharing with C and C knows what this key $SK_A = SK_B$ is. Meanwhile, B need not be present at all. This attack is an impersonation-of-responder attack[10]. Through the attack, the authentication mechanism of the protocol is completely compromised. More specifically, Yang et al.'s protocol does not satisfy *implicit key authentication*(i.e. both parties are ensured that no other principals aside from their intended peers may learn the established secret key [22,23]) , which is the fundamental security property that any given key exchange protocol is expected to possess, of course, in the presence of active adversaries who may read, modify, insert, delete, replay and delay messages [24–27].

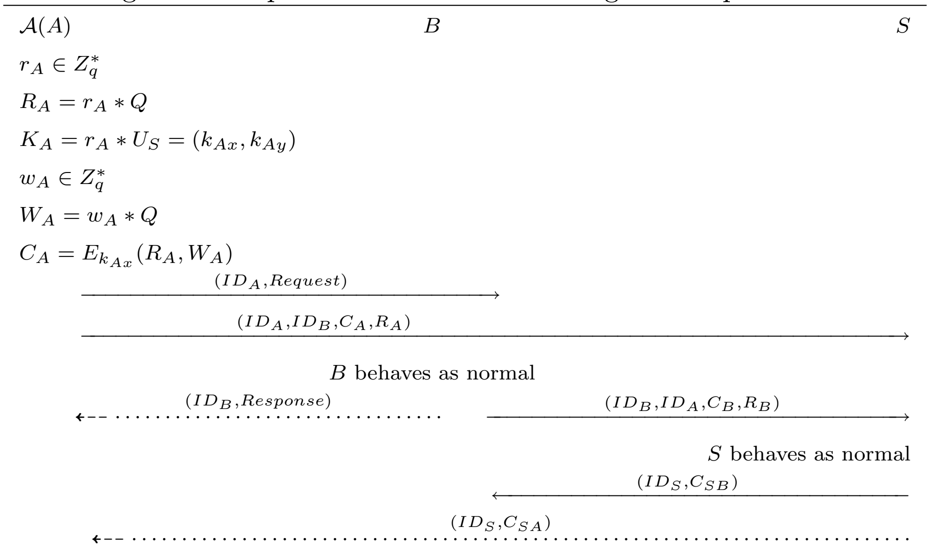
Similarly, C also can mounts an impersonation-of-initiator attack, in which it can impersonate A to initiate an instance of the protocol with B and end up sharing a session key with user B but with B fooled into believing that C is A. Since the rationale for it is quite similar to that of the impersonation-of-responder attack described above, the description is omitted here. One can easily remark that C can perform a man-in-the-middle attack[28] between A and B, not as Yang et al. claimed in[18], by subtly employing the impersonation-of-initiator attack and the impersonation-of-responder attack described here. Consequently, A will be fooled into believing that C is B and B will be fooled into believing that C is A. Furthermore, since C knows the two session keys shared with A and B respectively, C can decrypt all the ciphertexts transmitted between A and B.

As a result, any legitimate user (an insider) not supposedly involved in a protocol run can easily exploit it to compromise the authentication mechanism of the protocol completely. Our attacks also demonstrate that, when moving from two parties to three parties, the existence of malicious legitimate users needs to be taken into consideration [29–31]. Please note a malicious user can also be interpreted to a adversary that has compromised some legitimate user, say C, and thus known its long-term keys [16]. This approach is what we use in the security model(to be introduced later).

2.3.2 Impersonation attack

Firstly, we shows that it is potentially vulnerable to an impersonation attack. In particular, the adversary $\mathcal{A}$ can impersonates successfully any legitimate user, say A, to fool another user B into believing that it is A. The attack scenario is outlined in Fig. 3. A more detailed description of the attack is as follows:

Fig. 3. An impersonation attack on Yang et al.'s protocol.



