Acta Scientiarum

http://www.uem.br/acta
ISSN printed: 1806-2563
ISSN on-line: 1807-8664
Doi: 10.4025/actascitechnol.v35i4.16434

Efficient identity-based authenticated multiple key exchange protocol

Yitao Chen[1] and Weiwei Han[2*]

*[1]School of Mathematics and Statistics, Wuhan University, Wuhan, China. [2]Department of Mathematics and Computer Science, Guangdong University of Business Studies, Guangzhou, China. *Author for correspondence. E-mail: hww_2006@163.com*

ABSTRACT. Authenticated multiple key exchange (AMKE) protocols not only allow participants to warrant multiple session keys within one run of the protocol but also ensure the authenticity of the other party. Many AMKE protocols using bilinear pairings have been proposed. However, the relative computation cost of the pairing is approximately twenty times higher than that of the scalar multiplication over elliptic curve group. In order to improve the performance, an ID-based AMKE protocol without bilinear pairing is proposed. Since the running time is largely saved, suggested protocol is more practical than the previous related protocols for practical applications.

Keywords: ID-based, authenticated multiple key exchange, elliptic curve, bilinear pairing, random oracle.

Protocolo eficiente de troca autenticada entre múltiplas chaves com base em indentidade

RESUMO. Os protocolos de troca autenticada entre múltiplas chaves (TAMC) não só permitem que os participantes concordem com o uso de várias chaves de sessão aquando duma execução da sessão através de um único protocolo mas também assegura a autenticidade da outra parte. Muitos dos protocolos TAMC que utilizam emparelhamentos bilineares têm sido propostos. Contudo, os custos relativos de computação do emparalheamento é aproximadamente vinte vezes superior aos de multiplicação escalar sobre o grupo de curva elíptica. De modo a melhorar o desempenho, nós propomos um protocoloTAMC com base em IDs destituído de emparelhamentos bilineares. Com o tempo de execução a ser econonomizado de forma elevada, o nosso protocolo é mais prático do que outros protocolos de objectivo de aplicação prática.

Palavras-chave: Base em IDs, troca autenticada entre múltiplas chaves, curva elíptica, emparelhamento bilinear, oráculo aleatório.

Introduction

A key exchange protocol allows two entities to share a key which may be used to provide secure communication between them. Additionally, key exchange protocol should provide an authentication mechanism in order to ensure that the key is only shared between two entities. An authenticated key exchange protocol plays an important role in many modern network-based applications such as collaborative or distributed applications. The Diffie-Hellman protocol (DIFFIE; HELLMAN, 1976) is the first key exchange protocol based on asymmetric cryptography. Since that event, many key exchange protocols (CAO; KOU, 2011; CHEN et al., 2007; HE et al., 2011a and c; HE, 2012; HE et al., 2012a and b) have been proposed to satisfy the application´s requirement.

It is sometimes necessary for each communication party to produce several session keys within one run of an authenticated key exchange protocol. Harn and Lin (2001) proposed the first authenticated multiple key exchange (AMKE) protocol in which two parties generate four shared keys at a time. However, only three of these keys may provide perfect forward secrecy. Nevertheless, their protocol was broken by Lee and Wu (2004) by the modification attack. Recently, Lee et al. (2008) proposed two AMKE protocols: one is based on elliptic curve cryptography (KOBLITZ, 1987) and the other on bilinear pairings. Unfortunately, Vo et al. (2010) showed that both of Lee et al.'s protoclol and Lee and Wu's protocol are insecure against impersonation attacks and cannot provide perfect forward secrecy. Vo et al. (2010) also proposed an improved protocol.

The above AMKE protocols (HARN; LIN, 2001; LEE; WU, 2004; LEE et al., 2008; VO et al., 2010) are based on the traditional public key cryptography (PKC). Consequently, the system needs a public key infrastructure (PKI) to maintain certificates for users' public keys. Tan (2011) proposed an ID-based AMKE protocol on ECC to solve the above problem. 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, users do not need to perform additional computations to verify the corresponding certificates. Moreover, the system does not need to maintain a large public-key table.

Although the elliptic curve pairings are used in the above ID-based AMKE protocol (TAN, 2011), the pairing is still regarded as an expensive cryptography primitive. The relative computation cost of a pairing is approximately twenty times higher than that of the scalar multiplication over elliptic curve group (CAO; KOU, 2011; CHEN et al., 2007). Therefore, ID-based AMKE protocol without bilinear pairing would be more appealing in terms of efficiency. This paper provides an ID-based AMKE protocol without pairing. The protocol rests on the hardness of the discrete logarithm problem (DLP) and the computational Diffie and Hellman (1976) problem (CDHP). With the pairing-free realization, the protocol's overhead is lower than that of previous protocols in the computation and communication.

