

# A Novel Way of Issuing Multiple Private Keys in ID-based Cryptosystems\*

Donghyun Kim  
University of Texas at Dallas  
Department of Computer Science  
donghyunkim@student.udallas.edu

Sangjin Kim  
Korea Univ. of Technology and Education  
School of Internet Media Engineering  
Cheonan, Chungnam, Republic of Korea  
sangjin@kut.ac.kr

Heekuch Oh  
Hanyang University  
Department of Computer Science and Engineering  
Ansan, Gyeonggi, Republic of Korea  
hkoh@hanyang.ac.kr

## Abstract

*In this paper, we propose a new efficient model for issuing multiple private keys in IBC (Identity-Based Cryptosystem). In our model, the private key of a user is divided into two components: LTK (Long-Term Key) and STK (Short-Term Key). Both are issued separately by different parties. The LTK is issued in a threshold manner by KPAs (Key Privacy Agency). In contrast, the STK is issued by a single authority called KGC (Key Generation Center). A user can efficiently obtain a new private key containing the same identity information by contacting only the KGC and obtaining a new STK. We also give a security proof of the key issuing model and present a new IBE (Identity-Based Encryption) scheme based on our key issuing model.*

## 1 Introduction

The IBC is a public key cryptosystem where a user's public key is derived from his/her well-known ID (Identity). In 1984, Shamir introduced the concept of IBC that includes the idea of both IBE and IBS schemes [7]. However, Shamir only realized an IBS scheme based on the RSA assumption. Until Boneh and Franklin introduced an IBE scheme based on the Weil pairing in 2001 [3], there has been no fully satisfactory solution.

---

\*This research was supported by the MIC (Ministry of Information and Communication), Korea, under the HNRC (Home Network Research Center) - ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment)

## 1.1 Pros and Cons of IBC

The main concern in traditional PKI (Public Key Infrastructure) is providing a mechanism to authenticate public keys of users. To this end, a certificate, which is a digital signature that binds a public key and its owner, is used. On the other hand, IBCs do not require public key certificates to authenticate users' public keys. This is because, in IBC, a user's public key can be computed by anyone using the user's well-known ID such as an e-mail address without contacting the owner or third parties. Therefore, IBCs avoid most of certificate management problems in PKI. Nonetheless, IBCs cannot be considered as an alternative solution to the traditional PKI yet for the following reasons.

- Since private keys of users are created by the PKG (Private Key Generator), key escrow is inherent in this system. Key escrow is a useful property that can be used to prevent crimes or recover lost keys. However, there must be a consideration about how to balance the protection of privacy of individuals with the needs of law enforcement [2].
- There is no suitable private key revocation mechanism for IBCs. Current mechanisms using CRL (Certificate Revocation List) do not suit well to IBCs.
- A user of this system must authenticate him/herself to the PKG to obtain his/her private key, which should not be transmitted through a public channel.
- An efficient way to distribute authenticated public system parameters of the PKGs is required. Especially, this matters more when we assume the existence of multiple PKGs that use different parameters.

## 1.2 The Key Escrow Property of IBC

To date, several solutions to the inherent key escrow problem of IBCs have been proposed [3, 5, 1]. The most obvious solution to the key escrow problem is to use multiple PKGs in a threshold manner like Boneh and Franklin [3]. In their system, the master key, used to create private keys, is secretly shared between PKGs in a threshold manner. While this approach resolves the key escrow problem gracefully, it requires a user to contact several PKGs to obtain his/her private key. Thus, it increases the authentication, computation, and communication costs.

Gentry tried totally different approach to solve the key escrow problem [5]. He solved it by using some user chosen random secret. However, in this scheme, key escrow capability was totally removed. Moreover, it cannot be regarded as a true identity-based scheme because a user's public key cannot be obtained directly from the ID of the user. Al-Riyami and Paterson extended Gentry's idea and provided a scheme which preserves certificateless property of IBC [1]. However, they only provided implicit authentication of the public key. Therefore, each user cannot be sure whether the public key is genuine or not.

