An ID-based three-party authenticated key exchange protocol using elliptic curve cryptography for mobile-commerce environments

Debiao He*, Yitao Chen

School of Mathematics and Statistics, Wuhan University, Wuhan, People's Republic of China

Email: hedebiao@163.com

Abstract*:* For secure communications in public network environments, various three-party authenticated key exchange (3PAKE) protocols are proposed to provide the transaction confidentiality and efficiency. In 2009, Yang et al. proposed an efficient three-party authenticated key exchange protocol based upon elliptic curve cryptography(ECC) for mobile-commerce environments. Because the elliptic curve cryptography is used, their 3PAKE protocol has low computation costs and light communication loads. However, Tan demonstrated that Yang et al.'s protocol suffers from the impersonation attack and the parallel attack. Tan also proposed an enhanced protocol to improve the security and the performance. However, Yang et al.'s protocol and Tan's protocol bases on the public key infrastructure(PKI). Then the server has to maintain the certificates for users' public keys. When the number of users is increased, the server needs a large storage space to store users' public keys and certificates. In addition, the server needs additional computations to verify the other's certificate in their protocols. This causes the computation loads and the energy costs of mobile devices very high. In this paper, we propose an ID-based 3PAKE using ECC. Compared with the related protocol, our protocol does not need additional computations to verify certificate and has the better performance. Then our protocol is more suitable and practical for mobile-commerce environments.

Key words*: ID-based; Three-party key exchange; Elliptic curve Cryptosystem*

1. Introduction

Key exchange protocol is one of the most important cryptography mechanisms, by which a pair of person who communicate over a public insecure channel can generate a secret session key to guarantee their later communications' privacy and data integrity. Bellovin and Merritt [1] developed a two-party password-based authentication key exchange (2PAKE) protocol in which party authentication and key exchange techniques always are adopted. Two parties in

communication share a password, authenticate each other and obtain a common ephemeral session key[1]. Since then, many 2PAKE protocols are proposed [2,3].

Although 2PAKE protocols are quite useful for user－server architectures, they are not suitable for large scale communication environments since 2PAKE protocols require every pair of communication entities to share a password, it is very inconvenient in key management for user－user communications in large-scale communication environments. To avoid this inconvenience, several three-party password-based authenticated key exchange (3PAKE) protocols [4-8] have been proposed. Generally speaking, 3PAKE protocols can be classified into two classes: without servers [4] and with a server[5-8]. The former is a special case of multi-party key agreement protocols. In the later, two parties can cooperate to produce a common session key with the help of the server. In the following, the protocols to be discussed are 3PAKE protocols with a server. The 3PAKE protocol with a server demands that each communication entity shares a password with a trusted server. Thus, any two communication entities can be achieved mutual authentication and secure communication through the server's assistance.

In order to improve the performance of the previous protocols[5-8], Chen et al.[9] proposed an efficient three-party authenticated key exchange protocol based upon traditional public-key cryptosystems (PKC) [10]. Compared with the previous 3PAKE protocols[5-8], Chen et al.'s protocol uses fewer communication rounds to accomplish the mutual authentication and key exchange between any two parties in a large group. Besides, their protocol does not need to store a large password table in the server. Yang et al.[11] pointed that Chen et al.'s protocol has the following drawbacks. First, the computation cost of their protocol is still high because it needs to generate and verify Schnorr's digital signature. Thus, their protocol has to compute modular exponentiation, which is a time-consuming operation. Second, their protocol has heavy communication costs because it requires transmitting additional messages for mutual authentication. Third, their protocol suffers from the stolen-verifier attack. Then Yang et al.[11] proposed an efficient three-party authenticated key exchange protocol based upon elliptic curve cryptography(ECC) for mobile-commerce environments.

