

Identity-Based Key Agreement Protocols in a Multiple PKG Environment*

Hoonjung Lee¹, Donghyun Kim¹, Sangjin Kim², and Heekuck Oh¹

¹ Department of Computer Science and Engineering,
Hanyang University, Republic of Korea
{leehj, kimdh, hkoh}@cse.hanyang.ac.kr

² School of Internet Media Engineering,
Korea University of Technology and Education, Republic of Korea
sangjin@kut.ac.kr

Abstract. To date, most identity-based key agreement protocols are based on a single PKG (Private Key Generator) environment. In 2002, Chen and Kudla proposed an identity-based key agreement protocol for a multiple PKG environment, where each PKG shares identical system parameters but possesses a distinct master key. However, it is more realistic to assume that each PKG uses different system parameters. In this paper, we propose a new two party key agreement protocol between users belonging to different PKGs that do not share system parameters. We also extend this protocol to a tripartite key agreement protocol. Our two party protocol requires the same amount of pairing computation as Smart's protocol for a single PKG environment and provides PKG forward secrecy. We show that the proposed key agreement protocols satisfy every security requirements of key agreement protocols.

Keywords: ID-based cryptosystem, bilinear map, key agreement protocol, multiple PKG.

1 Introduction

Key establishment protocol is a cryptographic primitive that is used to share a common secret key between entities. This secret key is normally used as a session key to construct a secure channel between the entities concerned. Key establishment protocol can be subdivided into key transport protocol and key agreement protocol. In key transport protocol, one of the participant creates the shared key and distributes it to others securely. On the other hand, in key agreement protocol, each entity computes the common secret key using the information contributed by all the entities involved. In key transport protocol, all the participants have to trust the entity responsible for creating the new shared key. Therefore, normally a trusted third party is used as a server that creates and distributes shared keys. However, in this setting, this entity becomes a good

* This work was supported by the Ministry of Information and Communication, Korea, under the university HNRC-ITRC program supervised by the IITA.

target for attack and the success of such attack is catastrophic. Furthermore, this entity may become a bottleneck point. In contrast, key agreement protocols does not suffer from these problems, since no single entity determines the shared key.

1.1 Related Work

In 1984, Shamir introduced the concept of identity-based public key cryptosystem where public keys of users' are derived from their own unique identity information such as an email address [1]. In identity-based cryptosystem, private keys of users' are issued by a trusted authority called the PKG. The private key of a user is generated using the master key of the system. Therefore, PKG can inherently decrypt any ciphertext or forge signatures of any users'. In 2001, Boneh and Franklin proposed a first practical identity-based encryption scheme based on the Weil pairing [2]. Since then, most researches on identity-based cryptosystem are based on this system.

Identity-based two-party key agreement protocol was first proposed by Smart in 2001 [3]. This protocol is based on Boneh and Franklin's work and requires two pairing computation to compute the session key. However, Smart's protocol does not satisfy PKG forward secrecy. In 2002, Chen and Kudla introduced three new key agreement protocols. One of them extended Smart's protocol to provide PKG forward secrecy property. Another one extended Smart's protocol to multiple PKG environment where users who acquired their private keys from different PKG can share a common key [4]. However, in their setting, every PKG shares the common system parameters, but possess distinct master key.

Research on identity-based tripartite key agreement protocol was initiated by Joux in 2000 [5]. In 2003, Al-Riyami et al. pointed out that Joux's method does not support message authentication between participants, and proposed a new tripartite key agreement protocol that cures the drawback of Joux's [6]. A first identity-based tripartite key agreement protocol was introduced by Zhang et al. in 2002 [7]. In 2003, Shim also proposed another identity-based tripartite key agreement protocol, which requires less computation than Zhang et al.'s [8]. In 2004, Cheng et al. proposed a new identity-based tripartite key agreement protocol which is different from Shim's and Zhang et al.'s system [9]. However, this protocol has a serious flaw that allows an adversary to acquire the private key of a user easily.

1.2 Our Contribution

In identity-based cryptosystems, users acquire their private key from the PKG. A single PKG may be responsible for issuing private keys to members of a small-scale organization, but it is unrealistic to assume that a single PKG will be responsible for issuing private keys to members of different organizations, let alone the entire nation or the entire world. Furthermore, it is also unrealistic to assume that different PKGs will share common system parameters and differ only in the master key as done by Chen and Kudla. Therefore, we must consider multiple PKG environment where all the PKGs use different system parameters.