(1) $\mathcal{A}$ randomly selects an integer $r_A \in Z_q^*$ to compute $R_A = r_A * Q$, and $K_A = r_A * U_S = (k_{Ax}, k_{Ay})$. Then $\mathcal{A}$ randomly selects an integer $w_A \in Z_q^*$ to compute $W_A = w_A * Q$ and uses k_{Ax} as the encryption key to compute $C_A = E_{k_{Ax}}(R_A, W_A)$. Finally, he sends $(ID_A, Request)$ and (ID_A, ID_B, C_A, R_A) to B and S, respectively, as if it originated from A, causing them to believe that A it wishes to initiate a session.

(2) After receiving $(ID_A, Request)$, B behaves as normal. B then outputs $(ID_B, Response)$ and (ID_B, ID_A, C_B, R_B) to $A(\mathcal{A})$ and S, respectively.

(3) After receiving (ID_A, ID_B, C_A, R_A) and (ID_B, ID_A, C_B, R_B), S behaves as normal. Please note, since $K_A = d_S * R_A = d_S r_A * Q = r_A * U_S$ holds in the current case, S will confirm that $\mathcal{A}$ is the valid user A. Then S sends C_{SA} and C_{SB} to A and B, respectively.

(4) $\mathcal{A}$ intercepts the message C_{SA}. And he uses k_{Ax} as the decryption key to compute $(R_A, W_B) = D_{k_{Ax}}(C_{SA})$. Finally, $\mathcal{A}$ can obtain the session key by computing $SK_A = w_A * W_B$.

(5) After receiving C_{SB}, B proceeds as normal. Finally, A can obtain the session key by computing $SK_B = w_B * W_A$. Furthermore, he will believe he is sharing the session key with A while A is not present at all. B is actually sharing with $\mathcal{A}$ since $SK_A = SK_B$ still holds in the current case.

In the above attack, the adversary $\mathcal{A}$ impersonates A to interact with B and S. In the end B was fooled into believing that $\mathcal{A}$ is A and $\mathcal{A}$ knows the session. Through the attack, the authentication mechanism of the protocol is completely compromised. Similarly, $\mathcal{A}$ can impersonate B to respond to A successfully. Since the rationale for it is quite similar, the description is omitted here.

2.3.3 Disadvantages

We find that Yang et al.'s scheme [18] on ECC has the following disadvantages. (1)First, the authentication mechanism in their scheme depends solely on a long-term private key of each user, e.g. d_A , which can be risky because an attacker can successfully forge A to communicate with S if the card used to store this key is stolen by the attacker; (2) Second, their scheme accomplished authentication using public-key and thus the server needs a large storage space to store users' public keys and certificates. (3) Third, their scheme only has some heuristic security arguments and lacks formal security proof. To overcome these disadvantages, we propose an improved authentication scheme on ECC in the next section.

Finally, we should note that there are some potential security issues in the way of key derivation in Yang et al.'s scheme. In a modern context, for security of a key exchange protocol, we usually require that, far from obtaining the whole key, the adversary cannot even reliably distinguish between the session key and a randomly chosen string of the expected length. However, the final key in their scheme has the form $w_A w_B * Q$ and is therefore an element of the cyclic group G. This, however, can be distinguishable from a randomly chosen value of the same size in case that this value is not in G. Thus, the computation of $w_A w_B * Q$ is not enough and some additional randomness extraction operation should be executed.

3 Our Improved Protocol

In this section, we present a secure and efficient three-party authenticated key exchange protocol for mobile-commerce environments. Our improved scheme can not only defeat the attacks described in the previous section but also can overcome those disadvantages existing in Yang et al's scheme[18]. Finally, we also provide some important remarks on it in this section.