The ECC was first proposed by Miller [12] and Koblitz [13], and its security was based upon the difficulty of elliptic curve discrete logarithm problem (ECDLP). Compared with the traditional PKC, ECC offers a better performance

because it can achieve the same security with a smaller key size. For example, 160-bit ECC and 1024-bit RSA have the same security level in practice [14]. Thus, ECC-based authentication protocols are more suitable for mobile devices than the traditional PKC-based ones.

Because the ECC is used, Yang et al.'s protocol has low computation costs and light communication loads. Compared with Chen et al.'s protocol, their protocol is more suitable and practical for mobile-commerce environments. However, Yang et al.'s protocol is susceptible to the parallel attack[15] and the impersonation attack[15]. In order to improve the security, Tan [15] proposed an enhanced 3PAKE using the ECC.

However, Tan's protocol bases on the public key infrastructure (PKI). Then the server has to maintain the certificates for users' public keys. When the number of users is increased, the server needs a large storage space to store users' public keys and certificates. In addition, users need additional computations to verify the other's certificate in their protocols. This causes the computation loads and the energy costs of mobile devices very high.

The ID-based cryptography is the best solution to the above problem. The ID-based concept was first introduced by Shamir [16] in 1984. In an ID-based protocol, the user utilizes his unique identity (e.g., name, address, or email address) as his public key. Thus, the user cannot claim that the authentication information containing his identity does not belong to him. Without public keys, the users do not need to perform additional computations to verify the corresponding certificates. Moreover, the server does not need to maintain a large public-key table because there is no public key in the ID-based protocols.

According to the above descriptions, the ID-based 3PAKE on ECC is very suitable for mobile-commerce environments. Based on the above work, we propose an ID-based 3PAKE based upon ECC in this paper. Since the ID-based concept is used, the proposed method not only reduces the computation cost for users and the server but also is more efficient than Yang et al.'s protocol[11] and Tan's protocol [15]. For ECC can achieve the same level of security with smaller key size, the proposed method is suitable for mobile communication environments (e.g., GSM (Global System for Mobile Communications) and 3GPP (The Third Generation Partnership Project).

The remainder of this paper is organized as follows. Section 2 describes the preliminaries. In Section 3, we propose our efficient mutual authentication and key exchange protocol. The security analysis of the proposed protocol is presented in Section 4. In Section 5, performance analysis and some experimental results are presented. Conclusions are given in Section 6.

2. Preliminaries

Let the symbol E/F_p denote an elliptic curve E over a prime finite field F_p, defined by an equation

$$y^2 = x^3 + ax + b\ , \quad a,b \in F_p \qquad (1)$$

and with the discriminant

$$\Delta = 4a^3 + 27b^2 \neq 0\ . \qquad (2)$$

The points on E/F_p together with an extra point O called the point at infinity form a group

$$G = \{(x,y) : x,y \in F_p, E(x,y) = 0\} \cup \{O\}\ . \qquad (3)$$

Let the order of G be n. G is a cyclic additive group under the point addition "+" defined as follows: Let $P,Q \in G$, l be the line containing P and Q (tangent line to E/F_p if $P = Q$), and R, the third point of intersection of l with E/F_p. Let l' be the line connecting R and O. Then P "+" Q is the point such that l' intersects E/F_p at R and O and P "+" Q. Scalar multiplication over E/F_p can be computed as follows:

$$tP = P + P + \cdots + P(t\ times) \qquad (4).$$

3. The proposed protocol

In this section, we propose our ID-based 3PAKE protocol for mobile-commerce environments on ECC. The proposed protocol is divided into three phases: the system initialization phase, the user registration phase, and the authenticated key exchange phase.

- **The system initializing phase**

In this phase, S generates parameter of the system.

Step 1. S chooses an elliptic curve equation E .

Step 2. S selects a base point P with the order n over E.

Step 3. S selects its master key x and computes public key $P_{pub} = xP$.