To date, most of the identity-based key agreement protocols are based on a single PKG environment [3, 4, 5, 6, 7, 8, 9, 10]. In order to extend these protocols to a setting where multiple PKGs exist, there should be some way to combine entities' contribu-

tion from different settings into a single common value. The most obvious solution to accomplish this is to combine the results of separate key agreement protocols executed in each PKG environment. However, since each entity only has its private key from its PKG, direct execution of existing protocol is infeasible. To ameliorate this situation, in this paper, we propose a new efficient ID-based two-party key agreement protocol for multiple PKG environment. We also extend this protocol to a tripartite version. Our two-party key agreement protocol is as efficient as any previous ID-based two-party key agreement protocol that provides PKG forward secrecy.

2 Security Attributes of Key Agreement Protocol

The followings are the security requirements of key agreement protocols, some of which are specific to ID-based key agreement protocol.

Known-key security: Each run of the key agreement protocol should generate a unique and independent session key. An adversary must have non-negligible advantage on compromising future session keys, even though it compromised past session keys.

Forward secrecy: An adversary must have non-negligible advantage on compromising past session keys, even though it compromised long-term private keys of one or more participants. The notion of forward secrecy can be further extended to the following two types of secrecy.

- **Perfect forward secrecy:** The forward secrecy must be preserved even if long-term private keys of all the participants involved are compromised.
- **PKG forward secrecy:** The forward secrecy must be preserved even if the master key of the PKG is compromised.

Key-compromise resilience: An adversary must have non-negligible advantage on impersonating others to A , even if it has compromised A 's private key.

Unknown key share resilience: An adversary must have non-negligible advantage on coercing others into sharing a key with other entities when it is actually sharing with a different entity.

Key control: An adversary must have non-negligible advantage on forcing the session key to be a preselected value.

The PKG forward secrecy is a stronger notion of perfect forward secrecy that applies only to ID-based protocols. In ID-based cryptosystems, if the master key is compromised, an adversary can compute all the participants' private keys. Therefore, If ID-based key agreement protocol satisfies PKG forward secrecy, then it also satisfies perfect forward secrecy too. However, the opposite is not true. We can also define perfect PKG forward secrecy in multiple PKG environment. This secrecy implies that forward secrecy is preserved even if all the master keys of PKGs are compromised.

3 Mathematical Background

3.1 Pairings

Bilinear pairings such as Weil pairing and Tate pairing reduces the discrete logarithm problem on elliptic curves to that in a finite field. Originally pairings were introduced

as a tool that can be used to attack cryptosystems based on elliptic curves. Using pairings, decision Diffie-Hellman problem on elliptic curves can be easily solved. This is a well-known MOV (Menezes, Okamoto, Vanstone) reduction [11] and FR (Frey, Ruck) attack [12]. In recent years, bilinear pairings have found positive applications in cryptography to construct new cryptographic primitives. In 2000, Joux showed that the Weil pairing can be used to construct a simple tripartite Diffie-Hellman key agreement protocol [5]. Since then, most of identity-based primitives exploit pairing to achieve their goals.

From now on, we will use the following notations: 1) q means a large prime number, 2) \mathbb{G}_1 and \mathbb{G}_2 are two groups with the same order q , where \mathbb{G}_1 is an additive group on an elliptic curve, and \mathbb{G}_2 is a multiplicative group of a finite field, 3) P , Q , and R are random elements of \mathbb{G}_1 , and 4) a , b , x , y , and z are random elements of \mathbb{Z}_q^* .

Definition 1 (Admissible Bilinear Map). A map $e : G_1 \times G_2 \rightarrow G_2$ is an admissible bilinear map only if it satisfies the following properties:

- **Bilinear:** Given P , Q , and R , the followings hold.
 - $e(P + Q, R) = e(P, R) \cdot e(Q, R)$
 - $e(P, Q + R) = e(P, Q) \cdot e(P, R)$
 This property also implies the followings:
 $e(aP, bQ) = e(P, bQ)^a = e(aP, Q)^b = e(P, Q)^{ab} = e(abP, Q) = e(p, abQ)$.
- **Non-degenerate:** If P and Q are not identity elements of \mathbb{G}_1 , then $e(P, Q) \neq O$, where O is an identity element of \mathbb{G}_2 .
- **Computable:** There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$

3.2 Cryptographic Problems

Definition 2 (Discrete Logarithm Problem (DLP) in \mathbb{G}_1). DLP is as follow: Given $\langle P, xP \rangle$, compute $x \in \mathbb{Z}_q$.

