Abstract—Multiparty key agreement has many applications on internet services such as secure teleconferencing. A group of users can hold a conference securely over an open network by running a multiparty key agreement protocol to generate a common secret key. With the common secret key, data transmission over the internet is protected for confidentiality. In 2003, Barua (INDOCRYPT 2003) first proposed a multiparty key agreement protocol by using Weil pairing. The protocol is based on ternary trees and Joux’s tripartite key agreement. However, in Barua’s protocol the communication round for \( n \) entities is \( \lceil \log_3 n \rceil \), which is proportional to the number of participants. In this paper, we propose a new multiparty key agreement protocol from Weil pairing that needs only constant number of rounds. Besides, the message size, the total number of scalar multiplications, and the number of Weil pairing are reduced.

Index Terms—multiparty key agreement, tripartite authenticated key agreement protocol, conference key agreement, Weil pairing.

I. INTRODUCTION

Multiparty key agreement is an interesting research topic and has many applications on internet services such as secure teleconferencing. When a group of people want to have a conference securely over an open network, they have to run a multiparty key agreement protocol to share a common secret key \( K \). By using the common secret key, they can encrypt message to be shared with each other such that an adversary cannot decrypt it without the key.

In this paper, we propose a new multiparty key agreement protocol based on Weil pairing. We have two versions of the multiparty key agreement protocol: one is authenticated version and the other is unauthenticated version. In the proposed protocol we use the topology of broadcast network which is the same as in Barua’s protocol \( [1] \) (based on Joux’s tripartite key agreement protocol \( [2] \)). We show that for \( n > 2 \), where \( n \) is the communication parties, our protocols need constant rounds (two rounds) of message transmission, which is independent of the number of participants. In Barua’s protocol, the communication round for \( n \) entities is \( \lceil \log_3 n \rceil \), which is proportional to the number of participants. Besides, the message size, the total number of scalar multiplications, and the number of Weil pairing used in our proposed protocol are reduced.

The rest of the paper is organized as follows: The basic definition and properties of the bilinear pairing are described in Section 2. We present a new multiparty key agreement protocol based on Weil pairing in Section 3. Section 4 analyzes the security of our protocol. Section 5 shows the performance of the protocols and comparison with Braua’s protocol. Finally, we have a conclusion in Section 6.

II. MODIFIED WEIL PAIRING

Let \( p \) be a prime such that \( p = 2 \text{ (mod) } 3 \) and \( p = 6q – 1 \) for some prime \( q > 3 \). Let \( E \) be a super-singular curve defined by \( y^2 = x^3 + 1 \) over \( F_p \). The set of rational points \( E(F_p) = \{(x, y) \in F_p \times F_p : (x, y) \in E \} \) forms a cyclic group of order \( p+1 \). Furthermore, because \( p+1 = 6q \) for some prime \( q \), the set of points of order \( q \) in \( E(F_p) \) form a cyclic subgroup, denoted as \( G_q \). Let \( P \in E(F_p) \) be a generator of the group of points with order \( q = (p+1)/6 \). Let \( \mu_q \) be the subgroup of \( F_p^\times \) that contains all elements of order \( q \). The Weil pairing on the curve \( E(F_p) \) is a mapping \( e : G_q \times G_q \rightarrow \mu_q \). The modified Weil pairing is defined as \( \hat{e} : G_q \times G_q \rightarrow \mu_q \).
\( \hat{e}(P, Q) = e(P, \phi(Q)) \), where \( \phi(x, y) = (x, y, 1 \neq \xi \in F_{p^2}' \) is a solution of \( x^3 - 1 = 0 \pmod{p} \) and \( G_q \) is the group of points with order \( q \). The modified Weil pairing then satisfies the following properties:

1. Bilinear: \( \hat{e}(a \cdot P, b \cdot Q) = \hat{e}(P, Q)^{ab} \), for all \( P, Q \in G_q \) and \( a, b \in \mathbb{Z} \).

2. Alternative: \( \hat{e}(P, Q) = \hat{e}(Q, P)^{-1} \).

3. Non-degenerate: there exists a point \( P \in G_q \) such that \( \hat{e}(P, P) \neq 1 \).

4. Polynomial-time computable: \( \hat{e}(P, Q) \) is computable in polynomial time.

## III. PROPOSED PROTOCOL

Suppose that \( n \) entities who wish to agree on a common secret key, indicated as an entity set \( U = \{U_1, U_2, \ldots, U_n\} \). The public domain parameters \((p, q, E, P, \hat{e})\) are common to all entities. In the authenticated version, we assume that the static public keys are exchanged via certificates. \( Cert_i \) denotes \( U_i \)'s public-key certificate, containing static public key \( Y = a \cdot P \) for \( i = 1, 2, \ldots, n \), an unique identifier string \( U_i \) (such as \( U_i \)'s name), and a signature of certified authority (CA) on this information, where \( a_i \) is a random number (used as the long-term private key) selected by \( U_i \).