Step 4. The server chooses four secure hash functions $H_1 : \{0,1\}^* \to Z_n^*$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times \{0,1\}^* \to Z_n^*$, $H_4 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times G \to Z_n^*$ and $H_3 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times \{0,1\}^* \times G \times \{0,1\}^* \to Z_n^*$. The server keeps x in private and publishes (F_p, E, n, P, P_{pub}, H_1, H_2, H_3, H_4).

- **The user registration phase**

When a low-power computing user U with the identity ID_U wants to register to the server S, U submits its identity ID_U to S. After S receiving user's identity, S first uses ID_U to compute $h_U = H_1(ID_U)$ and then uses the system private key x to compute the user's private key $D_U = \frac{1}{x + h_U} P \in G$ and then delivers D_U to U.

To deliver the private key D_U to the user U, two approaches may be used, and both of them require to be performed only once. One is off-line approach that the server S stores the identity ID_U, the private key D_U into a smartcard and returns it to the user U. The other one is on-line approach. When the user U connects to the server S through Internet, the server S may use the Secure Socket Layer (SSL) channel in the https mode to deliver the private key D_U to the user U.

- **The authenticated key exchange phase**

In this phase, two user A and B, with the identity ID_A and ID_B separately, 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 shown as follows.

Round 1:

Step 1. A generates a random number $a \in Z_n^*$ to compute $R_A = aP$ and $R'_A = aD_A$.

Step 2. A computes $Auth_{AS} = H_2(ID_A, ID_B, R_A, R'_A, T_A)$, where T_A is the current timestamp.

Step 3. A sends $M_1 = (ID_A, Request)$ and $M_2 = (ID_A, ID_B, R_A, T_A, Auth_{AS})$ to B and S, respectively. Here, the message "*Request*" denotes a request that A asks B to share a session key with him.

We show Round 1 of this phase in Fig. 1.

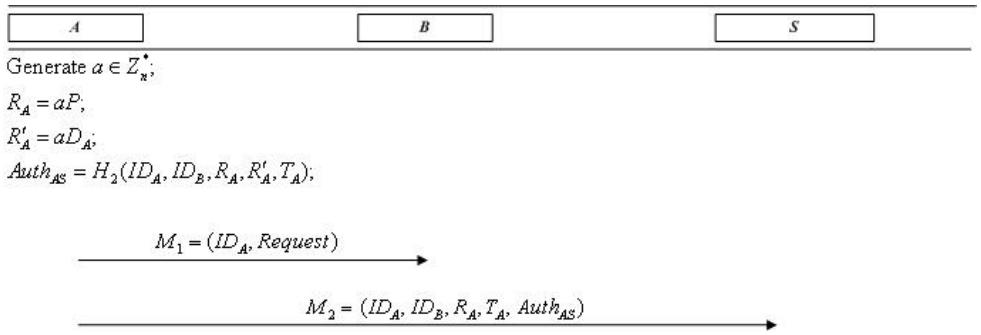

Fig. 1. Round 1 of the authenticated key exchange phase

Round 2:

Step 1. After receiving (ID_A, *Request*), B randomly selects an integer $b \in Z_n^*$ to compute $R_B = bP$ and $R'_B = bD_B$.

Step 2. B computes $Auth_{BS} = H_2(ID_B, ID_A, R_B, R'_B, T_B)$, where T_B is the current timestamp.

Step 3. B sends $M_3 = (ID_B, Request)$ and $M_4 = (ID_B, ID_A, R_B, T_B, Auth_{BS})$ to A and S, respectively. Here, the message "*Request*" denotes a response that B accepts A's request.

We illustrate Round 2 of this phase in Fig. 2.

