私立東海大學資訊科學與工程研究所

碩士論文

指導教授：林祝興 教授

A Study of Authenticated Key Agreement Protocols Based on Elliptic Curve Cryptography

研 究 生 ： 黃 國 榮

中 華 民 國 九 十 二 年 六 月

摘要

在本論文中，我們設計了兩個具有確認身份的金鑰交換方法，並且利用橢圓曲線密碼學來產生使用者的金鑰。傳統的 Diffie-Hellman 鑰匙交換方法並無提供通訊雙方身份的驗證，因此，Seo 和 Sweneey 在 1999 年提出了共享密碼的概念來驗證通訊雙方的身份，並利用冪次方運算產生通訊鑰匙。另外，Joux 在 2000 年利用 Wail Pairing 的特性提出了三方的 Diffie-Hellman 金鑰交換協定，在 Joux 的協定中每個人只需廣播一次公開的訊息就可協議出一把共同的通訊鑰匙，但無法提供使用者的身份驗證，在 2003 年 Kyungah Shim 年為了解決 Joux 協定的問題提出了具有身份驗證的三方金鑰交換協定，Kyungah 的協定主要的概念是利用憑證來作身份的驗證，並將 Wair Pairing 運用在冪次方的運算。

在我們的方法中，第一個方法是先利用共享密碼產生認證訊息，雙方再互相驗證訊息來確認通訊雙方的身份、第二個方法則是藉由憑證來確認通訊者的身份、加強金鑰交換協定的安全度，在提出的兩個方法都加入了時戳限制認證訊息的有效時間並且透過橢圓曲線來加快運算速度，此外，在計算量方面第一個方法只需 Seo-Sweeny 協定的四分之一，第二個方法則維持與 Kyungah Shim 協定相同的安全度下，減少通訊鑰匙的計算量。最後，我們在論文中會討論這兩種協定的安全性質，並對常見的攻擊作分析。

關鍵字：確認身份金鑰交換協定、橢圓曲線密碼學、Diffie-Hellman 鑰匙交換方法、共享密碼、憑證。

Contents

List of Figures

Abstract

In this thesis, we proposed two authenticated key agreement protocols on Elliptic Curve Cryptography. The basic Diffie-Hellman protocol doesn't authenticate the communicating entities and is vulnerable to the man-in-the-middle attack. To provide authenticity to key agreement protocols, we respectively use shared-password in our first protocol and certificates to our second protocol. Besides, we applied the elliptic curve cryptography for the generation of keys to improve the efficiency. In the first protocol, the authenticated message is generated with the shared-password and the receiver can verify it with his shared-password to ascertain the sender's identify. The second protocol is one round tripartite authenticated key agreement protocol on the public key infrastructure. Each entity in the second protocol must send a message including his own signature to demonstrate that he is the owner of the certificate. To avoid an adversary intercepting the signature and resending it to others, signature of the sender includes his ephemeral public key and a short-lived timestamp. Besides, we provide the security analysis about our protocols.

Keywords: Tripartite Authenticated Key Agreement Protocol, Elliptic Curve Cryptography, the Diffie-Hellman Key Agreement Protocol, Shared-password, Certificate, Man-in-the-middle Attack.

1. Introduction

Key agreement is a process whereby two (or more) participants can establish a shared secret key (session key). In a key agreement protocol, each entity transports his information to the other entities and uses the shared information to derive a join secret key. A key agreement protocol is said to provide implicit key authentication (of B to A) if A is assured that no other entity besides B can possibly ascertain the value of the secret key. A key agreement protocol that provides mutual implicit key authentication is called an authenticated key agreement (or AK protocol).

In 1976, Diffie and Hellman [1] proposed the first key agreement protocol. The Diffie-Hellman protocol is a fundamental technique providing unauthenticated key agreement using exponentiation. Its security is based on the difficulty of calculating exponentiation in the same field. Furthermore it doesn’t offer authentication between participants and suffers from the man-in-the-middle attack. There have been many attempts to add authentication for improving the Diffie-Hellman protocol.

In 1999, Seo and Sweeney [2] proposed a simple authenticated key agreement protocol, which solves the attack on the Diffie-Hellman key agreement protocol. In 2000, Joux [3] proposed a one round protocol for tripartite Diffie-Hellman based on the Weil pairing. Joux's protocol utilizes the Weil pairing to reduce communication rounds and it takes only one round of communication to generate a common session key. Moreover, Joux's protocol suffers from the man-in-the-middle attack because it doesn't authenticate the three participants. To provide authenticity to tripartite key agreement protocol, Kyungah [4] lately proposed one round tripartite authenticated key agreement protocol based on the pairing incorporating certified public keys. The main idea is to apply certificates of three entities, which are issued by a Certificate Authority (CA), to bind an entity’s identity with his public key. Signatures of CA provide the authenticity of the public keys. This is important because only these participant who posses the key pair (public key and private key) are able to compute the session keys.

In this thesis, we propose two AK protocols that the first protocol is a pre-shred password authenticated key agreement protocol and the second is on round tripartite authenticated key agreement protocol on public key infrastructure. In both of our protocols, timestamp concept is applied to the shored-lived message for preventing straight replay attack, reflective replay attack. Further, our protocols are more efficient for generation of key with Elliptic Curve Cryptosystem.

In the second protocol, user's certificate is applied to verify user's identity for resisting man-in-the-middle attack. Besides, the exchanged message M_{AB}, where A and B respectively denote the sender and recipient of M_{AB}, includes A's timestamp, ephemeral public key and signature. A's signature consists of ID_B plus A's timestamp and ephemeral public key. Further, this assures that no one can impersonate A to resend M_{AB} to others over the period of validity. The attribution of the protocol is that if an adversary wants to fake someone, he must offer a true certificate and compromise private key of the sender. Since the modification attack and unknown key shared attack cannot work in our protocol.

Organization of the Thesis as follows: In chapter 2 we introduce the background of the related technologies used in this thesis. The authenticated key agreement protocols are described in chapter 3. Our proposed protocols are specified in chapter 4. The complexity and security analysis of our protocols are presented in chapter 5. The conclusions of our proposed protocols are in chapter 6.

2. Background

2.1. The Diffie-Hellman Key Agreement Protocol

In 1976, the Diffie-Hellman key agreement protocol was published in the ground-breaking paper "New Directions in Cryptography." The protocol allows two users to agree on a secret key over an insecure medium without any prior secrets.