## 1.3 Multiple Private Keys for Users

In the initial Shamir's proposal of IBCs [7], a well-known human readable string such as e-mail address was considered as the ID of a user. Using this kind of ID as a public key has several advantages over conventional public key, a simple binary string. First, a sender can easily obtain or already know the ID of a receiver, from which the sender can derive the authenticated public key of the receiver on his/her own. Second, one can distribute his/her ID without requiring a complicated infrastructure. Finally, as certificates contain more than just the name of the owner, ID used in IBC can contain some additional useful information.

Boneh and Franklin proposed a form of ID, which consists of an e-mail address plus some user related information such as user's duty, capability, or lifetime of a private key [3]. As a result, an ID of a user becomes more expressive, and it is possible for each user to have multiple IDs and corresponding private keys that contain identical identity information but include different additional information.

We must note that if immutable user identity information such as social security number is used solely as the ID, the master key or the user identity information must be changed when the private key has to be reissued, which are both impractical. Moreover, in previous IBCs with limited key escrow, there is a trade off between the cost of issuing a private key and the degree of key escrow limit [3, 6].

**Table 1. Inputs used to construct a public key**

	LTK	STK
1.	foo@x.edu  2007	foo@x.edu  2007  Role1
2.	foo@x.edu  2007	foo@x.edu  2007  20070831
3.	foo@x.edu  2007	foo@x.edu  2007  Role1  20070831

## 1.4 Private Key Revocation Problem

In traditional PKI, up-to-date CRL (Certificate Revocation List) is maintained and users must check the status of a certificate against this list, which may be very large, before using it. This is done by making queries or requesting the CRL itself to a third-party. A revoked certificate is maintained on the CRL until the intended expiration date is over.

Gentry introduced a new time-slot approach to solve the key revocation problem in certificate-based cryptosystems [5]. This approach is based on revoking the owner of the key instead of the key itself. When a certificate is expired, the CA (Certification Authority) would issue a new certificate if and only if the owner was not revoked. However, there are two drawbacks. First, certificates of legitimate users have to be issued per each time-slot. Second, a certificate cannot be revoked in the middle of each time-slot. Gentry has ameliorated each problem using the hierarchical approach and the frequency of the certificates renewal, respectively.

Likewise, in IBCs, a user's private key may need to be revoked for some reasons. Currently, there are no efficient revocation mechanisms for IBC. Trivial solution for this problem would be to use the mechanisms that are used in PKI such as CRL. However, to use mechanisms such as CRL, it requires third party queries, which offset the main advantage of IBCs of obtaining authenticated public keys of users without contacting another party.

## 1.5 The Need of PKI Support

In IBCs, the PKG must authenticate the user before issuing the user's private key. This requirement cannot be satisfied using the IBC alone. To this end, most of the IBCs assume that PKI is used for this purpose [3]. Moreover, the user's private key must be transmitted securely to the user. Boneh and Franklin assume that PKI is also used for this purpose [3]. However, as shown by Lee et al. [6], simple blinding technique can be used to send the private key securely without using any other cryptosystems. Currently, the distribution of public system parameters also needs support from PKI to authenticate the parameters.

**Table 2. Notation**

Notation	Description
$q$	a large prime number
$\mathbb{G}_1$	additive group of order $q$
$\mathbb{G}_2$	multiplicative group of order $q$
$\mathbb{Z}_q^*$	a multiplicative group of order $q$
$P, Q, R$	random generators of $\mathbb{G}_1$
$\alpha, \beta, \text{ and } \gamma$	randomly distributed elements of $\mathbb{Z}_q^*$

## 2 Backgrounds

Throughout this paper, we will use the notations given in Table 2.

**Definition 1** (Discrete Logarithm Problem (DLP) in  $\mathbb{G}_1$ ). *DLP in  $\mathbb{G}_1$  is as follow: Given  $\langle P, \alpha P \rangle$ , acquire  $\alpha$ . An algorithm  $\mathcal{A}$  has advantage  $\epsilon$  in solving DLP in  $\mathbb{G}_1$ , if*

$$\Pr [\mathcal{A}(P, \alpha P) = \alpha] \geq \epsilon.$$

**Definition 2** (Admissible Bilinear Map).  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$  is an admissible bilinear map, if  $\hat{e}$  has the following properties.