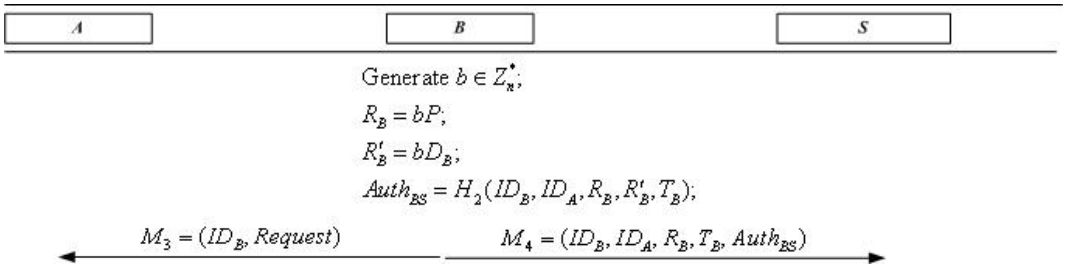

Fig. 2. Round 2 of the authenticated key exchange phase

Round 3:

Step 1. On receiving M_2 and M_4, S checks the freshness of T_A, T_B and the validity of ID_A, ID_B. If T_A or T_B is not fresh or ID_A or ID_B is not valid, S stops the session.

Step 2 . S computes $h_A = H_1(ID_A)$, $h_B = H_1(ID_B)$, $R''_A = \frac{1}{x + h_A} R_A$ and $R''_B = \frac{1}{x + h_B} R_B$.

Step 3 . S checks whether $Auth_{AS}$ and $Auth_{BS}$ equals $H_2(ID_A, ID_B, R_A, R''_A, T_A)$ and $H_2(ID_B, ID_A, R_B, R''_B, T_B)$ separately. If not, S stops the session.

Step 4 . S computes $Auth_{SA} = H_3(ID_A, ID_B, R_A, R''_A, T_A, R_B, T_S)$ and $Auth_{SB} = H_3(ID_B, ID_A, R_B, R''_B, T_B, R_A, T_S)$.

Step 5 . S sends $M_5 = (R_B, T_S, Auth_{SA})$ and $M_6 = (R_A, T_S, Auth_{SB})$ to A and B , respectively.

Step 6 . After receiving M_5 , A checks the freshness of T_S . If T_S is not fresh, A stops the session. Otherwise, A checks if the $Auth_{SA}$ equals to $H_3(ID_A, ID_B, R_A, R'_A, T_A, R_B, T_S)$. If they are equal, then A confirms that B has been authenticated by S , A computes $k_{AB} = aR_B$ and the session key $sk_{AB} = H_4(ID_A, ID_B, R_A, R_B, k_{AB})$. Otherwise, A rejects the transaction.

Step 7 . After receiving M_6 , B checks the freshness of T_S . If T_S is not fresh, A stops the session. Otherwise, B checks if the $Auth_{SB}$ equals to $H_3(ID_B, ID_A, R_B, R'_B, T_B, R_A, T_S)$. If they are equal, then B confirms that A has been authenticated by S , B computes $k_{BA} = bR_A$ and the session key $sk_{BA} = H_4(ID_A, ID_B, R_A, R_B, k_{BA})$. Otherwise, B rejects the transaction.

We illustrate Round 2 of this phase in Fig. 3.