The protocol has two system parameters p and g. They are both public and may be used by all the users in a system. Parameter p is a prime number and parameter g is an integer less than p, with the following property: for every number n between 1 and p-1 inclusive, there is a power k of g such that $n = g^k \bmod p$.

The description of the Diffie-Hellman key agreement protocol : A and B want to agreement on a shared secret key with the Diffie-Hellman key agreement protocol. The process is as follows:

First, A and B respectively generate a random private value a and b where both a and b are drawn from the set of integers {1, ..., p-2}. Then they derive their public values using parameters p and g and their private values. A's public value is $g^a \bmod p$ and B's public value is $g^b \bmod p$. They then exchange their public values. Finally, A computes $g^{ab} = (g^b)^a \bmod p$, and Bob computes $g^{ba} = (g^a)^b \bmod p$. Since $g^{ab} = g^{ba} = k$, A and B now have a shared secret key k.

The protocol depends on DLP (the discrete logarithm problem) for its security. Assume that it is computationally infeasible to calculate the shared secret key $k = g^{ab} \bmod p$ given the two public values $g^a \bmod p$ and $g^b \bmod p$ when the prime p is sufficiently large. Maurer has shown that breaking the Diffie-Hellman protocol is equivalent to computing discrete logarithms under certain assumptions.

2.2. The Man-in-the-middle Attack on the Diffie-Hellman Protocol

The Diffie-Hellman protocol is vulnerable to a man-in-the-middle attack because it doesn't attempt to authenticate the users. In this attack, an adversary E intercepts A's public value and resends her own public value to B. When B transmits his public value, E replaces it with her own and resends it to A. E and A agree on one shared session key and E and B agree on another. After this communication, E simply decrypts any messages, which Alice or Bob sends and reads these messages and possibly modifies them before re-encrypting with the appropriate key and transmits them to the other party.

2.3. Diffie-Hellman with three Parties

The Diffie-Hellman key agreement protocol can easily be extended to work with three or more people but it takes more round in the communication than on the tripartite protocol from the Wail Pairing. Assume that A, B and C want to agreement on a common secret key.

1. A chooses a random large integer x and sends B

   $X = g^x \bmod n$

2. B chooses a random large integer y and sends C

   $Y = g^y \bmod n$

3. C chooses a random large integer y and sends A

   $Z = g^z \bmod n$

4. A sends B

   $Z' = Z^x \bmod n$

5. B sends C

   $X' = X^y \bmod n$

6. C sends A

$$Y' = Y^z \bmod n$$

7. A computes

$$K_A = Y'^x$$

8. Bob computes

$$K_B = Z'^y$$

9. C computes

$$K_C = X'^z$$

The secret keys are equal to $g^{xyz} \bmod n$ but more participants need more communication rounds to agree on a common session key.

2.4. Elliptic Curve Cryptography

Elliptic Curve Cryptosystem (ECC) was present by Neal Koblitz [5] and Victor Miller in 1985. ECC offers an alternative way to establish public-key systems. The security of ECC is based on the fact that there is no sub-exponential algorithm known to solve the discrete logarithm problem (ECDLP) on a properly chosen elliptic curve. The reason of implement with ECC .is that smaller parameters can be used in ECC than in other competitive systems such RSA, but with equivalent levels of security. Some advantages of having smaller key size include faster computations, reductions in processing power, storage space and bandwidth. ECC has accepted by standard organizations. Such as Elliptic Curve Digital Signature Algorithm (ECDSA) [6] proposed in 1992 by Scott V anstone [7] was accepted in 1998 as an ISO standard (ISO 14888-3), accepted in 2000 as an IEEE standard (IEEE P1363) and a FIPS standard (FIPS 186-2).

In this section we give a short introduction to the theory of elliptic curves defined over finite field. Additional information on elliptic curve and its applications to cryptography can be learned in Blake et al , Menezes, chapter 6 of Koblitz's book.

The ways of defining equations for elliptic curves depend on whether the field is a prime finite field or a characteristic two finite field. The Weierstrass equation for finite field F_P is described in the next sections.

Elliptic Curves over F_P

Let $p > 3$ be an odd prime and $a, b \in F_p$ satisfy $4a^3 + 27b^2 \neq 0 (\text{mod } p)$. Then an elliptic curve $E(F_p)$ over F_P defined by the parameters $a, b \in F_p$ consists of a special point O called the *point at infinity* and the set of point $P = (x, y)$ for $x, y \in F_p$. The set of P satisfy the equation as follow:

$$y^2 = x^3 + ax + b$$

For given point $P = (x, y), x$ is called the x-coordinate of P, and y is called the y-coordinate of P. G is a generate point of order n on elliptic curve where n is a large integer. The addition formula on the elliptic curve is specified as follows:

1. $P + O = O + P = P$ for all $P \in E(F_p)$.
2. If $P = (x, y) \in E(F_p)$, then $(x, y) + (x, -y) = O$.(The point $(x, -y) \in E(F_p)$ is denoted $-P$, and is called the *negative* of P).
3. Let $P = (x_1, y_1) \in E(F_p)$ and $Q = (x_2, y_2) \in E(F_p)$, where $P \neq -Q$. Then $R = P + Q = (x_3, y_3)$, where

$$\begin{cases} x^3 = d^2 - x_1 - x_2 \\ y^3 = d(x_1 - x_3) \end{cases} \text{and}\, d = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & if \quad P \neq Q \\ \dfrac{3x_1{}^2 + a}{2y_1} & if \quad P = Q \end{cases}$$

2.5. Weil Pairing

In this section, we briefly describe the basic definition and properties of the bilinear pairing and the *BDH Assumption*. The Wail pairing is a pairing of bilinear pairings. The bilinear characteristic of Wail Pairing can be applied to reduce communication rouns than tripartite key agreement protocol with Diffie-Hellman's scheme (Joux' protocol just needs one round). Then we give a brief introduction of Joux's protocol and *man-in-the-middle attack* on Joux's protocol.