- **Bilinear:** Given  $P, Q, R \in \mathbb{G}_1$ , the following holds:  
 $\hat{e}(P, Q + R) = \hat{e}(P, Q) \cdot \hat{e}(P, R)$  and  
 $\hat{e}(P + Q, R) = \hat{e}(P, R) \cdot \hat{e}(Q, R)$ .
- **Non-degenerate:**  $\hat{e}(P, Q) \neq O$  for some  $P, Q \in \mathbb{G}_1$ , where  $O$  is an identity element of  $\mathbb{G}_1$ .
- **Computable:** There is an efficient algorithm to compute  $\hat{e}(P, Q)$  for any  $P, Q \in \mathbb{G}_1$ .

Bilinear property of admissible bilinear map also implies the following:

$$\hat{e}(\alpha P, \beta Q) = \hat{e}(\alpha P, Q)^\beta = \hat{e}(P, \beta Q)^\alpha = \hat{e}(P, Q)^{\alpha\beta}.$$

Admissible bilinear map can be constructed using the Weil or the Tate pairing on an elliptic curve over a finite field. For more detail, refer to [3].

**Definition 3** (Computational Diffie-Hellman Problem (CDHP)). *CDHP in  $\mathbb{G}_1$  is as follow: Given  $\langle P, \alpha P, \beta P \rangle$ , compute  $\alpha\beta P$ . An algorithm  $\mathcal{A}$  has advantage  $\epsilon$  in solving CDHP in  $\mathbb{G}_1$ , if  $\Pr [\mathcal{A}(P, \alpha P, \beta P) = \alpha\beta P] \geq \epsilon$ .*

**Definition 4** (BDH Parameter Generator). *A randomized algorithm  $\mathcal{G}$  is called a BDH parameter generator, if it satisfies the following properties.*

- It takes a single security parameter  $k \geq 1$ .
- It runs in polynomial time in  $k$ .

- It outputs the description of group  $\mathbb{G}_1$  and  $\mathbb{G}_2$  of order  $q$  and an admissible bilinear map  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ .

In this paper, we assume that the advantage on DLP and CDHP is both negligible with respect to above definitions. For more information, refer to [3].

## 3 A New Private Key Issuing Model

### 3.1 The Participants

There are four types of entities in our model and their roles are as follows.

- **KPAs:** KPAs are responsible for issuing LTK shares for users. A user has to contact at least  $t$  KPAs of total  $n$  KPAs to obtain his/her LTK. A user receives a share of his/her LTK from each KPA through a public channel and constructs his/her LTK in a threshold manner.
- **KGC:** The KGC is responsible for issuing STKs for users. A user sends his/her identity, the blinded LTK, and dynamic short-term information to obtain the STK. The STK is issued through a public channel.
- **KCA:** The KCA (Key Commitment Agency) is responsible for observing KGC activity. To enforce this observation, we use the KCA as an intermediary between the user and the KGC. We also assume that the KCA monitors KGC's activity, which prevents users directly contacting the KGC.
- **User:** A user needs to obtain his/her LTK shares from  $t$  KPAs and his/her STK from the KGC to construct his/her private key of the system.

### 3.2 System Setup

In this phase, each authority determines its public/private keys and computes public system parameters cooperatively.

#### 3.2.1 KGC setup

The KGC carries out the following procedures.

- **Step 1.** Runs  $\mathcal{G}$  and obtains  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ .
- **Step 2.** Chooses a random generator  $P$  of  $\mathbb{G}_1$ .
- **Step 3.** Selects cryptographic hash functions  $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$  and  $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^l$ , where  $l$  is the length of the message block.
- **Step 4.** Selects its master key  $s \in \mathbb{Z}_q^*$  and sets  $P_{\text{KGC}} = sP \in \mathbb{G}_1^*$ .

### 3.2.2 KPAs setup

Each KPAs computes its own private and public keys. They also compute the private key and public key of KPAs cooperatively.