### A. The unauthenticated protocol

The protocol uses the following parameters:

- \( U_i \): a participant in a communication round.
- \( x_i \): the short-term secret key randomly chosen by \( U_i \).
- \( T_i, X_i \): \( U_i \)'s public messages in each communication round.

#### Step1. Messages exchange (Round 1):

Each \( U_i, i = 1, \ldots, n \), chooses a random number \( x_i \), computes and broadcasts \( T_i = x_i \cdot P \).

#### Step2. Messages exchange (Round 2):

Each \( U_i, i = 1, \ldots, n \), computes and broadcasts
\[
X_i = e(T_{i+1}, (T_{i+2} - T_{i+1}))^{x_i}.
\]

### Step3. Key generation:

Each \( U_i, i = 1, \ldots, n \), computes \( K_i \) as follows:
\[
K_i = e(T_{i+1}, nT_{i+1})^{x_i} \cdot X_i^{n-1} \cdot X_i^{n-2} \cdots X_i^{n-i+2} \cdot X_i^{n-i+1} = e(P, P)^{(x_i^2 + x_i P) + (x_i^2 + x_i P) + \cdots + (x_i^2 + x_i P) + (x_i^2 + x_i P) + \cdots + (x_i^2 + x_i P)}.
\]

B. The authenticated protocol

The protocol uses the following parameters:

- \( U_i \): a participant in a communication round.
- \( a_i \): the long-term secret (private) key randomly chosen by \( U_i \).
- \( Y_i \): the long-term public key computed by \( Y_i = a_i \cdot P \).
- \( Cert_i \): \( U_i \)'s long-term public-key certificate.
- \( x_i \): the short-term (ephemeral) secret key randomly chosen by \( U_i \).
- \( T_i, X_i \): \( U_i \)'s public messages in each communication round.

#### Step1. Messages exchange (Round 1):

Each \( U_i, i = 1, \ldots, n \), chooses a random number \( x_i \), computes \( T_i = x_i \cdot P \) and broadcasts \( T_i \) and certificates \( Cert_i \).

#### Step2. Messages exchange (Round 2):

Each \( U_i, i = 1, \ldots, n \), computes and broadcasts
\[
X_i = e((Y_{i+1} + T_{i+1}), (Y_{i+2} + T_{i+2}) - (Y_{i+1} + T_{i+1}))^{(a_i^2 + x_i P)}.
\]

### Step3. Key generation:

Each \( U_i, i = 1, \ldots, n \), computes \( K_i \) as follows:
\[
K_i = e((Y_{i+1} + T_{i+1}), n(Y_{i+1} + T_{i+1}))^{(a_i^2 + x_i P)} \cdot X_i^{n-1} \cdot X_i^{n-2} \cdots X_i^{n-i+2} \cdot X_i^{n-i+1} = e(P, P)^{(a_i^2 + x_i P) + (a_i^2 + x_i P) + \cdots + (a_i^2 + x_i P) + (a_i^2 + x_i P) + \cdots + (a_i^2 + x_i P)}.
\]

Furthermore, both in the unauthenticated and the authenticated versions, the common shared secret key is then obtained as \( K = kdf(K_i || U_i || U_2 || \ldots || U_n) \) is an unique identifier of entity \( U_i \).
C. Examples

We have a simple example for the unauthenticated version in case of \(n = 10\).

**Step 1. Messages exchange (Round 1):**

Each \(U_i\) (\(i = 1, 2, \ldots, 10\)) computes and broadcasts \(T_i = x_iP\):

\[
T_1 = x_1P, \ T_2 = x_2P, \ T_3 = x_3P, \ T_4 = x_4P, \ \cdots, \ T_{10} = x_{10}P.
\]