Bilinear Pairings and the BDH Assumption

Let G_1 be a cyclic additive group generated by P, whose order is a prime q, and G_2 be a cyclic multiplicative group of the same order q. We assume that the discrete logarithm problems (DLP) in both G_1 and G_2 are hard. Let $e: G_1 \times G_1 \rightarrow G_2$ be a pairing which satisfies the following conditions:

1. Bilinear: $e(P_1 + P_2, Q) = e(P_1, Q) \cdot e(P_2, Q)$ and $e(P, Q_1 + Q_2) = e(P, Q_1)e(P, Q_2)$.
2. Non-degenerate: There exists $P \in G_1$ and $Q \in G_1$ such that $e(P, Q) \neq 1$.
3. Computability: There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

We note that the Weil pairings associated with supersingular elliptic curve can be modified to create such bilinear maps.

Definition 1. The Bilinear Diffie-Hellman (BDH) Problem for a bilinear pairing $e: G_1 \times G_1 \rightarrow G_2$ is defined as follows: given $P, aP, bP, cP \in G_1$, compute $e(P, P)^{abc}$, where a, b, c are randomly chosen from Z_q^*. An algorithm is said to solve the BDH problem with an advantage of e if

$$\Pr[A(P, aP, bP, cP) = e(P, P)^{abc}] \geq e .$$

BDH Assumption: We assume that the BDH problem is hard, which means there is no polynomial time algorithm to solve BDH problem with non-negligible probability.

2.6. Joux 's One Round Protocol for Tripartite Diffie-Hellman

Assume *A*, *B* and *C* want to agree on a common session key. *A*, *B* and *C*, respectively, choose random numbers x, y and z from Z_q^* and compute $T_A = aG$, $T_B = bG$ and $T_C = cG$ where *G* is a generate pointer in an elliptic curve. Then *A*, *B* and *C* broadcast these values.

<u>*Protocol messages*</u>:

$A \to B, C : aG$

$B \to A, C : bG$

$C \to A, B : cG$

In the protocol, "$\to$" denotes by broadcast to the others. After the communication is over, *A* computes $K_A = e(bG, cG)^a$, B computes $K_B = e(aG, cG)^b$, and *C* computes $K_C = e(aG, bG)^c$. By bilinearity of *e*, these are all equal to $K_{ABC} = e(G, G)^{abc}$ and K_{ABC} is the session key shared by *A*, *B* and *C*. The security of this protocol is based on the hardness of the bilinear Diffie-Hellman problem.

2.7. Man-in-the-middle attack on Joux protocol

Assume an adversary *E* creates ephemeral private keys a', b' and c'. *E* replaces T_A, T_B and T_C with $T_A' = a'G, T_B' = b'G$ and $T_C' = c'G$, respectively. *E*'s messages are as follow.

<u>E' *messages*</u>:

$E_{B,C} \to A : b'G, c'G$

$E_{A,C} \to B : a'G, c'G$

$E_{A,B} \to C : a'G, b'G$

In the attack, "$E_{B,C}$" is denoted that *E* impersonates both B and C. Then *A* computes a session key $K_A = e(b'G, c'G)^a$. B computes a session key $K_B = e(a'G, c'G)^b$. C computes a session key $K_C = e(a'P, b'P)^c$. Then *E* who knows the values a', b' and c' is also able to compute these session keys from known values as follows:

$$K_A = e(T_A, P)^{b'c'} = e(P, P)^{ab'c'}$$
$$K_B = e(T_B, P)^{a'c'} = e(P, P)^{a'bc'}$$

$$K_C = e(T_C, P)^{a'b'} = e(P, P)^{a'b'c}$$

When these keys are used to encrypt the communication between A, B and C, E can impersonate as anyone of them.

3. Authenticated Key Agreement Protocols

We introduce three AK protocols that Seo-Sweeney's SAKA in Section 3.1, Kyungah Shim's protocol in Section 3.2. Pre-shared password scheme is used in Seo-Sweeny's protocol to provide authentication of user's identity and session key. A certificate is applied Kyungah Shim's protocol. These schemes including pre-shared password and certificates for authentication are respectively added to our first and second protocol.

3.1. Seo and Sweeney's Simple Authenticated Key Agreement Protocol

There are two phases, which are key exchange phase and key commitment phase in SAKA. In the key exchange phase, Alice and Bob exchange public information to calculate the common session key. Moreover, in the key commitment phase, they transfer product of the session key and password with each other and verify session key by inverse of pre-shared password.

In the initial, Alice and Bob share a secret password *p*. Assume that the system has the same public values and g, where n is a large prime and g is a generator with order $n-1$ in $GF(n)$. Alice and Bob first calculate *q* and $q^{-1} \bmod (n-1)$ from *p*, where *q* is computed in a predetermined way and is relatively prime to $n-1$.

***Key exchange phase*:**

Step 1： Alice chooses a random integer a, and sends Bob

$$X_1 = g^{aq} \bmod n$$

Step 2： Bob chooses a random integer b, and sends Alice

$$Y_1 = g^{bq} \bmod n$$

Step 3： After Alice receives Y_1, she computes

$$Y = (Y_1)^{q^{-1}} \bmod n = g^b \bmod n$$

$$K_A = (Y)^a \bmod n = g^{ab} \bmod n$$

Step 4： After Bob receives X_1, he computes

$$X = (X_1)^{q^{-1}} \bmod n = g^a \bmod n$$

$$K_B = (X)^b \bmod n = g^{ab} \bmod n$$

***Key commitment phase*:**

Step 1： Alice computes and sends Bob

$$(K_A)^q \bmod n$$

Step 2： Bob computes and sends Alice

$$(K_B)^q \bmod n$$

Step 3： After Alice receives $(Key_2)^q \bmod n$, she computes and check whether

$$K_A \overset{?}{=} (K_B)^{qq^{-1}} \bmod n$$

Step 4： After Bob receives $(Key_1)^Q \bmod n$, he computes and check whether

$$K_A \overset{?}{=} (K_B)^{qq^{-1}} \bmod n$$

If someone intercepts and replaces the exchanged messages with his own messages in the key exchange phase, the scheme will be detected because Alice or Bob will multiply q^{-1} to check whether $q \cdot q^{-1} \bmod n \overset{?}{=} 1$ or not in the key commitment phase. In this way, we can know that the protocol is successful to prevent the man-in-the-middle attack. But the other attacks, such as the straight replay attack, the reflective replay attack and the modification attack, still can not be resistant in the protocol. Taking the reflective replay attack for example, if E intercepts the messages Y_1 and $(K_B)^q$ from B to A and resends them back to A sequentially, A computes and checks whether $K_A \overset{?}{=} (K_B)^{qq^{-1}} \bmod n$ or not. The result is true and A believes that she is B. The reflective replay attack is successful in SAKA.