The remainder of the paper is organized as follows. Section 2 describes the preliminaries. In Section 3, our efficient ID-based AMKE protocol is proposed. The security analysis of the proposed protocol is presented in Section 4 and performance analysis is dealt with in Section 5. Conclusions are given in Section 6.

Preliminaries

Notations

Common notations used in current paper are listed as follows.

- p,n: two large prime numbers;
- F_p: a finite field;
- E: an elliptic curve defined on finite field F_p with prime order n;
- G: the group of elliptic curve points on E;
- P: a point on elliptic curve E with order n;
- $H_1(\cdot)$: a secure one-way hash function, where $H_1: \{0,1\}^* \times G \to Z_n^*$;
- $H_2(\cdot)$: a secure one-way hash function, where $H_2: \{0,1\}^* \times \{0,1\}^* \times G \times G \times G \to Z_n^*$;
- $H_3(\cdot)$: a secure one-way hash function, where $H_3: \{0,1\}^* \times G \times G \times G \times \{0,1\}^* \times G \times G \times G \times G \to Z_n^*$;
- U: a user;
- ID_U: the identity of the client U;
- (x,P_{pub}): the Key Generation Centre (KGC)'s private/public key pair, where $P_{pub} = xP$.

Background of elliptic curve cryptograph

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

$$y^2 = x^3 + ax + b, \quad a,b \in F_p \tag{1}$$

and with the discriminant

$$\Delta = 4a^3 + 27b^2 \neq 0 \tag{2}$$

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

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

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 a point such that l' intersects E/F_p at R and O and P '+' Q. Scalar multiplication over E/F_p are computed as follows:

$$tP = P + P + \ldots + P(t\ times) \tag{4}$$

Definition 1. DLP in G is as follows: given $P \in G$ with order n and $Q \in G$. It is intractable to find a such that $Q = aP$.

Definition 2. The computational Diffie and Hellman (1976) problem (CDHP) is as follows: given $aP,bP \in G$, it is hard to compute $abP \in G$.

The proposed protocol

In this section, we will present an ID-based AMKE protocol without pairing. It is composed of three phases: '*Setup*', 'KeyExtract' and 'KeyExchange' and involves a trusted Key Generation Centre (KGC), an initiator set and a responder set. In the following, A represents an initiator with identity ID_A and B is a responder with identity ID_B.

Setup phase

In this phase, KGC generates the parameter of the system.

Step 1. KGC chooses an elliptic curve equation E.

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

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

Step 4. KGC chooses four secure hash functions $H_1 : \{0,1\}^* \to Z_n^*$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times G \to Z_n^*$, and $H_3 : \{0,1\}^* \times G \times G \times G \times \{0,1\}^* \times G \times G \times G \to Z_n^*$. KGC keeps x secretly and publishes (F_p, E, n, P, P_{pub}, H_1, H_2, H_3) as system paramters.

Key extract phase

When a user U with the identity ID_U wants to register to KGC, U submits its identity ID_U to KGC. When KGC receives user's identity, KGC generates a random number r_U, compute $R_U = r_U P$, $h_U = H_1(ID_U, R_U)$ and $d_U = r_U + h_U x \in G$ and then delivers the tuple (d_U, R_U) to U.

To deliver the private key the tuple (d_U, R_U) to user U, two approaches may be used and both require to be performed only once. One is off-line approach that KGC stores the identity ID_U, the tuple (d_U,T_U) into a smartcard and returns it to the user U. The other one is on-line approach. When the user U connects to KGC through Internet, KGC may use the Secure Socket Layer (SSL) channel in the https mode to deliver the tuple (d_U,R_U) to the user U. U may validate his/her private key by checking whether the equation $d_U \cdot P = R_U + h_U \cdot P_{pub}$ holds. The private key is valid if the equation holds and vice versa.

Key exchange phase

In this phase, two users A and B, with identity ID_A and ID_B separatly, authenticate each other with S's help; A and B can thus share a multiple session keys. This phase, as shown in Figure 1, is described as follows.

Step 1. A generates two random numbers $a, a' \in Z_n^*$ to compute $X_A = aP$, $X'_A = a'P$, $l_A = H_2(ID_A, ID_B, R_A, X_A, X'_A)$ and $s_A = (a + l_A)^{-1} d_A \bmod n$. Then, A sends the message $M_1 = (ID_A, ID_B, R_A, X_A, X'_A, s_A)$ to B.

Step 2. After receiving M_1, B computes $l_A = H_2(ID_A, ID_B, R_A, X_A, X'_A)$, $h_A = H_1(ID_A, R_A)$ and checks whether the equation $s_A(X_A + l_A P) = R_A + h_A P_{pub}$ holds. If the equation does not hold, B stops the session. Otherwise, B generates two random numbers $b, b' \in Z_n^*$ to compute $Y_B = bP$, $Y'_B = b'P$, $l_B = H_2(ID_A, ID_B, R_B, Y_B, Y'_B)$ and $S_B = (b + l_B)^{-1} d_B \bmod n$. Then B sends the message $M_2 = (ID_A, ID_B, R_B, Y_B, Y'_B, s_B)$ to A.