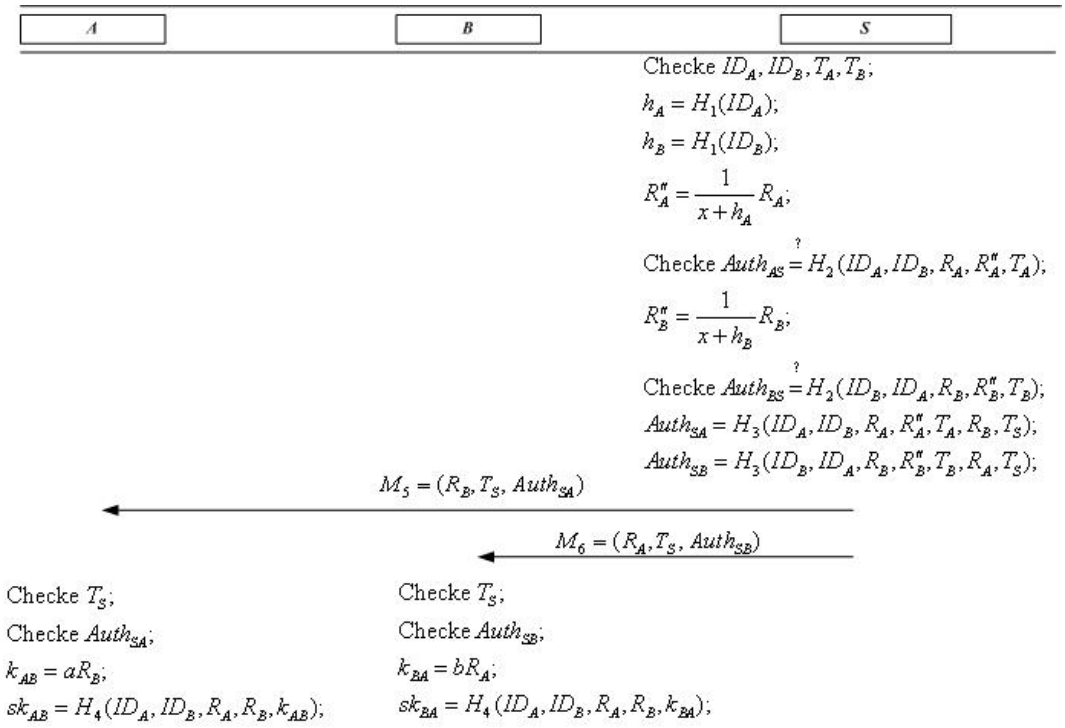

Fig. 3. Round 3 of the authenticated key exchange phase

The protocol is correct because:

$$R''_A = \frac{1}{x+h_A} \cdot R_A = \frac{1}{x+h_A} \cdot a \cdot P = a \cdot \frac{1}{x+h_A} \cdot P = a \cdot D_A = R'_A \tag{5}$$

$$R''_B = \frac{1}{x+h_B} \cdot R_B = \frac{1}{x+h_B} \cdot b \cdot P = b \cdot \frac{1}{x+h_B} \cdot P = b \cdot D_B = R'_B \tag{6}$$

and

$$k_{AB} = aR_B = abP = baP = bR_A = k_{BA} \tag{7}$$

Thus the agreed session key for A and B can be computed as:

$$\begin{aligned} sk_{AB} &= H_4(ID_A, ID_B, R_A, R_B, k_{AB}) \\ &= H_4(ID_A, ID_B, R_A, R_B, k_{BA}) = sk_{BA} \end{aligned} \tag{8}$$

4. Security analysis

The proposed protocol is based on the ECC. Thus, the security of our protocol is founded in the difficulty of solving the discrete logarithm problem(DLP). We will review some security terms needed for security analysis.

Definition 1. A secure hash function, $h: x \to y$, is one-way, if given x, it is easy to compute $h(x) = y$; however, given y, it is hard to compute $h^{-1}(y) = x$.

Definition 2. The DLP in G is as follows: Given $P \in G$ with order n and $Q \in G$. It is intractable to find r such that $Q = rP$.

Definition 3. The computational Diffie-Hellman problem (CDHP) is as follows: Given $aP, bP \in G$, it is hard to compute $abP \in G$.

Next, we will discuss the security and performance of our protocol as follows.

In the proposed protocol, from the transmitted information $R_A = aP$ and $R_B = bP$. Based on the DHP, with the random a and the private key D_A, only A can compute $k_{AB} = aR_B$, the common session key $sk_{AB} = H_4(ID_A, ID_B, R_A, R_B, k_{AB})$ and checks the validity of $Auth_{SA}$. Similarly, with the random b and the private key D_B, B can compute $k_{BA} = bR_A$, the key $sk_{BA} = H_4(ID_A, ID_B, R_A, R_B, k_{BA})$ and checks the validity of $Auth_{SB}$. On the other hand, due to the DLP, an adversary is useless to derive or guess a, b and x from R_A, R_B and P_{pub}, where $R_A = aP, R_B = bP$, and $P_{pub} = xP$. Without having the private key D_A of A, and the private key D_B of B, it is not helpful for the adversary to obtain valid login in message M_2 and M_4 for users A and B. Therefore, without knowing the owner's user private key, it is very difficult for someone to impersonate the valid user.