3.2. Kyungah's Efficient One Round Tripartite Authenticated Key Agreement Protocol from Weil Pairing

In 2003 January, Kyungash has proposed a new protocol to improve Joux's protocol. *In initial step*, a certification authority (CA) is used to provide certificates, which conjoin users' identities to long-term key. The certificate of entity A will be of the form:

$$Cert_A = (I_A \| P_A \| S_{CA}(I_A \| P_A))$$

where I_A denotes A's identity string, P_A is A's public key, "$\|$" denotes the concatenation of data items, and S_{CA} is CA's signature.

In Kyungah's protocol, a, b and c are A, B and C's private key and $P_A = aG$, $P_B = bG$ and $P_C = cG$ are A, B and C's public key. x, y and $z \in Z_q^*$ are selected at random as the ephemeral private key of A, B and C . The ephemeral public key of A, B, C are, respectively $Q_A = xG, Q_B = yG$ and $Q_C = zG$.

Protocol messages:

$A \to B, C : Q_A, Cert_A$

$B \to A, C : Q_B, Cert_B$

$C \to A, B : Q_C, Cert_C$

Key generation

Four types of key generation are in the following. The keys computed by the three entities are given below.

$$K_A = \hat{e}(Q_B, Q_C)^{ax\hat{e}(P_B,P_C)^a} = \hat{e}(G,G)^{abcxyz\hat{e}(G,G)^{abc}}$$

$$K_B = \hat{e}(Q_A, Q_C)^{by\hat{e}(P_A,P_C)^b} = \hat{e}(G,G)^{abcxyz\hat{e}(G,G)^{abc}}$$

$$K_C = \hat{e}(Q_A, Q_B)^{cz\hat{e}(P_A,P_C)^c} = \hat{e}(G,G)^{abcxyz\hat{e}(G,G)^{abc}}$$

$$K_{ABC} = K_A = K_B = K_C = \hat{e}(G,G)^{abcxyz\hat{e}(G,G)^{abc}}$$

4. Proposed Schemes

4.1. Authenticated Key Agreement Protocol on Elliptic Curve Cryptography

In this section, we will briefly describe the notation and then introduce the proposed scheme. Our proposed scheme is based on the elliptic curve public key system.

4.1.1. Notations

- EC : An elliptic curve defined over Z_p where Z_p denotes the multiplicative group modulo p.
- G : A base point $G \in EC(Z_p)$ of order n which is prime.
- (a, P_a) : The key pair of Alice where a is the secret number that Alice selected, P_a is the public key and $P_a = aG$
- (b, P_b) : The key pair of Bob where b is the secret number that Bob selected, P_b is the public key and $P_b = bG$.
- s : The secret password that Alice and Bob shared secretly.
- q : The number which is computed from s through a predefined function.
- (x, y) : A point on the Elliptic Curve and $(x, y) = qG \bmod n$.
- P_y : A point on the Elliptic Curve and $P_y = yG \bmod n$.
- A : The message which Alice sends to Bob and $M_A = xP_a + t_a P_y \bmod n$.
- B : The message which Bob sends to Alice and $M_B = xP_b + t_b P_y \bmod n$
- t_a, t_b : t_a is the timestamp which Alice generates message A and t_b is the timestamp which Bob generates message B.
- ΔT : ΔT is the predefined acceptable time delay.

4.1.2. Proposed Scheme

We assume that A and B have shared a secret password s and both of them can compute q from s through a predefined function. Then A and B compute the points $(x, y) = qG \bmod n$ and $P_y = yG$ respectively.

We add timestamps to enhance the strength of the security. The following is the detail description of our scheme：

Step 1 A computes M_A, where $M_A = xP_a + t_a P_y \bmod n$ and then Alice sends Bob $\{P_a, M_A, t_a\}$.

Step 2 After Bob receives $\{P_a, M_A, t_a\}$, Bob checks whether t_a is in $\triangle T$ or not. If the result is false, Bob will terminate the connection and do nothing. If the result is true, Bob will compute B, where $M_B = xP_b + t_b P_y$. And then Bob sends $\{P_b, M_B, t_b\}$ to Alice.

Step 3 Alice verifies $M_B \overset{?}{\neq} M_A$ and t_b.is in $\triangle T$. If the result is false, Alice will terminate the connection and do nothing. If the result is true, Alice will check if $M_B \overset{?}{=} xP_b + t_b P_y$. If the result is false, Alice terminates the connection.

Step 4 Bob checks whether $M_A \overset{?}{=} (xP_a + t_a P_y) \bmod n$ or not. If the result is false, Bob terminates the connection.

Step 5 Alice generates K_A where $K_A = aP_b \bmod n = a(bG) \bmod n$.

Step 6 Bob generates K_B where $K_B = bP_a \bmod n = b(aG) \bmod n$.

$$K_{AB} = K_A = K_B = ab(G)$$

4.2. Tripartite Authenticated Key Agreement Protocol on Public Key Infrastructure

The generation of session key in Joux's tripartite protocol just needs one round and takes less round than previous tripartite key agreement. Since the participants in communication are not authenticated, it is vulnerable from the man-in-the-middle attack.

In this section, we propose a tripartite authenticated key agreement protocol on public key infrastructure. The protocol needs only one round of communication to send a certificate and authentication messages including the sender's signature on ephemeral public key and timestamp. This authentication message assures that no one can forward it to others and the short-lived timestamp limits the use of the signature in $\triangle T$. We use both short-term key and long-term key pairs to compute the session key. Thus our protocol offers the security attributes including known session key security, perfect forward secrecy, no key-compromise impersonation and no key control. Besides, the discussion of the attack on our protocol is present in section 5.

In initial step, a certification authority (CA) is used to provide certificates, which conjoin users' identities to long-term key. The certificate of entity A will be of the form:

$$Cert_A = (I_A \| P_A \| S_{CA}(I_A \| P_A))$$

where I_A denotes A's identity string, P_A is A's long-term public key, "$\|$" denotes the concatenation of data items, and S_{CA} is CA's signature.

Brief description of the notation will be given and then introduce a tripartite authenticated key agreement protocol on public key infrastructure.