**Step 2. Messages exchange (Round 2):**

Each \(U_i\) computes and broadcasts

\[
X_i = e(T_{i+1}, (T_{i+2} - T_{i+3}))^{x_i}.
\]

\[
X_1 = e(T_2, (T_3 - T_{10}))^{x_1}, \ X_2 = e(T_3, (T_4 - T_2))^{x_2}, \ X_3 = e(T_4, (T_5 - T_3))^{x_3}, \ X_4 = e(T_5, (T_6 - T_4))^{x_4}, \ X_5 = e(T_6, (T_7 - T_5))^{x_5}, \ X_6 = e(T_7, (T_8 - T_6))^{x_6}, \ X_7 = e(T_8, (T_9 - T_7))^{x_7}, \ X_8 = e(T_9, (T_{10} - T_8))^{x_8}, \ X_9 = e(T_{10}, (T_1 - T_9))^{x_9}, \ X_{10} = e(T_1, (T_2 - T_10))^{x_{10}}.
\]

**Step 3. Key generation:**

Each \(U_i\) computes his/her \(K_i\) as follows:

\[
K_i = e(T_{i+1}, nT_{i+1})^{x_i} \cdot X_i^{n+1} \cdot X_{i+1}^{n+2} \cdots X_{i+2}.
\]

\[
K_1 = e(T_2, 10T_2)^{x_1} \cdot X_2^{n+1} \cdot X_3^{n+2} \cdots X_4^{n+1}.
\]

\[
K_{10} = e(T_1, 10T_1)^{x_{10}} \cdot X_1^{n+1} \cdot X_2^{n+2} \cdots X_3^{n+1}.
\]

Following the computation equation of \(K_i\), we can obtain that \(K_1 = K_2 = K_3 = K_4 = K_5 = K_6 = K_7 = K_8 = K_9 = K_{10}\).

IV. Security Analysis

We analyze the security of the proposed protocol by inspecting attacks from both of the passive adversary and the active adversary.

**A. Passive adversary**

A passive adversary (an eavesdropper) is not a participant who tries to compute the common shared secret key by listening to the broadcast messages among the legal participants. If a multiparty key agreement protocol is secure against passive adversary, a passive adversary is unable to obtain information about the common shared secret key by eavesdropping messages transmitted over the broadcast channel.

To prove this, we will follow a well-known security assumption. We use the bilinear Diffie-Hellman problem assumption to prove our protocol is secure against passive adversary. The similar technique is used in literatures such as Boneh’s scheme [3]. We say that passive adversary cannot work under the assumption that solving the bilinear Diffie-Hellman problem (BDHP) will be infeasible. The definition of BDHP is to compute \(\hat{e}(P, P)^{x_{25}}\) by given \((P, xP, bP, cP)\). That is, given \(T_1 = x_1P, T_2 = x_2P, T_3 = x_3P,\) and \(x_1, x_2, x_3\) are randomly chosen from \(Z\), the two tuples of random variables, \((T_1, T_2, T_3, \hat{e}(P, P)^{x_{25}})\) and \((T_1, T_2, T_3, T)\), where \(T\) is a random value in \(\mu_q\) are computationally indistinguishable. In other words, there is no efficient algorithm \(A\) satisfying

\[
|\Pr[A(x_1P, x_2P, x_3P, \hat{e}(P, P)^{x_{25}}) = true]| - \Pr[A(x_1P, x_2P, x_3P, T) = true]| > 1/O(|q|).
\]

for any polynomial \(O\), where the probability is over the random choice of \(x_1, x_2, x_3\) and \(T\).

First, we consider the case of passive adversary on our unauthenticated protocol. If an eavesdropper \(E\) intends to compromise \(K_i = e(T_{i+1}, nT_{i+1})^{x_i} \cdot X_i^{n+1} \cdot X_{i+1}^{n+2} \cdots X_{i+2}\) in our unauthenticated protocol, she needs to compute \(T = e(T_{i+1}, nT_{i+1})^{x_i} (equal \ to \ \hat{e}(P, P)^{x_{25}}))\) and \(X = X_i^{n+1} \cdot X_{i+1}^{n+2} \cdots X_{i+2}\), respectively, where \(T, X \in \mu_q\) and then she can obtain \(K_i = (T \cdot X)\). We assume that she can compute the value of \(X\) from
the public messages $X_i$'s. However, she cannot obtain the correct $x_i$ form $X_i$ (equal to $e(T_{i+1}, T_{i+2-T_{i+1}})$). Without knowing $x_{i+1}, x_i$ and $x_{i+1}$, she cannot compute the correct value of $T = \hat{e}(P, P)^{\frac{1}{n+1}}$. Because that she faces the hardness of BDHP for the pair of groups $G_q \times \mu_q$. To compute $T$ by given $P$, $x_{i+1}$, $x_i$, $x_{i+1}$, with that $x_{i+1}, x_i$ and $x_{i+1}$ are chosen randomly. That is, the two tuples of random variables $(T_{i+1}, T_{i}T_{i+1}e(P, P)^{\frac{1}{n+1}})$ and $(T_{i+1}, T_{i}, T_{i+1})$, where $T$ is a random value in $\mu_q$, are computationally indistinguishable. In other words, there is no efficient algorithm $A$ satisfying