Definition 3 (Computational Diffie-Hellman Problem (CDHP) in \mathbb{G}_1). CDHP is as follow: Given $\langle P, xP, yP \rangle$, compute $xyP \in \mathbb{G}_1$.

Definition 4 (Bilinear Diffie-Hellman Problem (BDHP) in \mathbb{G}_1 and \mathbb{G}_2). BDHP is as follow: Given $\langle P, xP, yP, zP \rangle$, compute $e(P, P)^{xyz} \in \mathbb{G}_2$.

Currently, solving DLP, CDHP, and BDHP is computationally infeasible. For more detail, refer to [2].

4 The Protocols

4.1 System Setup

Basically, the system setup phase is similar to that of Boneh and Franklin's work. However, in our system, there are total n different PKGs, which do not share common system parameters. Therefore, each PKG must configure its parameters as follows:

- Each PKG_i chooses its basic system parameter: $\langle \mathbb{G}_1^{(i)}, \mathbb{G}_2^{(i)}, e^{(i)} \rangle$, where $\mathbb{G}_1^{(i)}$ is an additive group of order $q^{(i)}$, $\mathbb{G}_2^{(i)}$ is a multiplicative group of order $q^{(i)}$, and $e^{(i)}$ is admissible bilinear map between $\mathbb{G}_1^{(i)}$ and $\mathbb{G}_2^{(i)}$.
- PKG_i chooses $P^{(i)}$, a random generator of $\mathbb{G}_1^{(i)}$. It also chooses cryptographic hash functions $H_1^{(i)} : \{0, 1\}^* \rightarrow \mathbb{G}_1^{(i)}$ and $H_2^{(i)} : \mathbb{G}_2^{(i)} \rightarrow \{0, 1\}^k$, where k is a length of partial session key.
- Finally, PKG_i chooses its master key $s^{(i)} \in \mathbb{Z}_{q^{(i)}}^*$ randomly. It also computes its public key $P_{\text{pub}}^{(i)} = s^{(i)}P^{(i)}$.

After completing the system setup phase, each PKG publishes its public system parameters:

$$\langle \mathbb{G}_1^{(i)}, \mathbb{G}_2^{(i)}, P^{(i)}, P_{\text{pub}}^{(i)}, H_1^{(i)}, H_2^{(i)}, e^{(i)} \rangle.$$

4.2 Identity-Based Two-Party Key Agreement Protocol

In this section, we will introduce a new key agreement protocol between two entity A and B , each of which have acquired its private key from PKG_1 and PKG_2 , respectively. We assume that A and B have chosen their ephemeral key $a \in \mathbb{Z}_{q^1}^*$ and $b \in \mathbb{Z}_{q^2}^*$ respectively. Then, the protocol runs as follows:

$$\begin{aligned} \text{Message 1: } A \rightarrow B: & T_{AB}^{(2)} = a^{(2)}P^{(2)}, W_A^{(1)} = a^{(1)}P_{\text{pub}}^{(1)} \\ \text{Message 2: } B \rightarrow A: & T_{BA}^{(1)} = b^{(1)}P^{(1)}, W_B^{(2)} = b^{(2)}P_{\text{pub}}^{(2)} \end{aligned}$$

Protocol 1

After the messages are exchanged, each participant computes the two partial session keys as follows:

- A computes partial session keys, $K_{AB}^{(1)} = e^{(1)}(a^{(1)}S_A^{(1)}, T_{BA}^{(1)})$, and $K_{AB}^{(2)} = e^{(2)}(a^{(2)}Q_B^{(2)}, W_B^{(2)})$.
- B computes partial session keys, $K_{BA}^{(1)} = e^{(1)}(b^{(1)}Q_A^{(1)}, W_A^{(1)})$, and $K_{BA}^{(2)} = e^{(2)}(b^{(2)}S_B^{(2)}, T_{AB}^{(2)})$.