3.1 Description

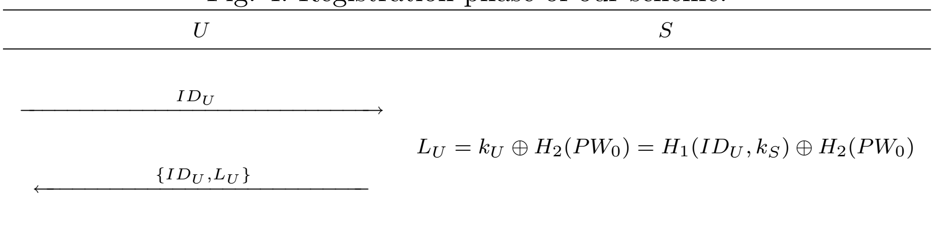
We combine two factors authentication mechanisms in our scheme so that each user must have the smart-card and know the password in order to accomplish authentication and key exchange with other users. That is, our scheme is a smart-card-based password authenticated key exchange protocol in the three party setting. Our scheme consists of two phases: user registration phase and the authenticated key exchange phase. In user registration phase, each user must register to the server S in order to become a legal user and will receive a smart card issued by the server S. In the authenticated key exchange phase, two users A and B authenticate each other with S's help, then A and B can share a common session key.

First, we define some notations used in our scheme. Let Q be a base point in an elliptic curve with large prime order q and G the cyclic addition group generated by Q. $H_1(\cdot)$, $H_2(\cdot)$ and $H_3(\cdot)$: $\{0,1\}^* \to \{0,1\}^l$ are three public one-way hash functions, where l is the security parameter. In our scheme, the server S has a private key k_S and a private/public key pair (d_S, U_S) with $U_S = d_S * Q$; and each user U has an unique identity ID_U and a smart card issued by the server. The security of our protocol mainly relies on the EC computational Diffie-Hellman (ECCDH) assumption. In the ECCDH assumption, given $W = w * Q$ and $V = v * Q$, where w and v are drawn randomly from Z_q^*, it is computationally infeasible to compute $uv * Q$(denoted by $ECCDH(W, V)$).

Now, we introduce our improved scheme as follows:

Registration phase: Server S issues a smart-card to user U as follows, which is described in Fig 4.

Fig. 4. Registration phase of our scheme.



Step 1. The user U sends his identity ID_U to the server.
Step 2. The server S computes $k_U = H_1(ID_U, k_S)$ and then $L_U = k_U \oplus H_2(PW_0)$, where $\oplus$ is the exclusive-OR operation on bit strings and PW_0 the initial password (e.g. a default password such as a string of all "1"). Then, S issues U a smart-card which contains ID_U, L_U and all the system parameters needed in our scheme. In our scheme, we assume some protections have implemented to prevent the secret information L_U from being read out of the card.

Step 3. After receiving the smart-card , U changes the password immediately.

Authenticated key exchange phase: A(resp. B) inserts his smart card into the mobile input device and enters password PW_A (resp. PW_B). The smart-card retrieves the value $k_A = L_A \oplus H_2(PW_A)$ (resp. $k_B = L_B \oplus H_2(PW_B)$). A and B (actually performed by the users' smart-cards) then use k_A and k_B respectively to perform authenticated key exchange with S's help. This phase is divided into three rounds which are illustrated as in Fig. 5. And a more detailed description follows.

Fig. 5. Authenticated key exchange phase of our scheme.