$$\left| \Pr[A (x_{i+1}, P, x_P, x_{i+1}, P, \hat{e}(P, P)^{\frac{1}{n+1}}) = true] \right| - \left| \Pr[A (x_{i+1}, P, x_P, x_{i+1}, P, T=\text{true})] > 1/Q(|\ell|) \right. \tag{8}$$

for any polynomial $Q$, where the probability is over the random choice of $x_{i+1}, x_i, x_{i+1}$ and $T$. Therefore, she cannot compute easily the correct $K_i$.

Next, it is similar when we consider the case of passive adversary on our authenticated protocol. If $E$ tends to compromise the session key $K_i$ (equals to $e((Y_{i+1}+T_{i+1}), n(Y_{i+1}+T_{i+1}))^{\frac{1}{n+1}}$, $X_i = X_{i+1}^{-1} \cdots X_2$), the public messages $X_i$'s form the public value $X_i = e((Y_{i+1}+T_{i+1}), (Y_{i+2}+T_{i+2})-(Y_{i+1}+T_{i+1}))^{\frac{1}{n+1}}$. Without correct $a_i$ and $a_X$, she cannot compute $T = e((Y_{i+1}+T_{i+1}), n(Y_{i+1}+T_{i+1}))^{\frac{1}{n+1}}$. Because that she faces the hardness of the BDHP problem for the group of pairs $G_q \times \mu_q$. To compute $T = e((Y_{i+1}+T_{i+1}), n(Y_{i+1}+T_{i+1}))^{\frac{1}{n+1}}$, by given $P$, $(a_i + a_{i+1}X_i)P$, $(a_i + a_{i+1}X_i)P$, and $(a_i + a_{i+1}X_i)P$, with that $a_{i+1}$, $a_i$, $x_{i+1}, x_i$, and $x_{i+1}$ are chosen randomly. That is, the two tuples of random variables $((Y_{i+1}+T_{i+1}), (Y_{i+1}+T_{i+1}), (Y_{i+2}+T_{i+2}), \hat{e}(P, P)^{\frac{1}{n+1}})_{(a_i^{a_{i+1}X_i})^{\frac{1}{n+1}}}$ and $(Y_{i+1}+T_{i+1}), (Y_{i+1}+T_{i+1}), (Y_{i+2}+T_{i+2}), T)$, where $T$ is a random value in $\mu_q$, are computationally indistinguishable. In other words, there is no efficient algorithm $A$ satisfying

$$\Pr[A \left| ((a_i^{a_{i+1}X_i})P, (a_i^{a_{i+1}X_i})P, (a_i^{a_{i+1}X_i})P, \hat{e}(P, P)^{\frac{1}{n+1}})_{(a_i^{a_{i+1}X_i})^{\frac{1}{n+1}}} = true \right| - \Pr[A \left| ((a_i^{a_{i+1}X_i})P, (a_i^{a_{i+1}X_i})P, (a_i^{a_{i+1}X_i})P, T) = true \right| > 1/Q(|\ell|) \right. \tag{9}$$

for any polynomial $Q$, where the probability is over the random choice of $x_{i+1}, x_i, x_{i+1}$ and $T$. Therefore, she cannot compute easily the correct $K_i$.

**B. Active adversary**

An active adversary is a dishonest participant who tries to disrupt the establishment of a common key among all of the participants. An active adversary can fool an honest participant into believe that he has computed the same common key as the other honest participants do. We will show that our proposed protocols are secure against active adversary.

(1) Known-key security: An entity in each run of the protocol computes a new ephemeral private keys $x_i$ to generate a unique session key. Thus, the knowledge of a previous key does not help in deducing a new key.

(2) Forward secrecy: Suppose that a malicious adversary has compromised one or more long-term private keys $a_i$. However, he cannot compute the previously established session key $K_i = e((Y_{i+1}+T_{i+1}), n(Y_{i+1}+T_{i+1}))^{\frac{1}{n+1}}$. $X_i$ without knowing the ephemeral private key $x_i$.