Now, each entity uses the two partial session key to compute the common session key. In detail, A computes the common session key $SK_{AB} = H(H_2^{(1)}(K_{AB}^{(1)}), H_2^{(2)}(K_{AB}^{(2)}))$, where H is a general hash function such as SHA-1. Similarly, B computes the session key $SK_{BA} = H(H_2^{(1)}(K_{BA}^{(1)}), H_2^{(2)}(K_{BA}^{(2)}))$. We can show that both participant have agreed on the same session key $SK = SK_{AB} = SK_{BA}$ by the followings:

$$\begin{aligned}
K_{AB}^{(1)} &= e^{(1)}(a^{(1)}S_A^{(1)}, T_{BA}^{(1)}) & K_{AB}^{(2)} &= e^{(2)}(a^{(2)}Q_B^{(2)}, W_B^{(2)}) \\
&= e^{(1)}(a^{(1)}s^{(1)}Q_A^{(1)}, b^{(1)}P^{(1)}) & &= e^{(2)}(a^{(2)}Q_B^{(2)}, b^{(2)}P_{\text{pub}}^{(2)}) \\
&= e^{(1)}(Q_A^{(1)}, P^{(1)})^{a^{(1)}s^{(1)}b^{(1)}} & &= e^{(2)}(a^{(2)}Q_B^{(2)}, b^{(2)}s^{(2)}P^{(2)}) \\
&= e^{(1)}(b^{(1)}Q_A^{(1)}, a^{(1)}s^{(1)}P^{(1)}) & &= e^{(2)}(Q_B^{(2)}, P^{(2)})^{a^{(2)}b^{(2)}s^{(2)}} \\
&= e^{(1)}(b^{(1)}Q_A^{(1)}, a^{(1)}P_{\text{pub}}^{(1)}) & &= e^{(2)}(b^{(2)}s^{(2)}Q_B^{(2)}, a^{(2)}P^{(2)}) \\
&= e^{(1)}(b^{(1)}Q_A^{(1)}, W_A^{(1)}) & &= e^{(2)}(b^{(2)}S_B^{(2)}, T_{AB}^{(2)}) \\
&= K_{BA}^{(1)}, & &= K_{BA}^{(2)}.
\end{aligned}$$

4.3 ID-Based Tripartite Key Agreement Protocol

In this subsection, we introduce a new tripartite key agreement protocol which is evolved from our two-party key agreement protocol. We assume that there are three participants A , B , and C , each of which have acquired its private key from PKG_1 , PKG_2 , and PKG_3 , respectively. We also assume that A , B , and C have chosen their ephemeral key $a \in \mathbb{Z}_{q^{(1)}}^*$, $b \in \mathbb{Z}_{q^{(2)}}^*$, and $c \in \mathbb{Z}_{q^{(3)}}^*$, respectively.

The First Round. The protocol is divided into two discrete rounds. In the first round, each entity constructs separate secure channel between others. To achieve this goal, every entity exploits our two-party key agreement protocol with the others individually. First of all, each participant broadcast messages to the others as follows:

<p>Message 1: $A \rightarrow B, C$: $T_{AB}^{(2)} = a^{(2)}P^{(2)}, T_{AC}^{(3)} = a^{(3)}P^{(3)}, W_A^{(1)} = a^{(1)}P_{\text{pub}}^{(1)}$ Message 2: $B \rightarrow C, A$: $T_{BA}^{(1)} = b^{(1)}P^{(1)}, T_{BC}^{(3)} = b^{(3)}P^{(3)}, W_B^{(2)} = b^{(2)}P_{\text{pub}}^{(2)}$ Message 3: $C \rightarrow A, B$: $T_{CA}^{(1)} = c^{(1)}P^{(1)}, T_{CB}^{(2)} = c^{(2)}P^{(2)}, W_C^{(3)} = c^{(3)}P_{\text{pub}}^{(3)}$</p>

First Round of Protocol 2

After broadcasting, each entity computes partial session keys. In detail, A computes partial keys K_{AB} , which is used to construct secure channel between A and B , and K_{AC} , which is also used to construct secure channel between A and C .