4.2.1. Notations

- EC: an *Elliptic curve* defined over Z_p where Z_p denotes the multiplicative group modulo p and p is a prime number.
- G : A generation point $G \in EC(Z_p)$ of order n, which is prime.
- ID_A, ID_B and ID_C : ID_A, ID_B and ID_C are respectively *A*'s, *B*'s and *C*'s *ID*.
- $Cert_A$, $Cert_B$, $Cert_C$: $Cert_A$, $Cert_B$, $Cert_C$ are respectively *A*'s certificate, *B*'s certificate and *C*'s certificate.
- (P_A, a): *A* random selects a number a as his long-term private key. P_A is *A*'s long-term public key ($P_A = aG$).
- (Q_A, x): In each communication, *A* random selects a new number x as *A*'s ephemeral private key. Q_A is *A*'s ephemeral public key.($Q_A = xG$).
- (P_B, b): *B* random selects a number b as his long-term private key. P_B is *B*'s long-term public key ($P_B = bG$).
- (Q_B, y): In each communication, *B* random selects a new number x_B as *B*'s ephemeral private key. Q_B is *B*'s ephemeral public key ($Q_B = yG$).
- (P_C, c): *C* random selects a number c as his long-term private key. P_C is *A*'s long-term public key ($P_C = cG$).
- (Q_C, z) : In each communication, *C* random selects a new number z as *C*'s ephemeral private key. Q_C is *A*'s ephemeral public key ($Q_C = zG$).
- M_{AB}, M_{AC} : M_{AB} is the message from *A* to *B* and M_{AC} is from *A* to *C*.
- M_{BA}, M_{BC} : M_{BA} is the message from B to A and M_{BC} is from *B* to C.
- M_{CA}, M_{CB} : M_{CA} is the message from C to A and M_{CB} is from *C* to *B*.
- T_A, T_B, T_C : T_A, T_B and T_C are the timestamp which *A*, *B* and *C* generate his authenticated message respectively.
- S_A : Signature of the message ($ID_R \| Q_A \| T_A$) signed by A. R denotes receiver of

the message.

- S_B : Signature of the message ($ID_R \| Q_B \| T_B$) signed by *B*.
- S_C : Signature of the message ($ID_R \| Q_C \| T_C$) signed by C.
- ΔT : ΔT is the predefined acceptable time delay.

4.2.2. Public Key Cryptosystems

Each entity in public key cryptosystem possesses a pair of keys that one is his public key and another is his private key. Each user replaces his public key in a public register such as CA and keeps his private key as secret. All participants can access the public key. The encryption ways have the following important character.

If a man who only knows the encryption key and the encryption function is hard to determine the decryption key.

If one of the two related keys is used to encrypt, the other is to decrypt. We assume that the timestamp is applied to a synchronization of clocks and the time that an entity *A* sends a message to another B takes ΔT . The presupposition is that *A* and *B* in a local area network LAN. If M_{AB} cannot arrive to *B* in ΔT , M_{AB} will be view as useless and *B* terminates the communication. Table 1 describes the predefined acceptable time delay.

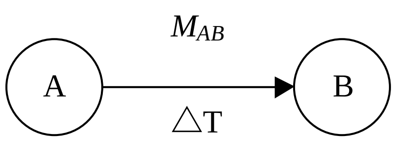

Figure 1. The predefined acceptable time delay

4.2.3. Proposed Scheme

Step 1 T_A, A randomly selects a new number a to compute the ephemeral public key $Q_A = xG$. Then A computes M_{AB} and M_{AC}

$M_{AB} = (W_{AB} \| S_A(W_{AB}))$, where $W_{AB} = ID_B \| Q_A \| T_A$

$M_{AC} = (W_{AC} \| S_A(W_{AC}))$, where $W_{AC} = ID_C \| Q_A \| T_A$

A respectively sends M_{AB}, $Cert_A$ to B and M_{AC}, $Cert_A$ to C.

Step 2 T_B, B randomly selects a new number y to compute the ephemeral public key $Q_B = yG$. Then B computes M_{BA} and M_{BC}

$M_{BA} = (W_{BA} \| S_B(W_{BA}))$, where $W_{BA} = ID_A \| Q_B \| T_B$.

$M_{BC} = (W_{BC} \| S_B(W_{BC}))$, where $W_{BC} = ID_C \| Q_B \| T_B$.

B respectively sends M_{BA}, $Cert_B$ to A and M_{BC}, $Cert_B$ to C.

Step 3 T_C, C randomly selects a new number z to compute the ephemeral public key $Q_C = zG$. Then C computes M_{CA} and M_{CB}

$M_{CA} = (W_{CA} \| S_C(W_{CA}))$, where $W_{CA} = ID_C \| Q_C \| T_C$.

$M_{CB} = (W_{CB} \| S_C(W_{CB}))$, where $W_{CB} = ID_A \| Q_C \| T_C$.

C respectively sends M_{CA}, $Cert_C$ to A and M_{CB}, $Cert_C$ to B.

Step 4 After A receives M_{BA} and $Cert_B$, A verifies it as follow:

(1) Check whether timestamp is in ΔT or not.

(2) Check the validity of M_{BA} by verifying the W_{BA}.

After A receives M_{CA} and $Cert_C$, A verifies M_{CA} as follow:

(1) Check whether timestamp is in ΔT or not.

(2) Check the validity of M_{CA} by verifying the W_{CA}.

If both M_{BA} and M_{CA} are validity, A computes K_A

$$K_A = e(P_B + Q_B, P_C + Q_C)^{a+x}$$

Step 5 After B receives M_{AB}, $Cert_A$ and M_{CB}, $Cert_C$, B verifies M_{AB} and M_{CB} as A. If both M_{AB} and M_{CB} are correct, then B computes K_B

$$K_B = e(P_A + Q_A, P_C + Q_C)^{b+y}$$

Step 6 After C receives M_{AC}, $Cert_A$ and M_{BC}, $Cert_B$, C verifies M_{AC} and M_{BC} If both M_{AC} and M_{BC} are correct, then C computes

$$K_C = e(P_A + Q_A, P_B + Q_B)^{c+z}$$

Finally, A, B and C share a common session key

$$K_{ABC} = K_A = K_B = K_C = e(G,G)^{(a+x)(b+y)(c+z)}$$

5. Complexity and Security Analysis

5.1. Complexity Analysis of Authenticated Key Agreement on Elliptic Curve Cryptography

We will compare the performance of the proposed scheme with Seo-Sweeney's scheme. Since Seo-Sweeney's scheme is based on the DLP difficulty (Discrete Logarithm Problem), the proposed scheme is based on the ECDLP difficulty. For practical implementation, we often choose a 1024-bit large prime as the modulus to ensure that solving DLP will be difficult. An elliptical curve $EC(\mathrm{Z}_p)$ with a point $P \in EC(Z_p)$ whose order is 160-bits prime offers approximately the same level of security as DLP with 1024-bits modulus. The following *assumptions* are made:

- The secret password s is an 160 bit random integer
- The private keys in both Seo- Sweeney's and our schemes are 160-bit random integers.
- In Seo and Sweeney's scheme, we assume that $X_1 = g^{aq} \bmod n$ which n is a 1024-bit prime and aq is an160-bits number.
- In our scheme, we assumed that an elliptical curve is chosen $EC(\mathrm{Z}_p)$ with $p \approx 2^{160}$.
- Some notations are defined as follows:

  (1). T_{MUL} : the time needed for an1024-bits modular multiplication.

  (2). T_{EXP} : the time needed for the modular exponentiation with 1024-bits modulus.

  (3). T_{EC_MUL} : the time needed for an elliptic-curve multiplication with 160-bits multiplier.

  (4). T_{EC_ADD} : the time needed for an elliptic-curve addition over $E(\mathrm{Z}_p)$

According to [8], we will know the relationship:

$T_{EXP} \approx 240T_{MUL}$; $T_{EC\_MUL} \approx 29T_{MUL}$; $T_{EC\_ADD} \approx 5T_{MUL}$

In Seo-Sweeney's scheme, the time for private key generation can be ignored because the key is a randomly chosen 160-bit integer. In the key exchange phase, Seo-Sweeney takes $4T_{EXP}$. In the key commitment phase, Seo-Sweeney takes $4T_{EXP}$. Through a series of statistics, we find that Seo-Sweeney's scheme must take $8T_{EXP}$.

In our scheme, the cost of computing q is negligible if the predefined way is a simple mathematics function. Through a series of statistics, we find that our scheme takes 16 elliptic-curve multiplications, 4 elliptic-curve additions, i.e. our scheme takes $16T_{EC\_MUL} + 4T_{EC\_ADD}$. Using the above assumptions, the computation time for our scheme against Seo-Sweeney's scheme is summarized in Figure 2.

	Seo-Sweeney's scheme	The proposed scheme
Total cost	$8T_{EXP}$=8×(240T_{MUL}) =1920T_{MUL}	$16T_{EC\_MUL}$+4T_{EC_ADD} =16×(29T_{MUL})+4×(5T_{MUL}) =484T_{MUL}

Figure 2. Comparison of computation with Seo-Sweeney

Obviously, our scheme is more efficient than Seo-Sweeney's scheme.

5.2. Complexity Analysis of Tripartite Authenticated Key Agreement Protocol on Public Key Infrastructure

We give a comprehensive idea about the number of computations per entity in Joux's, our second and Kyungah's protocol. The basic computations are that T_{ECC_MUL} denotes Elliptic Curve scalar Multiplications T_{ECC_ADD} is Addition of points on the Elliptic Curve (T_{ECC_ADD}), T_w is the evaluation of the Weil pairings. Kyungah's protocol uses certificates to authenticate identity of entities but way of authentication is not discussed. If Kyungah's protocol wants to offer authentication, signature must be applied in it. Therefore, we omit the operation of signature to compare computations with Kyungah's protocol. Figure 3 is comparison of computation to be performed by each user in these tripartite protocols.

	EC Scalar Multiplications	EC Additions	Weil pairing
Our second	1	2	1
Joux	1	None	1
Kyungah	1	None	2

Figure 3. Number of computations to be performed by each user

The main idea of Kyungah's protocol is that only one who knows a valid pair((a,x), (b,y), (c,z)) can compute $\hat{e}(G,G)^{abcxyz\hat{e}(G,G)^{abc}}$. The same concept of our second protocol is also that only one who knows a valid pair ((a, x), (b, y), (c, z)) can compute $e(G,G)^{(a+x)(b+y)(c+z)}$. There are $1T_{ECC\_MUL}$, $2T_{ECC\_ADD}$, $1T_w$ in our protocol and $1T_{ECC\_MUL}$, $2T_w$ in Kyungah' s protocol. Further, our protocol takes 2 T_{ECC_ADD} more and less $1T_w$ than Kyungah's. 2 T_{ECC_ADD} is less time than $1T_w$. Since our protocol is more efficient than Kyungah's protocol in the same security.

5.3. Security Analysis of Authenticated Key Agreement on Elliptic Curve Cryptography

We assume that Eve is an adversary and she can intercept the exchanged messages. She can get the following messages:

- ✧ $\{P_a, M_A, t_a\}$
- ✧ $\{P_b, M_B, t_b\}$

If Eve wants to fake Bob, she must pass the following checks:

(1)*Whether* t_a *is in* ΔT *or not*---------------------check(1)

(2) $M_A \overset{?}{=} xP_a + t_a P_y \bmod n$ -------------------------check(2)

If Eve wants to fake Alice, she must pass the following checks:

(3) *Whether* t_b *is in* ΔT *or not* --------------------check(3)

(4) $M_B \overset{?}{\neq} M_A$--check(4)

(5) $M_B \overset{?}{=} xP_b + t_b P_y$-----------------------------------check(5)

5.3.1. Attack 1：Straight Replay Attack

When A and B exchange the messages ($\{P_a, M_A, T_a\}, \{P_b, M_B, T_b\}$), E eavesdrops and duplicates the messages. After A and B stop communication, E pretends A to send B the messages ($\{P_a', M_A', t_a'\} = \{P_a, M_A, t_a\}$). After B receives the messages, she will check whether $T_1 - T_a \overset{?}{\leq} \Delta T$ or not. Because E spends $2\Delta T$ in the communication, the result is false. If the result is false, B will terminate the protocol.

5.3.2. Attack 2：Reflective Replay Attack

When Alice sends the message $\{P_a, M_A, t_a\}$ to Bob in the step 1, Eve intercepts the message and resends another message $\{P_a', M_A', t_a'\} = \{P_a, M_A, t_a\}$ back to A. This will make A believe that she is communicating with B. But it does not work in our scheme. The activity, which E intercepts the messages from A takes ΔT and another activity which E resends the messages back to A takes ΔT. The time that E spends is equal to $2\Delta T$. When A checks whether t_a is in ΔT or not and then finds that the result is false. Besides, when A checks that the receiving message A' is not equal to the message A, the result is false. So A will terminate the protocol.