- **Step 1.** Each  $KPA_i$  generates its master key  $x_i \in \mathbb{Z}_q^*$ , ( $1 \leq i \leq n$ ) in a distributed fashion using the techniques of [4]. It also computes its public key  $P_{KPA_i} = x_i P \in \mathbb{G}_1^*$ .
- **Step 2.** The private key of KPAs is  $x = \sum_{i \in I} L_i x_i \in \mathbb{Z}_q^*$ , where  $L_i$  is appropriate Lagrange coefficient and  $I$  is a set of  $t$  numbers which are greater than or equal to 1 and less than or equal to  $n$ . The corresponding public key of KPAs is  $P_{KPAs} = xP \in \mathbb{G}_1^*$ .

The public parameters of this system is  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, l, P, P_{KGC}, P_{KPAs}, P_{KPA_1}, \dots, P_{KPA_n}, H_1, H_2 \rangle$ .

### 3.3 User Private Key Extraction

A user acquires his/her private key using the following two procedures: LTK extraction and STK extraction.

#### 3.3.1 LTK extraction

The user requests his/her LTK shares to  $t$  different  $KPA_i$ , ( $1 \leq i \leq n$ ) using the following steps. We must note that the user has to authenticate him/herself to each KPA involved in this procedure before proceeding the step 1. We assume PKI is used for this purpose.

- **Step 1.** The user chooses his/her random blinding factor  $b_i \in \mathbb{Z}_q^*$  and computes  $b_i Q_{ID} = b_i H_1(ID)$ .
- **Step 2.** The user sends  $\langle ID, b_i P, b_i Q_{ID} \rangle$  to  $KPA_i$ .
- **Step 3.** The  $KPA_i$  checks the identification of the user and verifies the validness of  $b_i Q_{ID}$  using the following equation:

$$\hat{e}(Q_{ID}, b_i P) \stackrel{?}{=} \hat{e}(b_i Q_{ID}, P). \quad (1)$$

- **Step 4.** The  $KPA_i$  computes the blinded LTK share  $b_i d_{ID}^{(i)} = b_i x_i Q_{ID} \in \mathbb{G}_1^*$ . It returns  $b_i d_{ID}^{(i)}$  to the user.
- **Step 5.** The user removes the blind factor by computing  $b_i^{-1} b_i d_{ID}^{(i)} = d_{ID}^{(i)}$  and verifies the correctness of  $d_{ID}^{(i)} = x_i Q_{ID}$  using the following equation:

$$\hat{e}(P, d_{ID}^{(i)}) \stackrel{?}{=} \hat{e}(P_{KPA_i}, Q_{ID}). \quad (2)$$

After collecting  $t$  different LTK shares, the user computes his/her LTK:

$$d_{ID} = \sum_{i \in I} L_i d_{ID}^{(i)} = \sum_{i \in I} L_i x_i Q_{ID} \in \mathbb{G}_1^*,$$

where  $L_i$  is the appropriate Lagrange coefficient. The user can check the correctness of his/her LTK using the following equation:

$$\hat{e}(P, d_{ID}) \stackrel{?}{=} \hat{e}(P_{KPAs}, Q_{ID}).$$

#### 3.3.2 STK extraction

The user requests the STK by sending  $\langle ID, T, X = bP_{KPAs}, Y = b d_{ID}, Z = bQ_{(ID,T)} \rangle$  to the KCA, where  $b \in \mathbb{Z}_q^*$  is a random blinding factor chosen by the user. The KCA forwards the request message of the user to the KGC without modifying it. The KGC would reject the request of the user, if the user's LTK  $d_{ID}$  has been revoked. Otherwise, the KGC computes the requested STK and sends it back to the KCA using the following steps.