A	B	S
$k_A = L_A \oplus H_2(PW_A)$	$k_B = L_B \oplus H_2(PW_B)$	$k_S, (d_S, U_S)$
$r_A \in Z_q^*$		
$R_A = r_A * Q$		
$\alpha_A = H_3(\text{“1”}, ID_A, ID_B, R_A, k_A)$		
$\xrightarrow{(ID_A, Request)}$		
$\xrightarrow{(ID_A, ID_B, R_A, \alpha_A)}$		
	$r_B \in Z_q^*$	
	$R_B = r_B * Q$	
	$\alpha_B = H_3(\text{“1”}, ID_B, ID_A, R_B, k_B)$	
	$\xleftarrow{(ID_B, Response)}$ $\xrightarrow{(ID_B, ID_A, R_B, \alpha_B)}$	
		$k_A = H_1(ID_A, k_S)$
		$k_B = H_1(ID_B, k_S)$
		$\alpha_A' = H_3(\text{“1”}, ID_A, ID_B, R_A, k_A)$
		$\alpha_B' = H_3(\text{“1”}, ID_B, ID_A, R_B, k_B)$
		$Check\ \ \alpha_A' \stackrel{?}{=} \alpha_A$
		$Check\ \ \alpha_B' \stackrel{?}{=} \alpha_B$
		$T_A = d_S * R_A$
		$T_B = d_S * R_B$
		$\beta_A = H_3(\text{“2”}, ID_A, ID_B, R_A, T_A, k_A)$
		$\beta_B = H_3(\text{“2”}, ID_B, ID_A, R_B, T_B, k_B)$
		$\xleftarrow{(ID_S, R_A, \beta_B)}$
		$\xleftarrow{(ID_S, R_B, \beta_A)}$
	$T_B = r_B * U_S$	
$T_A = r_A * U_S$	$\beta_B' = H_3(\text{“2”}, ID_B, ID_A, R_B, T_B, k_B)$	
$\beta_A' = H_3(\text{“2”}, ID_A, ID_B, R_A, T_A, k_A)$	$Check\ \ \beta_B' \stackrel{?}{=} \beta_B$	
$Check\ \ \beta_A' \stackrel{?}{=} \beta_A$	$Z = r_B * R_A$	
$Z = r_A * R_B$	$SK_B = H_3(\text{“3”}, ID_A, ID_B, R_A, R_B, Z)$	
$SK_A = H_3(\text{“3”}, ID_A, ID_B, R_A, R_B, Z)$		

Round 1:

Step 1. A randomly selects an integer $r_A \in Z_q^*$ to compute $R_A = r_A * Q$.
Step 2. A computes the authenticator $\alpha_A = H_3(\text{“1”}, ID_A, ID_B, R_A, k_A)$.
Step 3. A sends $(ID_A, Request)$ and $(ID_A, ID_B, R_A, \alpha_A)$ to B and S, re-

spectively. Here, the message "*Request*" denotes a request that A asks B to share a session key with him.

Round 2:

Step 1. After receiving $(ID_A, Request)$, B randomly selects an integer $r_B \in Z_q^*$ to compute $R_B = r_B * Q$.

Step 2. B computes the authenticator $\alpha_B = H_3(\text{"1"}, ID_B, ID_A, R_B, k_B)$.

Step 3. B sends $(ID_B, Response)$ and $(ID_B, ID_A, R_B, \alpha_B)$ to A and S, respectively. Here, the message "*Response*" denotes a response that B accepts A's request.

Round 3:

Step 1. After receiving $(ID_A, ID_B, R_A, \alpha_A)$ and $(ID_B, ID_A, R_B, \alpha_B)$, S uses k_S to compute $k_A = H_1(ID_A, k_S)$ and $k_B = H_1(ID_B, k_S)$.

Step 2. S uses k_A and k_B to compute $\alpha_A^{'} = H_3(\text{"1"}, ID_A, ID_B, R_A, k_A)$ and $\alpha_B^{'} = H_3(\text{"1"}, ID_B, ID_A, R_B, k_B)$ respectively.

Step 3. S checks if the computed $\alpha_A^{'}$ is the same as α_A that was sent from A in Round 1. If they are the same, then S confirms that A is a valid user. Otherwise, S stops the protocol and sends an authentication-failed message to B. At the same time, S checks if the computed $\alpha_B^{'}$ is the same as α_B that was sent from B in Round 2. If they are the same, then S confirms that B is a valid user. Otherwise, S stops the protocol and sends an authentication-failed message to A.

Step 4. If A and B are both valid users, then S uses k_A and k_B to compute the authenticators $\beta_A = H_3(\text{"2"}, ID_A, ID_B, R_A, T_A, k_A)$ and $\beta_B = H_3(\text{"2"}, ID_B, ID_A, R_B, T_B, k_B)$ respectively, where $T_A = d_S * R_A$, $T_B = d_S * R_B$.