$$\begin{aligned}
K_{AB} &= H(H_2^{(2)}(e^{(2)}(a^{(2)}Q_B^{(2)}, W_B^{(2)})), H_2^{(1)}(e^{(1)}(a^{(1)}S_A^{(1)}, T_{BA}^{(1)}))) \\
K_{AC} &= H(H_2^{(3)}(e^{(3)}(a^{(3)}Q_C^{(3)}, W_C^{(3)})), H_2^{(1)}(e^{(1)}(a^{(1)}S_A^{(1)}, T_{CA}^{(1)})))
\end{aligned}$$

Similarly, B also computes its partial session keys, which is used for building secure channel with A and C , respectively.

$$\begin{aligned}
K_{BA} &= H(H_2^{(1)}(e^{(1)}(b^{(1)}Q_A^{(1)}, W_A^{(1)})), H_2^{(2)}(e^{(2)}(b^{(2)}S_B^{(2)}, T_{AB}^{(2)}))) \\
K_{BC} &= H(H_2^{(3)}(e^{(3)}(b^{(3)}Q_C^{(3)}, W_C^{(3)})), H_2^{(2)}(e^{(2)}(b^{(2)}S_B^{(2)}, T_{CB}^{(2)})))
\end{aligned}$$

Finally, C computes its partial session keys.

$$\begin{aligned}
K_{CA} &= H(H_2^{(1)}(e^{(1)}(c^{(1)}Q_A^{(1)}, W_A^{(1)})), H_2^{(3)}(e^{(3)}(c^{(3)}S_C^{(3)}, T_{AC}^{(3)}))) \\
K_{CB} &= H(H_2^{(2)}(e^{(2)}(c^{(2)}Q_B^{(2)}, W_B^{(2)})), H_2^{(3)}(e^{(3)}(c^{(3)}S_C^{(3)}, T_{BC}^{(3)})))
\end{aligned}$$

At the end of this phase, each entity can compute pairwise keys with other entities, separately as being introduced in our two-party key agreement protocol.

The Second Round. In the second round, each entity exchanges some values with others in order to share a common secret key. First of all, each entity broadcasts some values, each of which includes the ephemeral key of each entity's. Protocol runs as follows:

Message 4: $A \rightarrow B, C: \{a^{(1)}P^{(1)}\}_{K_{AB}}, \{a^{(1)}P^{(1)}\}_{K_{AC}}$
 Message 5: $B \rightarrow C, A: \{b^{(2)}P^{(2)}\}_{K_{BA}}, \{b^{(2)}P^{(2)}\}_{K_{BC}}$
 Message 6: $C \rightarrow A, B: \{c^{(3)}P^{(3)}\}_{K_{CA}}, \{c^{(3)}P^{(3)}\}_{K_{CB}}$

Second Round of Protocol 2

After broadcasting, A computes a common session key SK as follows:

$$\begin{aligned} K_{ABC}^{(1)} &= e^{(1)}(b^{(1)}P^{(1)}, c^{(1)}P^{(1)})^{a^{(1)}} = e^{(1)}(P^{(1)}, P^{(1)})^{a^{(1)}b^{(1)}c^{(1)}} \\ K_{ABC}^{(2)} &= e^{(2)}(b^{(2)}P^{(2)}, c^{(2)}P^{(2)})^{a^{(2)}} = e^{(2)}(P^{(2)}, P^{(2)})^{a^{(2)}b^{(2)}c^{(2)}} \\ K_{ABC}^{(3)} &= e^{(3)}(b^{(3)}P^{(3)}, c^{(3)}P^{(3)})^{a^{(3)}} = e^{(3)}(P^{(3)}, P^{(3)})^{a^{(3)}b^{(3)}c^{(3)}} \end{aligned}$$

$$SK = H(H_2^{(1)}(K_{ABC}^{(1)}), H_2^{(2)}(K_{ABC}^{(2)}), H_2^{(3)}(K_{ABC}^{(3)}))$$

B also computes a common session key SK as follows:

$$\begin{aligned} K_{ABC}^{(1)} &= e^{(1)}(aP^{(1)}, cP^{(1)})^b = e^{(1)}(P^{(1)}, P^{(1)})^{abc} \\ K_{ABC}^{(2)} &= e^{(2)}(aP^{(2)}, cP^{(2)})^b = e^{(2)}(P^{(2)}, P^{(2)})^{abc} \\ K_{ABC}^{(3)} &= e^{(3)}(aP^{(3)}, cP^{(3)})^b = e^{(3)}(P^{(3)}, P^{(3)})^{abc} \\ SK &= H(H_2^{(1)}(K_{ABC}^{(1)}), H_2^{(2)}(K_{ABC}^{(2)}), H_2^{(3)}(K_{ABC}^{(3)})) \end{aligned}$$