Step 3. After receiving M_2, A computes $l_B = H_2(ID_A, ID_B, R_B, Y_B, Y'_B)$, $h_B = H_1(ID_B, R_B)$ and checks whether the equation $s_B(Y_B + l_B P) = R_B + h_B P_{pub}$ holds. If the equation does not hold, A stops the session.

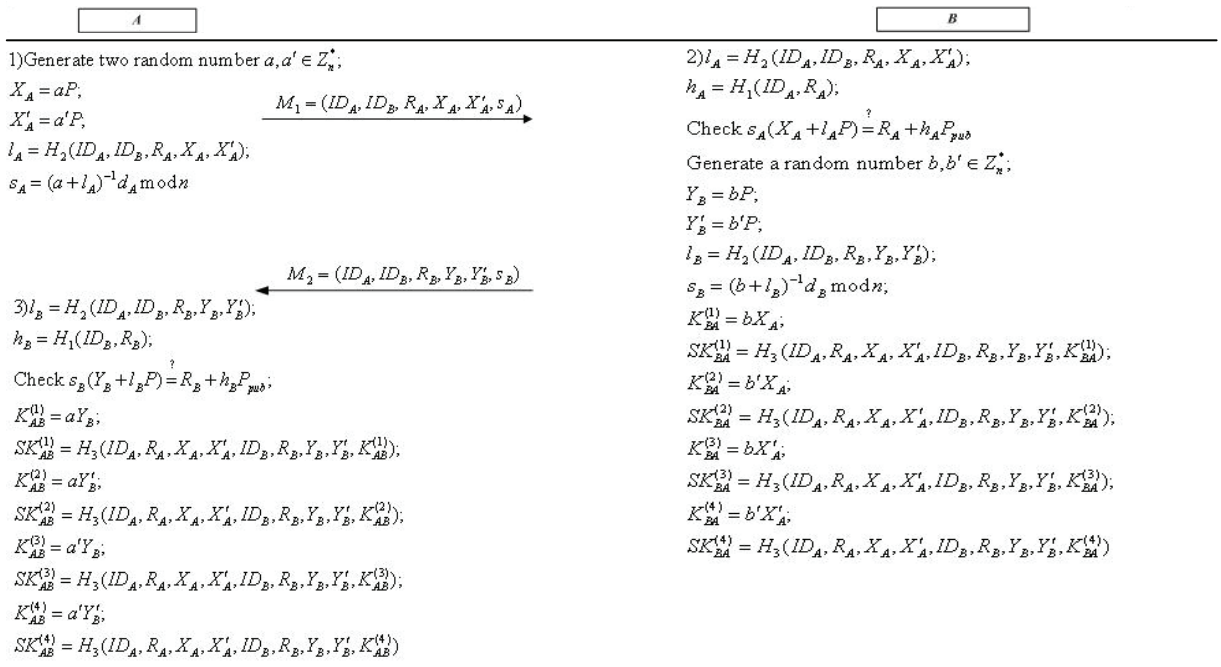

Figure 1. Key exchange phase.

Then both A and B can compute the shared secrets as follows.

A computes

$$K_{AB}^{(1)} = aY_B, SK_{AB}^{(1)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(1)}) \quad (5)$$

$$K_{AB}^{(2)} = aY'_B, SK_{AB}^{(2)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(2)}) \quad (6)$$

$$K_{AB}^{(3)} = a'Y_B, SK_{AB}^{(3)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(3)}) \quad (7)$$

$$K_{AB}^{(4)} = a'Y'_B, SK_{AB}^{(4)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(4)}) \quad (8)$$

B computes

$$K_{BA}^{(1)} = bX_A, SK_{BA}^{(1)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(1)}) \quad (9)$$

$$K_{BA}^{(2)} = b'X_A, SK_{BA}^{(2)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(2)}) \quad (10)$$

$$K_{BA}^{(3)} = bX'_A, SK_{BA}^{(3)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(3)}) \quad (11)$$

$$K_{BA}^{(4)} = b'X'_A, SK_{BA}^{(4)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(4)}) \quad (12)$$