(3) Key compromise impersonation resilience: The key-compromise impersonation attack means that the attacker $E$ who has compromised the long-term private key of one entity $U_i$ would not only impersonate the compromised entity but also impersonate any other one to fool the compromised entity. For example, an outsider attacker $E$, who has compromised $U_i$'s static private key $a_i$, can also impersonate the other entities to fool $U_i$. Suppose that $E$ who impersonates $U_i$ to fool $U_i$ can then forge a message $T_2' = uP$. Then $E$ broadcasts $\{T_2', Cert_2\}$ and claims that it is sent by $U_2$, where $u$ is chosen by $E$. Now, $U_i$ will compute $K_1 = e((Y_{i+2}+T_2), n(Y_{i+2}+T_2))^{\frac{1}{n+1}}$. $X_i^{n-1} \cdots X_2^{n-2} \cdots X_1 = e(P, P)^{\frac{1}{n+1}}$. $X_i^{n-1} \cdots X_2^{n-2} \cdots X_1 = e(P, P)^{\frac{1}{n+1}}$. $X_i^{n-1} \cdots X_2^{n-2} \cdots X_1$. However, $E$ cannot compute $K_2' = e((Y_{i+3}+T_3), n(Y_{i+3}+T_3))^{\frac{1}{n+1}}$. $X_i^{n-1} \cdots X_2^{n-2} \cdots X_1$. $\frac{1}{n+1}$.
(equals to \(e((Y_3+T_3), n(Y_2+T_2))^{a_1 x_{i1}} \cdot X_2^{n-1} \cdot X_3^{n-2} \cdots X_n \)). It fails because she does not know the correct value of \(a_i\) or \(a_i x_{i1}\). The proposed protocol provides the property of key-compromise impersonation resilience.

(4) Unknown key-shared resilience: The identity of a participant is included in the key derivation function of our proposed protocol. It provides unknown key-shared resilience as well as public-key substitution unknown key-shared attack.

(5) No key control: Each entity in a run of the protocol chooses a new ephemeral private keys \(x_i\) to generate a unique session key. In our protocol, no participant does control and predict the value of a common session key.

V. Performance Analysis

The performance analysis includes communication round, message size and computation cost. In the following, we give the definition for some notations.

- \(R(n)\) : the total number of rounds.
- \(B(n)\) : the total number of broadcast messages. If an entity broadcasts a message with value in \(G_q\) or \(\mu_q\) then \(B(n)\) is increased by 1.
- \(E(n)\) : the total number of scalar multiplications (namely computing \(kP\), where \(P \in G_q\) and \(k \in \mu_q\)).
- \(P(n)\) : the total number of Weil pairing operations.

We show the result for our proposed protocols as follows.

- \(R(n) = 2\) : All entity use two round to broadcast all messages and compute the shared secret key.
- \(B(n) = 2n\) : In first round, each entity broadcasts \(T_i \in G_q\). In second round, each entity broadcasts \(X_i \in \mu_q\). There are \(n\) entities in the system, consequently, \(B(n) = 2n\).
- \(E(n) = n\) : In the first round, each entity computes \(T_i = x_i P\) (in the unauthenticated case) or computes \(T_i = x_i Y_i\) (in the authenticated case). Consequently, \(E(n) = n\).
- \(P(n) = 2n\) : In second round, each entity uses two Weil pairing to compute \(X_i\) and the secret key \(K_i\). Consequently, \(P(n) = 2n\).

Note that, in our protocol, the above four performance complexities are the same for both of the unauthenticated case and the authenticated case.

As shown in table I, We can see that our protocol has better performance than Barua’s.

<table>
<thead>
<tr>
<th>()</th>
<th>()</th>
<th>()</th>
</tr>
</thead>
<tbody>
<tr>
<td>()</td>
<td>Unauthenticated</td>
<td>Authenticated</td>
</tr>
<tr>
<td>()</td>
<td>Barua’s</td>
<td>Ours</td>
</tr>
<tr>
<td>(R(n))</td>
<td>([-\log_2 n])</td>
<td>2</td>
</tr>
<tr>
<td>(B(n))</td>
<td>(&lt;5(n-1)/2)</td>
<td>(2n)</td>
</tr>
<tr>
<td>(E(n))</td>
<td>(&lt;5(n-1)/2)</td>
<td>(n)</td>
</tr>
<tr>
<td>(P(n))</td>
<td>(\leq n^2 \log_3 n)</td>
<td>(2n)</td>
</tr>
</tbody>
</table>

VI. Conclusion

In this paper, we proposed a new multiparty key agreement protocol for secure teleconferencing based on Weil pairing which provides both round number and computation efficiency. Security analysis shows that our protocol is the same as Barua’s protocol if the Bilinear Diffie-Hellman problem (BDHP) is hard.

ACKNOWLEDGMENT

This paper is partially supported by the National Science Council, Taiwan, NSC-94-2213-E-029-001.

REFERENCES