Finally, C compute a common session key SK as follows:

$$\begin{aligned} K_{ABC}^{(1)} &= e^{(1)}(aP^{(1)}, bP^{(1)})^c = e^{(1)}(P^{(1)}, P^{(1)})^{abc} \\ K_{ABC}^{(2)} &= e^{(2)}(aP^{(2)}, bP^{(2)})^c = e^{(2)}(P^{(2)}, P^{(2)})^{abc} \\ K_{ABC}^{(3)} &= e^{(3)}(aP^{(3)}, bP^{(3)})^c = e^{(3)}(P^{(3)}, P^{(3)})^{abc} \\ SK &= H(H_2^{(1)}(K_{ABC}^{(1)}), H_2^{(2)}(K_{ABC}^{(2)}), H_2^{(3)}(K_{ABC}^{(3)})) \end{aligned}$$

5 Analysis

In this section, we give the security and efficiency analysis of our proposed protocols. We first heuristically argue that our protocols satisfy the security requirements of the key agreement protocol. We then discuss the efficiency of our protocols by comparing the number of pairing computation required with other protocols.

5.1 Security Analysis

We only discuss the security of two-party key agreement protocol. If an adversary can obtain the session key of two-party agreement protocol, the adversary can decrypt the ciphertext exchanged during message 4 to 6 of tripartite protocol. However, if it is computationally infeasible for the adversary to obtain any one of the ephemeral key a , b , and c , it is also infeasible to obtain $K_{ABC}^{(i)}$ due to the difficulty of BDHP. It is computationally infeasible to obtain the ephemeral key from the publicly available information due to the difficulty of DLP.

1. **Man-in-the-middle-attack:** This kind of attack can be foiled if the origin authentication of values exchanged can be provided. Although, origin authentication is not provided, the way the final session key is computed prevents this kind of attack. If an attacker intercepts the two messages and sends the following to A:

$$T_{CA}^{(1)} = cP^{(1)}, W_C^{(2)} = cP_{pub}^2,$$

the computed partial key will be as follows:

$$K_{AC}^{(1)} = e^{(1)}(aS_A^{(1)}, T_{CA}^{(1)}) = e^{(1)}(Q_A^{(1)}, P^{(1)})^{acs^{(1)}},$$

$$K_{AC}^{(2)} = e^{(2)}(aQ_B^{(2)}, W_C^{(2)}) = e^{(2)}(Q_B^{(2)}, P^{(2)})^{acs^{(2)}}.$$

Although, the attacker can compute $K_{AC}^{(1)}$, it is infeasible for the attacker to compute $K_{AC}^{(2)}$ without acquiring ephemeral key a of A, or the master key s^2 of PKG₂.

2. **Known-key security:** In our protocol, ephemeral keys such as a , b , and c are used to construct the session key. As a result, each run of the protocol creates unique session key which is independent to past or future session keys. Therefore, compromise of past session keys do not affect the security of future session keys.
3. **PKG forward secrecy:** To satisfy PKG forward secrecy, the compromise of master key of PKG₁ and PKG₂ must not affected the security of past session keys. The past session key can be computed if the corresponding partial session keys $K_{AB}^{(1)}$ and $K_{BA}^{(2)}$ can be computed. If we assume that adversary knows $s^{(1)}$, $K_{AB}^{(1)} = e^{(1)}(Q_A^{(1)}, P^{(1)})^{as^{(1)}b}$ can be computed if the adversary can compute $e^{(1)}(aQ_A^{(1)}, bP^{(1)})$, $e^{(1)}(bQ_A^{(1)}, aP^{(1)})$, $e^{(1)}(Q_A^{(1)}, abP^{(1)})$, or $e^{(1)}(abQ_A^{(1)}, P^{(1)})$. Without acquiring ephemeral key a or b , it is infeasible to obtain any of these values from the publicly available information. The same argument also applies to $K_{BA}^{(2)}$.
4. **Key-compromise resilience:** Since both parties private key is needed to compute the session key, the compromise of A's private key does not help the adversary to impersonate others to A.
5. **Unknown key-share resilience:** Since a party always uses the other party's authenticated public key as one of the input used to compute the session key, an adversary cannot deceive a party into falsely believing the identity of the other party in concern.
6. **Key control:** Since each party contributes a fresh ephemeral key as one of the input used to compute the session key, one of the party cannot force the session key to be some preselected value.