The protocol is correct because:

$$s_A(X_A + l_A P) = (a + l_A)^{-1} d_A(X_A + l_A P) = (a + l_A)^{-1} d_A(aP + l_A P) = d_A P = (r_A + h_A x)P = R_A + h_A P_{pub} \quad (13)$$

$$s_B(Y_B + l_B P) = (b + l_B)^{-1} d_B(Y_B + l_B P) = (a + l_B)^{-1} d_B(aP + l_B P) = d_B P = (r_B + h_B x)P = R_B + h_B P_{pub} \quad (14)$$

$$K_{AB}^{(1)} = aY_B = abP = baP = bX_A = K_{BA}^{(1)} \quad (15)$$

$$K_{AB}^{(2)} = aY'_B = ab'P = b'aP = b'X_A = K_{BA}^{(2)} \quad (16)$$

$$K_{AB}^{(3)} = a'Y_B = a'bP = ba'P = bX'_A = K_{BA}^{(3)} \quad (17)$$

and

$$K_{AB}^{(4)} = a'Y'_B = a'b'P = b'a'P = b'X'_A = K_{BA}^{(4)} \quad (18)$$

Thus the multiple session keys for A and B may be computed as:

$$SK_{AB}^{(1)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(1)}) = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(1)}) = SK_{BA}^{(1)} \quad (19)$$

$$SK_{AB}^{(2)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(2)}) = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(2)}) = SK_{BA}^{(2)} \quad (20)$$

$$SK_{AB}^{(3)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(3)}) = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(1)}) = SK_{BA}^{(3)} \quad (21)$$

and

$$SK_{AB}^{(4)} = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{AB}^{(4)}) = H_3(ID_A, R_A, X_A, X'_A, ID_B, R_B, Y_B, Y'_B, K_{BA}^{(4)}) = SK_{BA}^{(4)} \quad (22)$$

Security analysis

Security model for CTAKA protocols

In this section, we present a security model for AMKE protocols based on Cao and Kou's work (CAO; KOU, 2011). The model is defined by the following game between a challenger C and an adversary A. In the model, A is modeled by a probabilistic polynomial time-turning machine. All communications go through the adversary A. Participants only respond to the queries by A and do not communicate directly among themselves. A can relay, modify, delay, interleave or delete all the message flows in the system. Note that A may act as a benign adversary, which means that A is deterministic and restricts his/her action to choosing a pair of oracles $\prod_{i,j}^{m}$ and $\prod_{j,i}^{n}$ and then faithfully conveying each message flow from one oracle to the other. Furthermore, A may ask a polynomially-bounded number of the following queries, as below.

Create (ID_i): This allows A to ask C to set up a new participant i with identity ID_i. On receiving such a query, C generates the private key for i.

Corrupt (ID_i): A requests the private key of a participant i whose identity is ID_i. To respond, C outputs the private key of participant i.

Send($\prod_{i,j}^{m}, M$): A may send a message M of his/her choice to an oracle, say $\prod_{i,j}^{n}$, in which case participant i assumes that the message has been sent by participant j. A may also make a special *Send* query with $M \neq \lambda$ to an oracle $\prod_{i,j}^{n}$, which instructs i to initiate a protocol run with j. The oracle is an initiator oracle if the first message it has received is λ. If the oracle does not receive message λ as its first message, then it is a responder oracle.

Reveal($\prod_{i,j}^{m}$): A can ask a particular oracle to reveal the session keys, it currently holds, to A.

Test($\prod_{i,j}^{m}, k$): At some point, A may choose one of the oracles, say $\prod_{i,j}^{m}$, to ask a single *Test* query. This oracle must be fresh. To answer the query, the oracle flips a fair coin $b_k \in \{0,1\}$, and returns the k^{th} session key held by $\prod_{i,j}^{m}$ if $b_k = 0$, or a random

sample from the distribution of the session key if $b_k = 1$, where $k = 1,2,3,4$.

After a *Test* query, the adversary may continue to query the oracles; however, it cannot make a *Reveal* query to the test oracle $\Pi_{i,j}^m$ or to $\Pi_{j,i}^n$, which is matching with $\Pi_{i,j}^m$, and it cannot corrupt participant *j*.

At the end of the game, A must output four guess bits b_k', where $k = 1,2,3,4$. A wins if and only if $b_k' = b_k$ for at least k, $k = 1,2,3,4$. A's advantage to win the above game, denoted by $Advantage^A(k)$, is defined as:

$$Advantage^A(k) = \max\{\left|\Pr[b_k' = b_k] - \frac{1}{2}\right|, k = 1,2,3,4\}.$$

The the security of AMKE protocol is defined as follows.

Definition 3. An authenticated multiple key exchange protocol is secure if

(1) in the presence of a benign adversary on $\Pi_{i,j}^m$ and $\Pi_{j,i}^n$, both oracles always agree on the same session key, and these keys are distributed uniformly at random.

(2) for any adversary, the probability of generating a legal message M_1 or M_2 is negligible.

(3) For any adversary, $Advantage^A(k)$ is negligible.

Security analysis of our protocol

To prove the security of our protocol in the random oracle model, we treat H_1, H_2 and H_3 as three random oracles [HE et al., 1013a and b] using the above model. For security, the following lemmas and theorems are provided.