5.3.3. Attack 3：Modification Attack

When the protocol begins, E intercepts the messages $(\{P_a, M_A, t_a\}, \{P_b, M_B, t_b\})$ between A and B. We assume that E sends message ($\{P_b, M_B', t_e\}$) to Alice. If E wants to fake A that she were B, she must pass check (3), (4) and (5). First, E must pass check (3) *whether* t_a *is in* ΔT *or no*, so she must replace the timestamp t_a by t_e. Secondly, she must make $M_B' \neq M_A$ to pass check (4) $M_B \neq M_A$. Finally, E must pass check (5) $M_B' - t_e P_y \overset{?}{=} xP_b$, where M_B', P_y, P_b are points on the elliptic curve.

E already knows M_B', t_e, P_b by intercepting the exchanged messages, but she doesn't know x and P_y. She must guess validly x and P_y to pass check (5) $M_B' - t_e P_y \overset{?}{=} xP_b$, but x and P_y are unknown. We can discuss three cases respectively with the following assumptions:

i. If E only gets P_y and x is unknown.

If E only gets P_y, she can compute M_C which M_C is a point on the elliptic curve and $M_C = M_B' - t_e P_y$. She uses M_C in substitution for $M_B' - t_e P_y$. She transforms (5) $M_B' - t_e P_y \overset{?}{=} xP_b$ to $xP_b \overset{?}{=} M_C$. If She wants to get a suitable x to pass (5), she must face the ECDLP problem.

ii. If E can only get x and P_y is unknown.

$$(6)\ M_B' - t_e P_y \overset{?}{=} M_C \text{ --check (6)}$$

(7) $M_B' - M_C \stackrel{?}{=} t_e P_y$ --check (7)

If she gets x, she computes M_C which M_C is a point on the elliptic curve and $M_C = xP_b$. She uses M_C in substitution for xP_y. She transforms check (5) $M_B' - t_e P_y \stackrel{?}{=} xP_b$ to $M_B' - t_e P_y \stackrel{?}{=} M_C$. She still transforms check (6) $M_B' - t_e P_y \stackrel{?}{=} M_C$ to check (7) $M_B' - M_C \stackrel{?}{=} t_e P_y$ which M_B' and M_C are known. She must try a suitable P_y to make (7) $M_B' - M_C \stackrel{?}{=} t_e P_y$ is successful. Because M_B' and M_C are known, we computes M_D which M_D is a point on EC and $M_D = M_B' - M_C$. We replace P_y and $M_B' - M_C$ with yG and M_D respectively. We can get the formula $t_e yG \stackrel{?}{=} M_D$ and *E* must guess a correct y to pass the check of the formula. If *E* wants to make *A* believe that she is *B*, she must face ECDLP problem.

iii. If a adversary *E* does not get both x and P_y

To pass check (5) without x and P_y is more difficult than without x or P_y. Therefore the problem that the modification attack can work in our proposed scheme is more difficult than the ECDLP problem.

5.3.4. Attack 4：Man-in-the-middle Attack

E intercepts the exchanged two messages ($\{P_a, M_A, t_a\}, \{P_b, M_B, t_b\}$), between *A* and *B*. And then *E* resends her own fabricated messages $\{P_c, M_C, t_c\}$ to *A* and *B*, which P_C is the public key that *E* fabricates, C is a point on the elliptic curve E, t_c is the timestamp. *E* tries to fake *A* and *B* that they are communicating with each other. But it doesn't work in our method. Because *E* must pass the check (3)、(4)、(5) if she wants to fake *A* or *B* with fabricated messages $\{P_c, M_C, t_c\}$. As the prior analysis of the modification attack, *E* can pass check (3) and check (4) so we directly analyze (5) check. If E wants to pass check (5), she must find out a correct pair of M_C and P_c and the problem is more difficult than the ECDLP.

5.4. Security Analysis of the Tripartite Authenticated Key Agreement Protocol on Public Key Infrastructure

We assume that A, B and C are the three entities in our protocol and want to share a common secret session key and E is an adversary intercepting the exchange messages between A, B and C.

First A, B and C send his certificate to others. The bellow is the messages between A, B and C.

A sends M_{AB} and M_{AC}, respectively B and C.

B sends M_{BA} and M_{BC}, respectively A and C.

C sends M_{CA} and M_{CB}, respectively A and B.

The message $M_{AB} = (W_{AB} \| S_A(W_{AB}))$ denotes that A and B are the sender and the receiver of M_{AB}. The sender A encrypts the message on B's public key and it assures that no one can decipher the message. B can check if M_{AB} receives in the predefined acceptable time delay and verify identity of the sender by the A's signature.

5.4.1. Straight replay attack

When A, B and C exchanges the messages, E eavesdrops and duplicates the messages. After termination of the communication, E impersonates A and B to send M_{AC} and M_{BC} to C. After C receives E's messages, C will check whether the timestamp is in ΔT or not. We discuss whether E can make C believe she were A or not. Here, $E(A)$ denotes that E impersonates A and $E(B)$ denotes that E impersonates B. Because E spends $2\Delta T$, including receiving A's message and sending A's messages to C, the result is false and E will terminate the communication with $E(A)$. As A's situation, C also terminates the communication with $E(B)$.

5.4.2. Reflective replay attack

When A sends the message M_{AB} to B, E intercepts M_{AB} and resends M_{AB} to A. E wants to make A believe that she were B. But this is not work in our protocol because when A deciphers M_{AB} on his private key, he will find the message is indiscriminate and terminate this communication with $E(B)$.

5.4.3. Modification attack

In the initial, E intercepts the messages, which are the cyphertext encrypting on the receiver's public key, between A, B and C. E can not modify the message and if E wants to modify the messages, E must comprise the receiver's private key, for example, to determine a from $P_A = aP$, is equivalent to solving the ECDLP in G_1. The modification attack cannot work in our protocol.