From the proposed method, the server S can use the master key x to check the validity of M_2 and M_4. However, without knowing one of the random numbers a or b, S cannot easily obtain the session key $k_{AB} = k_{BA} = abP$. Therefore, only the real users can derive the common key. This method can provide more secure connectivity for users A and B.

Moreover, some security properties of the proposed protocol can be showed as follows:

Replay attack: In replay attacks, an attacker may want to pretend to be A by replaying $M_2 = (ID_A, ID_B, R_A, T_A, Auth_{AS})$ to S. However, the timestamp is included in the message and the server will find the attack by checking the freshness of T_A. Then, the attacker cannot achieve authentication. Similarly, the attacker also cannot pretend to be B. In addition, the random numbers a and b are used only one time. Hence, the replay attack will fail.

Impersonation attack: If a masked server tries to deceive the requesting users A and B, it has to provide valid value $Auth_{SA} = H_3(ID_A, ID_B, R_A, R''_A, T_A, R_B, T_S)$ and

$Auth_{SB} = H_3(ID_B, ID_A, R_B, R_B'', T_B, R_A, T_S)$. However, without knowing the server's private key x, it is difficult for a masked server to compute the actual $R_A'' = \frac{1}{x + h_A} R_A$ and $R_B'' = \frac{1}{x + h_B} R_B$ from R_A and R_B. In this situation, the masked server will not be able to get invalid R_A'' and R_B'' for users A and B. Getting that far will only lead to false verification for users A and B. Hence, without knowing the server's private key x, it is very difficult for someone to impersonate the trusted server. Moreover, without having the random numbers a and b, it is not helpful for the adversary to obtain and confirm the valid common key $k_{AB} = k_{BA} = abP$ for users A and B. At the same way, we can get that it is very difficult for someone to impersonate the valid user without the knowledge of user's private key.

Man-in-the-middle attack: In 'man-in-the-middle' attack, if a malicious user C tries to impersonate user B and then communicate with A, then C has to send the valid message $M_4 = (ID_B, ID_A, R_B, T_B, Auth_{BS})$ to the serve S, where $R_B = bP$ $R_B' = bD_B$ and $Auth_{BS} = H_2(ID_B, ID_A, R_B, R_B', T_B)$. However, without the knowledge D_B of B, it is very difficult for C to provide the real value of $Auth_{BS}$. Getting that far will only lead to false verification for the server. Hence, the 'man-in-the-middle' attack can be resisted. At the same way, we can get that it is very difficult for someone to impersonate the server without the knowledge of the master private key.

Known session key security: A protocol is called *Known session key security*, if an adversary, having obtained some previous session keys, cannot get the session keys of the current run of the key agreement protocol. In our protocol, the agreed session key relies on the one-way hash function and session secrets. The output of hash function is distributed uniformly in $\{0,1\}^k$, thus one session key which is the output of hash function has no relation with the others. Besides, the session key is generated with the session secrets which are computed from the random ephemeral key, thus even one session's session secrets are revealed, the other session secrets will still remain safe.

Perfect forward secrecy: A protocol is called *Perfect forward secrecy*, if compromise of the three private keys of the participating entities does not affect the security of the previous session keys. Two aspects are related to this notion,

i.e., perfect forward secrecy (p-FS) and master key forward secrecy (m-FS). p-FS means that the compromise of both A's and B's long-term private keys would not affect the secrecy of the previously established session keys. m-FS is satisfied if the session key secrecy still holds even when the server's master key is compromised. Our protocol satisfies both p-FS and m-FS by using $H_4(ID_A, ID_B, R_A, R_B, k_{AB})$ or $H_4(ID_A, ID_B, R_A, R_B, k_{BA})$ as the shared secret. If user's private keys, or server's master key are compromised, the adversary can not compute k_{AB} or k_{BA} from R_A and R_B since he has to solve the *CDHP*, thus satisfying both p-FS and m-FS.