Lemma 1. If two oracles are matching, both will be accepted and will get the same session key which is distributed uniformly at random in the session key sample space.

Proof. From the correction analysis of our protocol in section 3, we know if two oracles are matching, then both are accepted and have the same four session keys. The session keys are distributed uniformly since *a*, *a'*, *b* and *b'* are selected uniformly during the protocol execution.

Lemma 2. Assuming that the DLP is intractable, the probability of generating a legal message M_1 or M_2 is negligible in the random oracle model for any adversary.

Proof. Suppose that there is an adversary A who may generate a legal message M_1 with non-negligible probability ε. Then, we may use the ability of A to construct an algorithm C solving the DLP.

Suppose C is challenged with a DLP instance (*P*,*Q*) and is tasked to compute $x \in Z_n^*$ satisfying $Q = x \cdot P$. Let q_c, q_h, q_s be the total number of *Create* queries, hash queries and *Send* queries. To achieve the above, C picks two identities ID_I and ID_J at random as the challenged *ID* in this game, and gives $\{F_p, E/F_p, G, P, P_{pub} = Q, H_1, H_2, H_3\}$ to A as the public parameter. Then C answers A 's queries as follows.

Create (*ID*): C maintains a hash list L_C of tuple $(ID, R_{ID}, d_{ID}, h_{ID})$. If *ID* is on L_C, then C responses with $(ID, R_{ID}, d_{ID}, h_{ID})$. Otherwise, C simulates the oracle as follows. It chooses $a, b, c \in Z_n^*$ at random, sets $R_{ID} = a \cdot P_{pub} + b \cdot P$, $d_{ID} = b$, $h_{ID} = H_1(ID, R_{ID}) \leftarrow -a \bmod n$, responses with $(ID, R_{ID}, d_{ID}, h_{ID})$, inserts (ID, R_{ID}, h_{ID}) into L_{H_1}. Note that $(ID, R_{ID}, d_{ID}, h_{ID})$ generated in this way satisfies the equation $d_{ID} \cdot P = R_{ID} + h_{ID} \cdot P_{pub}$ in the partial private key extraction algorithm. It is a valid secret key.

$H_1 - query$: C maintains a hash list L_{H_1} of tuple (ID, R_{ID}, h_{ID}), as explained below. The list is initially empty. When A makes a hash oracle query on *ID*, if the query *ID* has already appeared on L_{H_1}, then the previously defined value is returned. Otherwise, C queries *Create* (*ID*), gets $(ID, R_{ID}, d_{ID}, h_{ID})$ and responses with h_{ID}.

$H_2 - query$: C maintains a hash list L_{H_2} of tuple $(ID_i, ID_j, R_i, X_i, X_i', h)$. When A makes H_2 queries on the message $(ID_i, ID_j, R_i, X_i, X_i')$, C chooses a random value $h \in Z_n^*$, sets $h = H_2(ID_i, ID_j, R_i, X_i, X_i')$, adds $(ID_i, ID_j, R_i, X_i, X_i', h)$ to L_{H_2}, and sends h to A.

$H_3 - query$: C maintains a hash list L_{H_3} of tuple $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)}, h)$. When A makes H_3 queries on the message $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)}{}_i)$, C chooses a

random value $h \in Z_n^*$, sets $h = H_3(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y', K_{ij}^{(s)})$, adds $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y', K_{ij}^{(s)}, h)$ to L_{H_3}, and sends h to A.

Send$(\prod_{i,j}^m, M)$: C answers A's *Send* queries as the description of the protocol, since C knows all users' private keys.

Corrupt(ID): Whenever C receives this query, if $ID = ID_I$, C aborts; otherwise, C searches for a tuple $(ID, R_{ID}, d_{ID}, h_{ID})$ in L_C which is indexed by *ID* and returns d_{ID} as the answer.

Test$(\prod_{i,j}^m, k)$: C answers A's *Test* queries as the description of the game in the security model.

Finally, A stops and outputs a legal message $M_1^{(1)} = (ID_I, ID_J, R_I, X_I, X_I', s_I^{(1)})$ for the initiator's identity ID_i and the responder's identity *IDj* with non-negligible probability ε. If $ID_i \neq ID_I$ or $ID_j \neq ID_J$, C halts it.

From the forgery lemma (DAVID; JACQUE, 2000), if we have a replay of C with the same random tape but different choice of H_2, A will output another valid message $M_1^{(t)} = (ID_I, ID_J, R_I, X_I, X_I', s_I^{(t)})$, where $t = 2,3$. Then we have

$$s_I^{(t)} \cdot (X_I + l_I^{(t)} \cdot P) = R_{ID_I} + h_{ID_I} P_{pub}, t = 1,2,3 \quad (23)$$

By a, r_{ID_I}, x, we now denote discrete logarithms of X_I, R_{ID_I} and P_{pub} respectively, i.e., $X_I = aP$, $R_{ID_I} = r_{ID_I} P$ and $P_{pub} = xP$.

$$s_I^{(t)} \cdot (a + l_I^{(t)}) = r_{ID_I} + h_{ID_I} x, t = 1,2,3 \quad (24)$$

In these equations, only a, r_{ID_I}, x are unknown to C. C solves these values from the above three linear independent equations, and outputs x as the solution of the DLP.