- **Step 1.** The KGC checks whether the user has a valid LTK using the following equation:

$$\hat{e}(Q_{ID}, X) \stackrel{?}{=} \hat{e}(P, Y). \quad (3)$$

- **Step 2.** The KGC sets  $Q_{(ID,T)} = H_1(ID||T)$  and checks whether the user has sent a valid  $Z$  using the following equation:

$$\hat{e}(Q_{(ID,T)}, X) \stackrel{?}{=} \hat{e}(Z, P_{KPAs}). \quad (4)$$

- **Step 3.** The KGC sends the blinded STK  $sZ$  to the KCA. It also transmits the verifiably encrypted STK  $\langle V = sQ_{(ID,T)} + \bar{b}P_{KPAs}, \bar{V} = \bar{b}P \rangle$ , where  $\bar{b}$  is a random element of  $\mathbb{Z}_q^*$  to the KCA.

The KCA checks the correctness of  $\langle V, \bar{V} \rangle$  by using the prior registered request as follows:

$$\hat{e}(V, P) \stackrel{?}{=} \hat{e}(Q_{(ID,T)}, P_{KGC}) \hat{e}(\bar{V}, P_{KPAs}). \quad (5)$$

If  $\langle V, \bar{V} \rangle$  pair is correct, the KCA publishes  $\langle ID, T, V, \bar{V} \rangle$  and sends  $sZ$  to the user. The user computes his/her STK  $d_{(ID,T)}$  by eliminating the blinding factor as follows:  $b^{-1} sZ = b^{-1} s b Q_{(ID,T)} = s Q_{(ID,T)} = d_{(ID,T)}$ . The user also checks the correctness of  $d_{(ID,T)}$  using the following:

$$\hat{e}(P, d_{(ID,T)}) \stackrel{?}{=} \hat{e}(P_{KGC}, Q_{(ID,T)}). \quad (6)$$

Finally, the user computes his/her private key  $D_{(ID,T)} = d_{ID} + d_{(ID,T)} \in \mathbb{G}_1$ .

We must note that the introduction of KCA is inevitable to preserve the threshold property of our scheme. Without KCA, we must trust the KGC to correctly publish  $\langle ID, T, V, \bar{V} \rangle$  for each STK it issues.

### 3.3.3 Key Escrow

In the proposed scheme, a TTP (Trusted Third Party) which already knows  $\langle \text{ID}, T, V, \bar{V} \rangle$ , can recover a private key of a user by cooperating with more than  $t$  KPAs as follows.

- The TTP sends  $\bar{V} = \bar{b}P$  to more than  $t$  KPAs. Each  $\text{KPA}_i$  computes  $\bar{V}^{(i)} = x_i \bar{V} = x_i \bar{b}P$ , and sends it back to the TTP. The TTP then checks the validity of each  $\bar{V}^{(i)}$  as follows:

$$\hat{e}(\bar{V}, P_{\text{KPA}_i}) \stackrel{?}{=} \hat{e}(\bar{V}^{(i)}, P).$$

- The TTP computes  $\tilde{V} = \sum_{i \in I} L_i \bar{V}^{(i)} = \sum_{i \in I} L_i x_i \bar{b}P = \bar{b}xP = \bar{b}P_{\text{KPAs}}$ , and recovers the STK of a user using the following equation:

$$\begin{aligned} V - \tilde{V} &= sQ_{\langle \text{ID}, T \rangle} + \bar{b}P_{\text{KPAs}} - \bar{b}P_{\text{KPAs}} \\ &= sQ_{\langle \text{ID}, T \rangle} = d_{\langle \text{ID}, T \rangle}. \end{aligned}$$

- The TTP also requests LTK share of the user to more than  $t$  KPAs. This procedure is an analogous of LTK extraction phase. Finally, the TTP recovers the private key of the user  $D_{\langle \text{ID}, T \rangle} = d_{\text{ID}} + d_{\langle \text{ID}, T \rangle}$ .

### 3.4 A New Encryption Scheme

In this section, we present a new IBE scheme similar to the Boneh and Franklin's IBE scheme [3] using our model. In the proposed IBE scheme, a public key of a receiver is derived from  $\langle \text{ID}, T \rangle$  pair. The protocol runs as follows.