Table 1. Comparison of pairing computation in two-party key agreement protocol

protocol	In single PKG environment		In multiple PKG environment	
	pairing(each)	pairing(all)	pairing(each)	pairing(all)
Chen and Kudla's protocol	2	4	2*	4
Our protocol			2**	4

*: each PKGs shares the common system parameters but distinct master keys.
 **: each PKGs uses different system parameters.

Table 2. Comparison of pairing computation in tripartite key agreement protocol

protocol	In single PKG environment		In multiple PKG environment	
	pairing(each)	pairing(all)	pairing(each)	pairing(all)
Cheng et al's protocol	5	15	15	45
Our protocol 2			7	21

5.2 Efficiency Analysis

In this subsection, we compare the number of pairing computation performed by each user with other protocols. Since pairing computation out weigh other computations, the protocol that requires less pairing computation can be considered as a more efficient protocol.

In table 1, we compare our two party protocol with Chen and Kudla's protocol that provides PKG forward secrecy. We can see that the efficiency of our protocol is equal to Chen and Kudla's even though in our setting each PKGs uses different system parameters.

6 Conclusion

In this paper, we have proposed a new efficient ID-based two-party key agreement protocol for multiple PKG environment. We have also extended this protocol to a tripartite version. We have showed that our proposed protocols satisfy all the security requirements including the PKG forward secrecy. The efficiency of two-party key agreement protocol is equal to previous ID-based protocols for single PKG environment. The security of our proposed key agreement protocols are based on the difficulty of DLP, CDHP, and BDHP on an elliptic curve.

References

1. Shamir, A.: Identity-based Cryptosystems and Signature Schemes. In Advances in Cryptology, Crypto 1984. Lecture Notes in Computer Science, Vol. 196. Springer-Verlag (1985) 47–53

2. Boneh, D., Franklin, M.: Identity-based Encryption from Weil pairing. In *Advances in Cryptology, Crypto 2001*. Lecture Notes in Computer Science, Vol. 2139. Springer-Verlag (2001) 213–229
3. Smart, N.: An Identity-based Authenticated Key Agreement Protocol Based on Weil Pairing. In *Electronic Letters*, Vol. 38. IEEE (2002) 630–632
4. Chen, L., Kudla C.: Identity-based Authenticated Key Agreement Protocols from Pairings. In *Proceedings of the 16th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press (2003) 219–233
5. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. In *Proceedings of Algorithmic Number Theory Symposium, ANTS-IV*. Lecture Notes in Computer Science, Vol. 1838. Springer-Verlag (2000) 385–394
6. Al-Riyami, S., Patterson, K.: Tripartite Authenticated Key Agreement Protocols from Pairings. In *Proceedings of IMA Conference on Cryptography and Coding*. Lecture Notes in Computer Science, Vol. 2898. Springer-Verlag (2003) 332–359
7. Zhang, F., Liu, S., Kim, K.: ID-Based One Round Authenticated Tripartite Key Agreement Protocols with Pairings. *Cryptology ePrint Archive*, Report 2002/122
8. Shim, K.: Efficient One Round Tripartite Authenticated Key Agreement Protocol Based on Weil Pairing. In *Electronic Letters*, Vol. 39, IEEE (2003) 208–209
9. Cheng, Z., Vasiu, L., Comley, R.: Pairing-Based One-Round Tripartite Key Agreement Protocols. *Cryptology ePrint Archive*, Report 2004/079
10. Chen, L., Kudla, C.: Identity-based Authenticated Key Agreement Protocols from Pairings. *Cryptology ePrint Archive*, Report 2002/184
11. Menezes, A., Okamoto, T., Vanstone, S.: Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *Transaction of Information Theory*, Vol. 39. IEEE (1993) 1639–1646
12. Frey, G., Ruck, H.: A Remark Concerning m -divisibility and The Discrete Logarithm in The Divisor Class Group of Curves. *Mathematics of Computation*, Vol. 62. (1994) 865–874