Since $ID_i = ID_I$ and $ID_j = ID_J$ with the probability $\frac{1}{q_c(q_c - 1)}$, then C can solve the DLP with the probability $\frac{1}{q_c(q_c - 1)}\varepsilon$.

Through the same method, we can prove that no adversary can generate a legal message M_2.

Lemma 3. Assuming that the CDHP is intractable, the advantage of an adversary against our protocol is negligible in the random oracle model.

Proof. Suppose that an adversary A can win the game described in the security model with non-negligible probability ε. Then, we can use the ability of A to construct an algorithm C solving the CDHP.

Let q_c, q_h, q_s be the total number of *Create* queries, hash queries and *Send* queries. Suppose C is challenged with a CDHP instance $(P_1 = aP, P_2 = bP)$ and is tasked to compute abP, where C does know the value of a, b. To do so, C picks $x \in Z_n^*$, $l_1, l_2 \in \{1, \ldots, q_s\}$, two identities ID_I and ID_J at random as the challenged *ID* in this game, and gives the system parameters $\{F_p, E/F_p, G, P, P_{pub} = xP, H_1, H_2, H_3\}$ to A as the public parameters. Let q_c, q_h, q_s be the total number of *Create* queries, hash queries and *Send* queries separately. Then C answers A 's queries as follows.

Create(ID): C maintains a hash list L_C of tuple $(ID, R_{ID}, d_{ID}, h_{ID})$. If ID is on L_C, then C responses with $(ID, R_{ID}, d_{ID}, h_{ID})$. Otherwise, C simulates the oracle as follows. It chooses $r_{ID}, h \in Z_n^*$ at random, sets $R_{ID} = r_{ID} P$, $h_{ID} = H_1(ID, R_{ID}) \leftarrow h$, $d_{ID} = r_{ID} + h_{ID} x$ responses with $(ID, R_{ID}, d_{ID}, h_{ID})$, inserts (ID, R_{ID}, h_{ID}) into L_{H_1}.

$H_1 - query$: C maintains a hash list L_{H_1} of tuple (ID, R_{ID}, h_{ID}), as explained below. The list is initially empty. When A makes a hash oracle query on *ID*, if the query *ID* has already appeared on L_{H_1}, then the previously defined value is returned. Otherwise, C queries *Create*(ID), gets $(ID, R_{ID}, d_{ID}, h_{ID})$ and responses with h_{ID}.

$H_2 - query$: C maintains a hash list L_{H_2} of tuple $(ID_i, ID_j, R_i, X_i, X_i', h)$. When A makes H_2 queries on the message $(ID_i, ID_j, R_i, X_i, X_i')$, C chooses a random value $h \in Z_n^*$, sets $h = H_2(ID_i, ID_j, R_i, X_i, X_i')$, adds $(ID_i, ID_j, R_i, X_i, X_i', h)$ to L_{H_2}, and sends h to A.

$H_3 - query$: C maintains a hash list L_{H_3} of tuple $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)}, h)$. When A makes

H_3 queries on the message $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)}{}_i)$, C chooses a random value $h \in Z_n^*$, sets $h = H_3(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)})$, adds $(ID_i, R_i, X_i, X_i', ID_j, R_j, Y_j, Y_j', K_{ij}^{(s)}, h)$ to L_{H_3}, and sends h to A.

$Send(\prod_{i,j}^m, M)$: C answers A 's *Send* queries as follows

- If $\prod_{i,j}^m = \prod_{I,J}^{l_1}$ and $M = \lambda$, C generates two random number $r_I, r_I' \in Z_n^*$, computes $X_I = r_I P_1$, $X_I' = r_I' P_1$ and defines $s_I \leftarrow s_I(X_I + l_I P) = R_I + h_I P_{pub}$. Then C responds with $M_1 = (ID_I, ID_J, R_I, X_I, X_I', s_I)$.

- Or, if $\prod_{i,j}^m = \prod_{J,I}^{l_2}$ and $M = M_1$, C checks whether the equation $s_I(X_I + l_I P) = R_I + h_I P_{pub}$ holds. If the equation does not hold, C stops the session. Otherwise, C generates two random numbers $r_J, r_J' \in Z_n^*$, computes $Y_J = r_J P_2$, $Y_J' = r_J' P_2$ and defines $s_J \leftarrow s_J(Y_J + l_J P) = R_J + h_J P_{pub}$. Then C responds with $M_2 = (ID_I, ID_J, R_J, Y_J, Y_J', s_J)$.