#### 3.4.1 Encryption

Let  $T$  be the dynamic information used in computing the STK of the user. We assume that the sender and the receiver have agreed on  $T$  in advance. The sender obtains the public key of the user by computing  $Q_{\text{ID}} = H_1(\text{ID}) \in \mathbb{G}_1^*$  and  $Q_{\langle \text{ID}, T \rangle} = H_1(\text{ID}||T) \in \mathbb{G}_1^*$ . Then the sender sets

$$g = \hat{e}(P_{\text{KPAs}}, Q_{\text{ID}}) \hat{e}(P_{\text{KGC}}, Q_{\langle \text{ID}, T \rangle}) \in \mathbb{G}_2$$

and selects a random secret  $r \in \mathbb{Z}_q^*$ . The resulting ciphertext of a message  $m \in \{0, 1\}^l$  is  $C = \langle rP, m \oplus H_2(g^r) \rangle \in \mathbb{G}_1^* \times \{0, 1\}^l$ .

#### 3.4.2 Decryption

The receiver can decrypt  $C = \langle U, V \rangle$  using his/her private key  $D_{\langle \text{ID}, T \rangle}$ :

$$m = V \oplus (H_2(\hat{e}(U, D_{\langle \text{ID}, T \rangle}))).$$

The correctness of the encryption scheme can be easily verified as follows:

$$\begin{aligned} \hat{e}(U, D_{\langle \text{ID}, T \rangle}) &= \hat{e}(U, xQ_{\text{ID}} + sQ_{\langle \text{ID}, T \rangle}) \\ &= \hat{e}(U, xQ_{\text{ID}}) \hat{e}(U, sQ_{\langle \text{ID}, T \rangle}) \\ &= \hat{e}(rP, xQ_{\text{ID}}) \hat{e}(rP, sQ_{\langle \text{ID}, T \rangle}) \\ &= \hat{e}(P, Q_{\text{ID}})^{xr} \hat{e}(P, Q_{\langle \text{ID}, T \rangle})^{sr} \\ &= \hat{e}(xP, Q_{\text{ID}})^r \hat{e}(sP, Q_{\langle \text{ID}, T \rangle})^r \\ &= \hat{e}(P_{\text{KPAs}}, Q_{\text{ID}})^r \hat{e}(P_{\text{KGC}}, Q_{\langle \text{ID}, T \rangle})^r \\ &= g^r. \end{aligned}$$

## 4 Analysis

### 4.1 Efficiency Analysis

Before proceeding in detail, we note that Boneh and Franklin's [3] and ours are based on threshold technique but Lee et al.'s [6] is not. Moreover, Boneh and Franklin's [3] and ours can communicate in parallel, while Lee et al.'s [6] must do in a sequential manner. This means that as the number of authorities increases, Lee et al.'s [6] will require more communication time.

For more precise and fair comparison, we assume that the threshold variable  $t$  of both Boneh and Franklin's [3] and ours are equal to the number of KPAs in Lee et al.'s [6]. In other words, we consider the setting where the number of authorities used in ours is equal to that of Lee et al.'s [6], but one more than that of Boneh and Franklin's [3] without considering the KCA.

- **Pairing computation:** As shown in Table 3, Boneh and Franklin's [3] requires the least pairing computations during the initial key issuance. However if the method used in our scheme to remove the secure channel assumption is applied to Boneh and Franklin's [3], the computation cost of Boneh and Franklin's [3] becomes  $4t + 2$ . In this setting, we can conclude that ours requires 11 more pairing computation than Boneh and Franklin's [3] and 9 more pairing computation than Lee et al.'s [6] during the initial key issuance. These additional computations are due to the STK extraction and checking the validity of the verifiable encrypted STK by the KCA. However, in our scheme, the computation cost of reissuing a private key is constant.
- **Communication:** Boneh and Franklin's [3] requires the least communication cost. Our protocol requires 4 more communication than Boneh and Franklin's [3] which is due to the STK extraction where the KCA plays an intermediary role between the user and the KGC. Additionally, Lee et al.'s [6] will need extra communication time because of its sequential property. In our scheme, the communication cost of reissuing a private key is also constant.