Step 5. S sends (ID_S, R_B, β_A) and (ID_S, R_A, β_B) to A and B, respectively.

Step 6. After receiving (ID_S, R_B, β_A), A computes $T_A = r_A * U_S$ and $\beta_A^{'} = H_3(\text{"2"}, ID_A, ID_B, R_A, T_A, k_A)$. Then, A checks if the computed $\beta_A^{'}$ is the same as β_A that was sent from S. If they are the same, then A confirms that B has been authenticated by S and he can obtain the session key by computing $Z = r_A * R_B$ and then $SK_A = H_3(\text{"3"}, ID_A, ID_B, R_A, R_B, Z)$. Otherwise, A rejects the transaction.

Step 7. After receiving (ID_S, R_A, β_B), B computes $T_B = r_B * U_S$ and $\beta_B^{'} = H_3(\text{"2"}, ID_B, ID_A, R_B, T_B, k_B)$. Then, B checks if the computed $\beta_B^{'}$ is the same as β_B that was sent from S. If they are the same, then B confirms that A has been authenticated by S and he can obtain the session key by computing $Z = r_B * R_A$ and then $SK_B = H_3(\text{"3"}, ID_A, ID_B, R_A, R_B, Z)$. Otherwise, B rejects the transaction.

The correctness of our protocol follows from the fact that, in an honest execution of the protocol, $T_A = r_A * U_S = d_S \cdot R_A$, $T_B = r_B * U_S = d_S \cdot R_B$ and $Z = r_A \cdot R_B = r_B \cdot R_A$. Furthermore, we can provide the rigorous proof of the security for our scheme under the assumptions that the hash function closely behaves like a random oracle and that the EC computational Diffie-Hellman problem is difficult. The security model is that was used in [31]. We omitted it here.

And one can easily remark that the attacks described in Section 2.3.1 is no longer effective in our protocol. The weakness existing in Yang et al's scheme is due to the fact that there is no way for the server to check whether the two identities of users contained in the received message are correctly paired or not and the protocol does not provide each user with any proof necessary to verify that the other user is as it think the latter is. The problems are fixed now in our scheme by having the identities of two users included as part of the message inputs in the computation of the authenticators: α_A, α_B, β_A and β_B. This technique effectively defeats the attacks mentioned above.

You may wonder why we should include $T_A = r_A * U_S = d_S \cdot R_A$ and $T_B = r_B * U_S = d_S \cdot R_B$ in the computation of β_A and β_B respectively rather than just compute S's authenticators as $\beta_A = H_3(\text{“2”}, ID_A, ID_B, R_A, k_A)$ and $\beta_B = H_3(\text{“2”}, ID_B, ID_A, R_B, k_B)$. To understand this, let us consider an extreme case that A's authentication data k_A is compromised, or equivalently both his password and smart-card are gained by the attacker. In that case, the attacker will not only be able to masquerade as A but also as B (or the server). By using this technique, the adversary can not masquerade as B by taking the role of the server S to reply with a valid authenticator β_A to A for the challenge R_A any more since he can not know the value of $T = ECCDH(R_A, U_S)$ based on the hardness of elliptic curve computational Diffie-Hellman problem. In other words, our scheme can provide resilience against key-compromise impersonation. This security property is a desirable one that any given key exchange protocol is expected to possess[22].

Finally, unlike Yang et al.'s protocol, a key derivation function is used to get session keys from the agreed upon secret Z in our protocol. More precisely, the hash function H_3 is used in the computation of SK_A and SK_B, i.e. A and B computes $SK_A = H_3(\text{“3”}, ID_A, ID_B, R_A, R_B, Z)$, $SK_B = H_3(\text{“3”}, ID_A, ID_B, R_A, R_B, Z)$ respectively. As a result, the adversary cannot reliably distinguish between the session key and a randomly chosen string of the expected length. On the other hand, it is also necessary to use a hash function to compute the session key in our proof. Actually, under the assumption that the computational Diffie-Hellman problem in G is difficult, we can show its security in random oracle(ideal hash model). We will discuss it in details later.