Key-Compromise Impersonation Resilience. Suppose that A's long-time private key D_A is disclosed to an adversary who wants to impersonate B to A. However, the adversary can only replay since she does not know D_B and hence cannot generate a valid message $M_4 = (ID_B, ID_A, R_B, T_B, Auth_{BS})$ for B which would pass the verification process. Then, we claim that our protocol has the attribute of *key-compromise impersonation* resilience.

Unknown Key-Share Resilience. To implement such an attack on the proposed protocol, the adversary is required to learn the private key of some entity. In the protocol only the entity who knows both the ephemeral secret key a and the long-term private key D_A can generate valid $(ID_A, ID_B, R_A, T_A, Auth_{AS})$. Thus our protocol offers *unknown key-share resilience*.

No key control: From the execution of the proposed protocol, we know that the session key is the hash value of $H_4(ID_A, ID_B, R_A, R_B, k_{AB})$ or $H_4(ID_A, ID_B, R_A, R_B, k_{BA})$, where $k_{AB} = aR_B$, $k_{BA} = bR_A$ and a, b are generated by the user and the server separately. So neither entity can force the session key to a pre-selected value. Hence, we claim that our protocol has the attribute of *no key control*.

5. Performance comparison

For the convenience of evaluating the computational cost, we define some notations as follows

TG_{mul} : The time of executing a scalar multiplication operation of point.

TG_{inv} : The time of executing a modular inversion operation.

TG_h : The time of executing a one-way hash function.

TG_{sym} : The time of executing a symmetric encryption/decryption operation.

In Table 1, we summarize the performance results of the proposed 3PAKE protocol. The user is a low-power computing device while the server is regarded as a powerful device. In Table 1, we know that the client side requires only $3TG_{mul} + 3TG_h$. Furthermore, if the client side can pre-compute $R_A = aP$, $R'_A = aD_A$ of the user authentication and key exchange phase, then the client side requires only $TG_{mul} + 3TG_h$ on-line computation.

Table 1. Performance evaluation of our protocol.

	User	Server
User authentication phase (Total)	$3TG_{mul} + 3TG_h$	$2TG_{mul} + 6TG_h + 2TG_{inv}$
User authentication phase (On-line)	$TG_{mul} + 3TG_h$	$2TG_{mul} + 6TG_h + 2TG_{inv}$

Then, we compare the performance of different protocols in practical environment. All the operations are built with MIRACLE [17], a standard cryptographic library. The hardware platform is a PIV 3-GHZ processor with 512-MB memory and a Windows XP operation system. For the pairing-based protocol, to achieve the 1024-bit RSA level security, we employed the ECC group on Koblitz elliptic curve $y^2 = x^3 + ax^2 + b$ defined on $F_{2^{163}}$ with $a = 1$ and b a 163-bit random prime. The running times of different operations are listed in Table 2.

Table 2. Cryptographic operation time(in milliseconds).