**Table 3. Comparison of Private Key Issuance Cost**

	Initial Issuance Cost			Reissuance Cost		
	[3]	[6]	Ours	[3]	[6]	Ours
Pairing computation	$2t + 2^*$	$4t + 4$	$(4t + 4) + 9$	$2t + 2^*$	$4t + 4$	9
Communication	$2t$	$2t + 2$	$2t + 4$	$2t$	$2t + 2$	4
Authentication	$t$	1	$t$	$t$	1	0

$t$  denotes the number of PKGs participating in [3] and the number of KPAs participating in [6] and ours.

\* [3] requires secure channels whereas [6] and ours do not. If the method used in our system to remove the secure channel assumption is applied to [3], the cost becomes  $4t + 2$ .

- **Authentication:** Lee et al.'s [6] needs the least authentication. However, in our scheme, authentication is not required when key is being reissued.
- **Key Escrow Procedure:** In Boneh and Franklin's [3], key escrow procedure is done by doing usual secret recovering protocol in the threshold scheme which does not require pairing computation. Our scheme requires twice as much amount of computation as Boneh and Franklin's [3] does assuming that the committed STK is known. Since ours and Boneh and Franklin's [3] are based on the traditional threshold concept, any  $t$  out of  $n$  authorities can recover the key. This procedure does not require sequential connection. On the other hand, in Lee et al.'s [6], the private key is recovered in a sequential manner by all the authorities originally participated in the key issuing protocol. This means that Lee et al.'s [6] needs relatively more time to recover the private key than others and all authorities must participate.

From these, we can conclude that our scheme outperforms others with respect to reissuance, and performs nearly equal to others with respect to initial issuance.

## 4.2 Security Analysis

There are four types of entities in our model:  $n$  KPAs, KGC, KCA, and users, all of whom we can consider as latent adversaries. We also have to consider a malicious third party, who does not participate in the key issuing procedure as a legitimate party, as a potential adversary. The goals of these adversary are as follows: 1) disturb the key issuing phase, 2) acquire a LTK of a user corresponding to an ID, 3) obtain a STK of a user related to  $\langle ID, T \rangle$ , and 4) acquire the private key of a user.

- **Malicious third party:** An adversary that does not participate as one of the legal participants are called a malicious third party. These adversaries can eavesdrop or modify messages which are sent through public communication channels.

- **KCA:** The KCA is supplementary authority supporting threshold property in our key issuing model. Since all the LTKs and STKS passing through the KCA are blinded, KCA cannot gain any explicit knowledge about any STKS or LTKs. As a result, the KCA only has ability as much as the malicious third party does.
- **KPAs:** If the threshold cryptosystem is correct and secure, it is infeasible for less than  $t$  KPAs to compute the correct LTK corresponding to an ID. Therefore, in this respect, although they can compute a share of the LTK, they have the same amount of ability as the malicious third party does.
- **KGC:** In our model, the KGC knows the master key  $s$  and can compute STKS corresponding to any  $\langle ID, T \rangle$  pairs of users. This means that the KGC not only has ability as much as the malicious third party does but also can compute any STK it wants.
- **Users:** A legitimate user can possess correct STKS and LTKs related to his/her own  $\langle ID, T \rangle$  pairs. He/she can also do whatever the malicious third party can.

From previous discussion, we can conclude the followings: 1) proposed key issuing protocol must not suffer from message modification, insertion or replay attack, 2) the LTK can be computed only if more than  $t$  KPAs cooperate, 3) the KGC has to be the only entity that can compute the correct STK, and 4) except the legitimate user, no other entities can acquire the private key of the user.

**Definition 5.** *The security requirements of our key issuing protocol is as follows.*

- **Correctness:** *Incorrect response from any entity, message modification during transmission, replying attack by using old messages, or incorrect packet insertion cannot make any harm to our protocol. Especially, a user must be able to determine whether a private key, issued through our key issuing model, is correct or not.*
- **Unforgeability:** *It has to be impossible to compute the correct LTK related to an ID without the cooperation*

of more than  $t$  KPAs. Moreover, the KGC is the only entity that can compute the correct STK of  $\langle \text{ID}, T \rangle$ .