3.2 Discussions

In this subsection, we discuss its attractive features in contrast to Yang et al's scheme.

In addition to resistance against the so-called unknown key-share attacks and impersonation, our scheme has the following advantages over Yang et al's scheme:

(1) Our scheme is a two-factor mutual authentication scheme based on smart cards and passwords. At the protocol level, k_A (resp. k_B) is indeed the authentication data used in the authenticated key exchange phase. Therefore, in order to gain help from the server, one must have the smart-card and know the password of the user so that the value k_A (resp. k_B) can be retrieved to perform the protocol. Even in an extreme case that the adversary gets the user's smart cart (with some protections to prevent the secret information from being read out), our scheme still can protect the password information against the notorious password guessing attacks by which attackers could search the relatively small space of human-memorable passwords. With the smart card of a user, say A, the attacker can guess a password and enter it into the card to run the protocol. As the specification, the card will output α_A associated with a fresh R_A. Since R_A is a random element chosen by the card and the expected α_A related with this new R_A unlikely appears in the previous executions with an instance of the user, the attacker can not verify his guess unless he forwards this message to the server and see whether the server accepts it. However, both the server and the card will invalidate or block the request from that user whenever a certain number of failed attempts occurs.
(2) Our scheme scheme utilizes each user's unique identity to accomplish authentication, instead of using public keys. The server S uses its private key k_S and the user's unique identity ID_A(resp. ID_B) to derive k_A (resp. k_B) for authentication. Thus, the server does not need to maintain a large public-key table while the number of users becomes very large. Therefore, our scheme provides high scalability for the user addition in mobile-commerce environments.

To sum up, our scheme overcomes all disadvantages mentioned in the section 2.3.

At the same time, the merits of Yang et al's original scheme in [18] are left unchanged in the our scheme. As pointed out in Section 3.1, our scheme also provides resilience against key-compromise impersonation. When the long-term key of an entity is compromised, the adversary will be able to masquerade as the the entity but the situation will be even worse if the adversary can also masquerade as another entity. We say a protocol offers resilience against key-compromise impersonation if it can prevent this attack. Furthermore, even when both k_A and k_B are compromised, the adversary still can not know the previous session keys that were established before the corruption (which is usually called forward secrecy).

Furthermore, our scheme is simpler and more efficient than Yang et al's scheme in[18].

- Firstly, our scheme is more efficient than Yang et al's scheme. As shown in Table 1, on one honest run of our authentication protocol, each party no longer needs to perform symmetric-key encryption/decryption operations. For simplicity, the computation costs of Table 1 do not include the hashing computations since it can be done much more efficiently both in time and energy consumption than point multiplication and symmetric-key encryption/decryption operations(note, to compute C_A in their scheme, the message (R_A, W_A) should be divided into some blocks and thus several encryption operations of AES are needed actually), based on the experimental results of related researches[32–37].
- Secondly, the server can efficiently test the validness of each message sent from each user. Since each message is sent along with a hashing value as its authenticator in our scheme, the server needs to perform a hashing operation and then make comparison in a straightway to validate a message. However, to achieve the same goal, each party needs to perform two point multiplication plus some symmetric-key decryption operations in Yang et al's scheme. As a result, the server can have a better tolerance of the so-called denial of service attack because a lot of operating time used in checking could be saved if an invalid message is received. Therefore, our scheme is more robust.
- Besides, our scheme allows mobile users to change their password freely.

Finally, we list the comparisons of our scheme and Yang et al's scheme on ECC in Table 1. Based on the results listed in the table, we conclude that our scheme is more practical for the users of mobile devices.

Table 1. Comparisons with Yang et al's work

Properties		Schemes	
		Yang et al	Ours
security	authentication mechanism	one-factor	two-factor
	against unknown key-share attack	No	Yes
	Provable security	No	Yes
performance	Storage requirement on S	High	Low
	Computation costs *	U: 3PM+2SE	U: 3PM
		S: 2PM+4SE	S: 2PM
	Communication rounds	3	3

*PM: Elliptic curve point multiplication; SE: Symmetric-key encryption/decryption.