	TG_{mul}	TG_{inv}	TG_h	TG_{sym}
Server	0.83	0.13	<0.0001	<0.0001
User	12.08	1.89	<0.001	<0.001

In Table 3, we demonstrate the comparisons among our protocol and the previously proposed ID-based user authentication protocols on elliptic curve cryptosystem [11, 15,] in terms of the computational costs, the execution time and security properties, where the execution times are measured using Tables 2. For simplicity, the computation costs of Table 3 do not include the certificate computations. Note that if the protocol requires certificate computations or

pairings computations, its computation costs in practice are larger than those in Table 1. However, Yang et al.'s protocol [11] suffered from the parallel attack and the impersonation attack. Tan's protocol[15] base on the PKI. Then the server in Tan's protocol has to maintain the certificates for users' public keys. When the number of users is increased, the server needs a large storage space to store users' public keys and certificates. In addition, users need additional computations to verify the other's certificate in their protocols. This causes the computation loads and the energy costs of mobile devices very high. Our protocol does not need to perform the certificate computations. Moreover, our protocol has the best performance among the related protocols on the client side. In fact, our protocol also has the best performance on the server side if considering the certificate computations. From the above descriptions, we conclude that our protocol is more efficient and practical than the related protocols for the mobile-commerce environgments.

Table 3. Comparisons among the our protocol and previously proposed protocols

	Yang et al.'s protocol[11]	Tan's protocol[15]	Our protocol
Computational cost(user)	$5TG_{mul} + 2TG_{sym}$	$4TG_{mul} + 2TG_{sym}$	$3TG_{mul} + 3TG_{h}$
Execution time(user)	60.40ms	48.32ms	36.24ms
Computational cost(Server)	$2TG_{mul} + 4TG_{sym}$	$2TG_{mul} + 4TG_{sym}$	$2TG_{mul} + 6TG_{h} + 2TG_{inv}$
Execution time(Server)	1.66ms	1.66ms	1.92ms
Certificate computations	Yes	Yes	No
Parallel attack	Yes[15]	No	No
Impersonation attack	Yes[15]	No	No

6. Conclusions

In this paper, we propose an ID-based 3PAKE protocol on ECC. Based upon ID-based concept, the proposed protocol does not require additional computations

for certificate. According to the comparisons in Section 5, the proposed protocol is more efficient and practical than the related works. In the future, we will investigate a 3PAKE protocol on ECC in multi-server environments such that it can be applied to more applications in electronic transactions.

Reference

[1]. Bellovin SM, Merrit M. Encrypted key exchange: password-based protocols secure against dictionary attacks. Proceedings of IEEE Symposium on Research in Security and Privacy Oakland, CA, 1992; 72–84.

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

[3]. M. Bellare, D. Pointcheval, P. Rogaway, Authenticated key exchange secure against dictionary attacks, Advances in Cryptology-Eurocrypt'00, pp. 139-155, 2000.

[4]. M. Hölbl, T. Welzer, B. Brumen, "Two proposed identity-based three-party authenticated key agreement protocols from pairings", Computers & Security 29,2010, pp.244-252.

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

[6]. Lee, S.W., Kim, H.S., Yoo, K.Y., Efficient verifier-based key agreement protocol for three parties without server's public key. Applied Mathematics and Computation 167 (2), 996 – 1003, 2005.

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

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

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

[10]. Schnorr, C.P., Efficient identification and signature for smart cards. in: Proceedings of CRYPTO'89 , pp.239-252, 1989.

[11]. Yang J.-H., Chin-Chen Chang, An efficient three-party authenticated key exchange protocol using elliptic curve cryptography for mobile-commerce environments, Journal of Systems and Software, 82, 1497-1520, 2009.

[12]. Miller VS. Use of elliptic curves in cryptography. In: Advances in cryptology, proceedings of CRYPTO'85, vol. 218. LNCS, Springer-Verlag; 1986. p. 417–26

[13]. Koblitz N. Elliptic curve cryptosystem. Mathematics of Computation 1987;48:203–9.

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

[15]. Tan Z., An Enhanced Three-Party Authentication Key Exchange Protocol for Mobile Commerce Environments, Journal of communications, 5(5), 436-443, 2010.

[16]. Shamir A. Identity based cryptosystems and signature protocols. In: Proceedings of CRYPTO’ 84. LNCS, Springer-Verlag; 1984. 47–53.

[17]. Shamus Software Ltd., Miracl library, http://www.shamus.ie/index.php?page=home.