- Or, if $\prod_{i,j}^m = \prod_{I,J}^{l_1}$ and $M = M_2$, C checks whether the equation $s_J(X_J + l_J P) = R_J + h_J P_{pub}$ holds. If the equation does not hold, C stops the session. Otherwise, C accepts the session.

- Or, C answers A's *Send* queries as the description of the protocol, since C knows all users' private keys.

$Corrupt(ID)$: Whenever C receives this query, if $ID = ID_I$ or $ID = ID_J$, C aborts; otherwise, C searches for a tuple $(ID, R_{ID}, d_{ID}, h_{ID})$ in L_C which is indexed by *ID* and returns d_{ID} as the answer.

$Test(\prod_{i,j}^m, k)$: C answers A's *Test* queries as the description of the game in section 4.1.

Simulation of C is perfectly indistinguishable from the proposed protocol excepting in following cases: $s_I \leftarrow s_I(X_I + l_I P) = R_I + h_I P_{pub}$ and $s_J \leftarrow s_J(Y_J + l_J P) = R_J + h_J P_{pub}$ cannot be done in the *Send* queries. Indeed, l_I and l_J is a value from H_2 and S_I and S_J are two elements from Z_n^*. This simulation with the failure probability is less than $\left(\frac{q_s q_h}{n q_c}\right)^2$.

The probability that A chooses $\prod_{I,J}^{l_1}$ as the *Test* oracle is $\frac{1}{q_c^2 q_s}$. In this case, A would not have made $Reveal(\prod_{I,J}^{l_1})$ or $Reveal(\prod_{J,I}^{l_2})$ queries, and so A would not have aborted. If A can win in such a game, then A must have made the corresponding H_3 query of the form $(ID_I, R_I, X_I, X_I', ID_J, R_J, Y_J, Y_J', K_{IJ}^{(k)})$ if $\prod_{I,J}^{l_1}$ is the initiator oracle with overwhelming probability because H_3 is a random oracle. Thus A can find the corresponding item in the H_3-list with the probability $\frac{1}{q_h}$ and output $(r_I r_J)^{-1} K_{IJ}^{(1)}$ as the solution to the CDHP when k is 1, $(r_I r_J')^{-1} K_{IJ}^{(2)}$ when k is 2, $(r_I' r_J)^{-1} K_{IJ}^{(3)}$ when k is 3, and $(r_I' r_J')^{-1} K_{IJ}^{(4)}$ when k is 4. So if the adversary computes the correct session key with non-negligible probability ε, then the probability that C solves the CDHP is $\frac{\varepsilon}{q_c^2 q_s q_h} - \left(\frac{q_s q_h}{n q_c}\right)^2$, contradicting the hardness of the CDHP.

From the above three lemmas, we may obtain the following theorem.

Theorem 1. Our protocol is a secure AMKE protocol.

Through a similar method, we may prove that our protocol provides forward secrecy property. We describe it by the following theorem.

Theorem 2. Our protocol has the perfect forward secrecy property if the CDHP is hard.

Performance comparison

In this section, we will compare the efficiency of our new protocol with two latest AMKE protocols, i.e. Vo et al.'s protocol (VO et al., 2010) and Tan's protocol (TAN, 2011). The purpose is to show the advantages of our protocol compared with existing solutions.

For the pairing-based protocol, to achieve the 1024-bit RSA level security, we have to use the Tate pairing defined over some supersingular elliptic curve on a finite field F_q, where the length of q is, at least, 512 bits (CAO; KOU, 2011; CHEN et al., 2007). For the ECC-based protocols, to achieve the same security level, we employ some secure elliptic curve on a finite field F_p or F_{2^m}, where the length of p is 160 bits, at least (CAO; KOU, 2011). Let T_S, T_{S_pair}, T_A, T_P, T_m, T_I and T_e represent one scalar multiplication,

one pairing-based scalar multiplication, one point addition, one pairing computation, one modular multiplication, one modular inversion and one modular exponent, respectively. Table 1 shows the results of the performance comparison.

From the theoretical analysis (CHEN et al., 2007) and the experimental result (CAO; KOU, 2011; HE et al., 2011a and b), we know the relative computation cost of the bilinear pairing operation, the modular exponentiation and the pairing-based scale multiplication are at least 19, 3 and 3 times that of the scalar multiplication, separately. Then we may conclude the total computational cost of our protocol is 5.72% that of Vo et al. (2010)'s protocol, and 11.49% of Tan's protocol. Assuming that the identities *ID* are 128-bit long, then the communication cost of our protocol is 29.71% of Vo et al. (2010) protocol, and 38.89% of Tan's protocol. Thus our protocol is more useful and efficient than the previous ones.

Table 1. Performance comparison.