4 Conclusion

In this paper, we have demonstrated that Yang et al's three-party authenticated protocol is potentially vulnerable to an unknown key-share attack and impersonation attack. Furthermore, we have proposed a secure and efficient three-party authenticated key exchange protocol for mobile-commerce environments. Our improved protocol not only defeats the attacks described by us but also overcomes all disadvantages of Yang et al.'s protocol. Thus the end result is more practical for the users of mobile devices.

References

[1] S.M. Bellovin and M. Merrit. Encrypted key exchanged: password-based protocols secure against dictionary attacks. In Proceedings of IEEE Symposium on Research in Security and Privacy, 1992, pp. 72-84.

[2] Y.J. Choie, E. Jeong, E. Lee. Efficient identity-based authenticated key agreement protocol from pairings. Applied Mathematics and Computation, 2005, 162, 179-188.

[3] X. Tiang, D.S. Wong, R.W. Zhu. Analysis and improvement of an authenticated key exchange protocol for sensor networks. IEEE Communications Letters, 2005, 9 (11), 970-972.

[4] C.C. Chang, Y.F. Chang. A novel three-party encrypted key exchange protocol. Computer Standards and Interfaces, 2004, 26 (5), 472-476.

[5] T.F. Lee, T. Hwang, C.L. Lin. Enhanced three-party encrypted key exchange without server public keys. Computer and Security, 2004, 23 (7), 571-577.

[6] C.L. Lin, H.M. Sun, T. Hwang. Three-party encrypted key exchange: attacks and a solution. ACM Operating Systems Review, 2000, 34, 12-20.

[7] H.M. Sun, B.C. Chen, T. Hwang. Secure key agreement protocols for three party against guessing attacks. Journal of Systems and Software, 2005, 75, 63-68.

[8] R.X. Lu , Z.F. Cao. Simple three-party key exchange protocol, Computers and Security 2007, Vol. 26, pp. 94-97.

[9] H. Guo, Z. Li, Y. Mu, X. Zhang. Cryptanalysis of simple three-party key exchange protocol. Computers and Security, 27(2008), pp. 16-21.

[10] H.-R. Chung, W.-C. Ku. Three weaknesses in a simple three-party key exchange protocol, Information Science, 2008, Vol. 178, pp. 220-229.

[11] N.W. Lo, Kuo-Hui Yeh. Cryptanalysis of two three party encrypted key exchange protocols, Computer Standards & Interfaces (2009), doi: 10.1016/j.csi.2009.03.002

[12] C.-W. Phan Raphael, W.-C. Yau , Bok-Min G.. Cryptanalysis of simple three-party key exchange protocol (S-3PAKE), Information Science 2008, Vol. 178, pp. 2849-2856.

[13] H.-S. Kim, J.-Y. Choi. Enhanced password-based simple three-party key exchange protocol, Computers and Electrical Engineering (2009), Vol. 35, pp. 107C114.

[14] J. Nam. Infringing and Improving Password Security of a Three-Party Key Exchange Protocol. Available at http://eprint.iacr.org/2008/065/.

[15] J. Nam , Y. Lee , S. Kim , D. Won. Security weakness in a three-party pairing-based protocol for password authenticated key exchange, Information Sciences, 2007, Vol. 177(6), pp. 1364-1375.

[16] K.-K.R. Choo, C. Boyd, Y. Hitchcock. Examining indistinguishability-based proof models for key establishment protocols, in: Advances in Cryptology C Asiacrypt05, LNCS, vol. 3788, 2005, pp. 585-604.

[17] T.H. Chen, W.B. Lee, H.B. Chen. A round-and computation-efficient threeparty authenticated key exchange protocol. Journal of Systems and Software, 2008, 81, 1581-1590.