5.4.4. Man-in-the-middle attack

Joux's protocol suffers from man-in-the-middle attack. The public key of the entities in Joux's protocol is not authenticated. An entity in our protocol possesses a certificate including personal information and public key. An adversary cannot impersonate others with the certificate. The exchanging messages need the sender's signature but an adversary is not able to make a counterfeit of sender's signature without sender's private. The man-in-the-middle attack can be overcome in our protocol.

5.4.5. Unknown key shared attack

According as Al-Riyami's protocol, we introduce unknown key shared attack in the following. Unknown key shared attack employs a potential registration weakness for public keys to create fraudulent certificates. In the initial, an adversary E registers A's public key P_A as her own, and CA issues $Cert_E = (I_E \parallel P_A \parallel S_{CA}(I_E \parallel P_A))$ to E.

She intercepts A's messages including $Cert_A$ and then replaces $Cert_A$ with $Cert_E$. E registering A's public key as her own doesn't know A's private key a. Therefore she cannot get the session key between A, B and C. However, B and C are thinking that they are have agreed a key with A. This drawback can be overcome if the CA does not allow the two entities that have registered possess the same public key. However, it is hard and time-consuming for the large or distributed system checking the public key of entity.

Even if the solution can prevent the basic unknown key shared attack, the smart adversary still attacks the protocol by modification of registering public key. E registers $P_E = aP_A$ and alters short-term key. The adversary can make the two participants B, C believe that messages came from her rather then from the participant A.

We present the attacks on our protocol. The authenticated message in our protocol including sender's signature and it means that even if E intercepts the message and resends it to another, the receiver doesn't believe the message came from her. Besides, A's signature and the timestamp in the message have encrypted on the sender's public key. Only the receiver can decipher the message and others can't get the ephemeral key.

We assume that an adversary learning the ephemeral key and A's signature, the ephemeral key cannot be used in the next round of our protocol because A's signature includes an effective timestamp which the ephemeral key cannot be used over the timestamp. Figure 4 is the comparison of these attacks on Joux’s , TAK and our protocol.

Attack name	Joux	Our
Straight replay	No	Yes$^{(1,2)}$
Reflective replay	No	Yes$^{(1,2)}$
Modification	No	Yes
Man-in-the-middle	No	Yes
Unknown key shared	No	Yes

Figure 4. The comparison of these attacks

(1). If the authenticated messages are limited in a short-lived timestamp, the adversary can resend it to others but doesn't know the session key. In general cast, the attacker who has learned previous session key launches the attack.

(2). A synchronization of clocks and a local area network are required.

6. Conclusions

In this thesis, we proposed two novel schemes for authenticated key agreement protocol. Our first scheme have improved Seo-Sweneey's protocol and developed a more efficient password-based authenticated key agreement protocol based on the elliptic curve. Our first protocol requires less communication load and quarter computation cost than Seo-Sweneey's protocol. Further, our first protocol can also prevent the attacks, the reflective replay attack, the straight replay attack, the man-in-the-middle attack, and the modification attack.

We proposed a tripartite authenticated key agreement protocol on public key infrastructure. Our second protocol prevents various attack such as straight replay attack, reflective replay attack, man-in-the-middle attack and unknown key shared attack. Each entity in our protocol possesses a log-term key pair and an ephemeral key pair. The authentication is built on the sender's signature and short-lived timestamp. We improve Joux's protocol in our protocol, especially in resisting these attacks. Besides, our second protocol is more efficient than Kyungah's protocol.

Reference:

[1]. W. Diffie and M.E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, v, IT-22, n, 6, Nov 1976, pp. 109-112.

[2]. Dong Hwi Seo and P. Sweneey: "Simple authenticated key agreement algorithm", *Electronics Letters*, 24, June, 1999, Vol.35, No.13, pp.1073-p1074

[3]. Joux A. 'A one round protocol for tripartite Diffie-Hellman', Proc. 4th Algorithmic Number Theory Symp. (ANTS IV), Leiden, The Netherlands, July 2000, pp.385-394

[4]. Kyungah Shim "Efficient one round tripartite authenticated key agreement protocol from Weil pairing," *IEE* 2003 *Electronics Letters* Online No:20030170.

[5]. N. Kobliz: " Elliptic Curve Cryptosystems Mathematics of Computation, 48, 1987, pp.203-209.

[6]. D. Johnson, A, Menezes, "The Elliptic Curve Digital Signature Algorithm," Technical Report COOR 99-34, Dept. of C&O, University of Waterloo, Canada, available at: http://www.cacr.math.uwaterloo.ca.

[7]. S. Vanstone: "Responses to NIST's Proposal," Communications of the ACM, 35, July 1992, pp. 50-52.

[8]. N. Koblitz, A. Menezes and S. Vanstone: " The State of Elliptic curve Cryptography, Design," Codes and Cryptography, 19, 2000, pp.173-193.

[9]. ANSI X9.63, Public Key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Key Transport Protocols, working draft, October 2000.

[10]. A. Menezes: " Elliptic curve Public Key Cryptosystems, " Kluwer Academic Publishers, 1993.

[11]. V. Miller: " Uses of Elliptic Curves in Cryptography, Advances in Cryptology -

85, Proceedings, Lecture Notes in Compute Science, No. 218, Springer-Verlag, New York, 1985, pp.417-426.

[12]. S. Al-Riyami and K. Paterson, 'Authenticated three party key agreement protocols from pairings', Cryptology ePrint Archive, Report 2002/035, available at http://eprint.iacr.org/2002/035

[13]. Kyungah Shim, 'Unknown key-share attack on authenticated multiple-key agreement protocol', IEE 2003 Electronics Letters Online No: 20030076.

[14]. Tzong-Sun Wu, Wei-Hua He and Chien-Lung Hsu, 'Security of authenticated multiple-key agreement protocols', Electronics Lett, March 1999 Vol. 35 No. 5.

[15]. Alfred Menezes, 'Elliptic Curve Public Key Cryptosystems' Kluwer Academic Publishers, 1993.

[16]. Willian Stallings, ' Cryptography and Network Security' Alan R. Apt Publisher, 2003.

[17]. D. Boneh and M. Franklin. "Identity-based encryption from the Weil Pairing". In Advances in Cryptolohy-CRYPTO 2001, Springer-Verlag LNCS 2139, 213-229, 2001.

[18]. Divya Nalla and K.C.Reddy "ID-based tripartite Authenticated Key agreement Protocols from pairing".

[19]. RSA Security Inc., http://www.rsaseurity.com/