	Vo et al.'s protocol	Tan's protocol	Our protocol
Short-term public keys	$2T_{s\_pair}$	$3T_{s\_pair}$	$2T_s$
Authentication part	$T_A + 2T_{s\_pair} + 2T_m$	$2T_{s\_pair} + T_A$	$T_s + T_l + T_m$
Verification	$T_A + 2T_{s\_pair} + T_P$	$3T_P + T_{s\_pair}$	$3T_s + 2T_A$
Generation of one key	$T_A + 3T_{s\_pair} + T_P$	T_{s_pair}	T_s
Generation of four keys	$4T_A + 8T_{s\_pair} + 4T_P$	$4T_{s\_pair}$	$4T_s$
Total computation time	$6T_A + 14T_{s\_pair} + 2T_m + 7T_P$	$T_A + 10T_{s\_pair} + 3T_P$	$2T_A + 9T_S + T_l + T_m$
Communitcation cost	3016bits	2304bits	896bits

Conclusion

We proposed in current paper a new ID-based AMKE protocol using the elliptic curve. Under the random oracle model, we showed that the proposed protocol is secure. According to the comparisons in Section 5, the proposed protocol is more efficient and practical than those of related works. In the future, we will investigate the ID-based AMKE without pairings operations in standard model such that it may be applied to more applications.

References

CAO, X.; KOU, W. A pairing-free identity-based Authenticated key agreement protocol with minimal message exchanges. **Information Sciences**, v. 180, n. 15, p. 2895-2903, 2011.

CHEN, L.; CHENG, Z.; SMART, N. Identity-based key agreement protocols from pairings. **Internal Journal of Information Security**, v. 6, n. 4, p. 213-241, 2007.

DAVID, P.; JACQUE, S. Security arguments for digital signatures and blind signatures. **Journal of Cryptology**, v. 13, n. 3. p. 361-396, 2000.

DIFFIE, W.; HELLMAN, M. New directions in cryptography. **IEEE Transactions Information Theory**, v. IT-22, n. 6, p. 644-654, 1976.

HARN, L.; LIN, H. Authenticated key agreement without using one-way hash function. **Electronic Letter**, v. 37, n. 10, p. 29-30, 2001.

HE, D. Cryptanalysis of an authenticated key agreement protocol for wireless mobile communications. **ETRI Journal**, v. 34, n. 3, p. 482-484, 2012.

HE, D.; CHEN, J.; HU, J. An authentication key agreement protocol using isogenies between elliptic curves. **International Journal of Computers Communications and Control**, v. 6, n. 2, p. 251-258, 2011a.

HE, D.; CHEN, J.; HU, J. An ID-based proxy signature protocols without bilinear pairings. **Annals of Telecommunications**, v. 66, n. 11-12, p. 657-662, 2011b.

HE, D.; CHEN, Y.; CHEN, J.; ZHANG, R. A new two-round certificateless authenticated key agreement protocol without bilinear pairings. **Mathematical and Computer Modelling**, v. 54, n. 11-12, p. 3143-3152, 2011c.

HE, D.; CHEN, J.; HU, J. A pairing-free certificate-less authenticated key agreement protocol. **International Journal of Communication Systems**, v. 25, n. 2, p. 221-230, 2012a.

HE, D.; PADHYE, S.; CHEN, J. An efficient certificateless two-party authenticated key agreement protocol. **Computers and Mathematics with Applications**, v. 64, n. 6, p. 1914-1926, 2012b.

HE, D.; CHEN, Y.; CHEN, J. A provably secure certificateless proxy signature scheme without pairings, **Mathematical and Computer Modelling**, v. 57, n. 9-10, p. 2510-2518, 2013a.

HE, D.; HUANG, B.; CHEN, J. New certificateless short signature scheme. **IET Information Security**, v. 7, n. 2, p. 113-117, 2013b.

KOBLITZ, N. Elliptic curve cryptosystem. **Mathematics of Computation**, v. 48, n. 177, p. 203-209, 1987.

LEE, N.; WU, C. Improved authentication key exchange protocol without using one-way hash function. **ACM Operation System Review**, v. 38, n. 2, p. 85-92, 2004.

LEE, N.; WU, C.; WANG, C. Authenticated multiple key exchange protocols based on elliptic curves and bilinear pairings. **Computers and Electrical Engineering**, v. 34, n. 1, p. 12-20, 2008.

TAN, Z. Efficient identity-based authenticated multiple key exchange protocol. **Computers and Electrical Engineering**, v. 37, n. 2, p. 191-198, 2011.

VO, D.; LEE, H.; YEUN, C.; KIM, K. Enhancements of authenticated multiple key exchange protocol based on bilinear pairings. **Computers and Electrical Engineering**, v. 36, n. 1, p. 155-159, 2010.

Received on March 21, 2012.
Accepted on October 4, 2012.