[18] J.-H. Yang, C.-C. Chang. An efficient three-party authenticated key exchange protocol using elliptic curve cryptography for mobile-commerce environments. The Journal of Systems and Software (2009), doi:10.1016/j.jss.2009.03.075.

[19] D. Hankerson, A. Menezes, S. Vanstone. Guide to elliptic curve cryptography. Springer-Verlag, New York, USA, 2004.

[20] N. Koblitz. Elliptic curve cryptosystem. Mathematics of Computation, 1987, 48, 203-209.

[21] Advanced Encryption Standard, <http://www.csrc.nist.gov/archieve/aes/>. Global System for Mobile Communications, <http://www.gsmworld.com/>.

[22] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment, Springer-Verlag, 2003.

[23] G. Ateniese, M. Steiner, G. Tsudik, New multiparty authentication services and key agreement protocols, IEEE Journal on Selected Areas in Communications, 2000, 18 (4), 628-639.

[24] M. Bellare, P. Rogaway, Entity authentication and key distribution, in: Proc. Crypto'93, LNCS, vol. 773, 1993, pp. 232-249.

[25] M. Bellare, P. Rogaway, Provably secure session key distributionthe three party case, in: Proc. 27th ACM Symposium on Theory of Computing (STOC'95), 1995, pp. 57-66.

[26] M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, in: Proc. Eurocrypt'00, LNCS, vol. 1807, 2000, pp. 139-155.

[27] R. Canetti, H. Krawczyk, Analysis of key-exchange protocols and their use for building secure channels, in: Proc. Eurocrypt'01, LNCS, vol. 2045, 2001, pp. 453-474.

[28] A.J. Menezes, P.C. Orschot, S.A. Vanstone. Hand-Book of Applied Cryptography. CRC Press, 1996.

[29] J. Katz, J.S. Shin, Modeling insider attacks on group-key exchange protocols, in: Proceedings of the ACM-CCS'05, 2005, pp. 180-189.

[30] M. Abdalla, P.-A. Fouque, and D. Pointcheval. Password-based authenticated key exchange in the three-party settin, International Workshop on Theory and Practice in Public Key Cryptography: Proceedings of PKC'05, LNCS 3386, Springer-Verlag, Berlin, 2005, pp. 65-84.

[31] M. Abdalla, D. Pointcheval. Interactive Diffie-Hellman Assumptions with Applications to Password-based Authentication . in , International Conference on Financial Cryptography: Proceedings of FC'05, LNCS 3570, Springer-Verlag, Berlin, 2005, pp. 341-356.

[32] D.S. Wong, H.H. Fuentes, A.H. Chan, The performance measurement of cryptographic primitives on palm devices, in: Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC 2001), New Orleans, USA, 2001, pp. 92-101.

[33] P.G. Argyroudis, R. Verma, H. Tewari, D. OMahony, Performance analysis of cryptographic protocols on handheld devices, in: Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications (NCA 2004), Cambridge, USA, Sep. 2004, pp. 169-174.

[34] M. Passing, F. Dressler, Experimental performance evaluation of cryptographic algorithms, in: Proceedings of the 3rd IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), Vancouver, Canada, 2006, pp. 882-887.

[35] M. Passing, F. Dressler, Practical evaluation of the performance impact of security mechanisms in sensor networks, in: Proceedings of the 31st IEEE Conference on Local Computer Networks, Tampa, USA, 2006, pp. 623-629.

[36] M.R. Doomun, K.S. Soyjaudah, D. Bundhoo, Energy consumption and computational analysis of Rijndael-AES, in: Proceedings of the Third IEEE International Conference in Central Asia on Internet the Next Generation of Mobile, Wireless and Optical Communications Networks (ICI 2007), Uzbekistan, 2007, pp. 1-6.

[37] N.R. Potlapally, S. Ravi, A. Raghunathan, N.K. Jha, A study of the energy consumption characteristics of cryptographic algorithms and security protocols, IEEE Transactions on Mobile Computing 5 (2) (2006) 128-143.