- **Privacy:** Except the owner of the ID, no other entities can acquire the private key corresponding to  $\langle \text{ID}, T \rangle$  unless more than  $t$  KPAs cooperate.

**Assumption 1.** The threshold scheme used in our system is secure in that it is computationally infeasible to obtain the shared secret when less than  $t$  KPAs collude.

For more detail information on the security of the threshold technique, refer to [4].

**Assumption 2.** Advantage of an adversary on DLP and CDHP in  $\mathbb{G}_1$  is negligible.

**Assumption 3.** Following public information is well-known to all entities in our key issuing model.

- The public system parameters:  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, l, P, P_{\text{KGC}}, P_{\text{KPAs}}, P_{\text{KPA}_1}, \dots, P_{\text{KPA}_n}, H_1, H_2 \rangle$ .
- $\langle \text{ID}, b_i P, b_i Q_{\text{ID}} \rangle$  and  $b_i d_{\text{ID}}^{(i)} = b_i x_i Q_{\text{ID}}$  pairs, which are published during LTK issuance phase.
- $\langle \text{ID}, T, X = b P_{\text{KPAs}}, Y = b d_{\text{ID}}, Z = b Q_{\langle \text{ID}, T \rangle} \rangle$ , and  $s Z = b d_{\langle \text{ID}, T \rangle}$ ,  $\langle \text{ID}, T, V = s Q_{\langle \text{ID}, T \rangle} + \bar{b} P_{\text{KPAs}}, \bar{V} = \bar{b} P \rangle$  pairs, which are known publicly during STK issuance phase.

There are five types of potential adversaries in our protocol. However, the ability of a malicious third party, the KCA, and less than  $t$  KPAs are equal based on Assumption 1. Furthermore, we can conclude that both the KGC, who can compute any STK it wishes, and users, who can acquire many LTKs and STKs of their own, are more powerful adversaries than other types of adversaries. Therefore, our protocol would be secure enough if it were secure against the KGC, users or even their collusion.

**Lemma 1.** Proposed key issuing model satisfies the correctness requirement.

*Proof.* In LTK extraction phase, a user sends  $\langle \text{ID}, b_i P, b_i Q_{\text{ID}} \rangle$  to  $t$  KPAs. Then each  $\text{KPA}_i$  1) authenticates the user, 2) computes LTK share  $b_i d_{\text{ID}}$ , and 3) sends  $b_i d_{\text{ID}}$  back to the user. The correctness of  $\langle \text{ID}, b_i P, b_i Q_{\text{ID}} \rangle$  can be verified using the Eq. 1. The correctness of  $b_i d_{\text{ID}}$  can be verified using the Eq. 2 as well.

In STK extraction phase, a user sends  $\langle \text{ID}, T, X = b P_{\text{KPAs}}, Y = b d_{\text{ID}}, Z = b Q_{\langle \text{ID}, T \rangle} \rangle$  to the KGC through the KCA. Then, the KGC sends  $b d_{\langle \text{ID}, T \rangle}$  and  $\langle \text{ID}, T, V = s Q_{\langle \text{ID}, T \rangle} + \bar{b} P_{\text{KPAs}}, \bar{V} = \bar{b} P \rangle$  back to the KCA. Finally, the KCA sends  $b d_{\langle \text{ID}, T \rangle}$  back to the user. Using the Eq. 3 and 4, the KGC can check the correctness of  $\langle \text{ID}, T, X =$

$b P_{\text{KPAs}}, Y = b d_{\text{ID}}, Z = b Q_{\langle \text{ID}, T \rangle} \rangle$ . The KCA is also able to verify the correctness of  $\langle \text{ID}, T, V = s Q_{\langle \text{ID}, T \rangle} + \bar{b} P_{\text{KPAs}}, \bar{V} = \bar{b} P \rangle$  using the Eq. 5. Finally, the user can verify the correctness of his/her STK  $d_{\langle \text{ID}, T \rangle}$  using the Eq. 6.

An adversary may disturb the LTK issuance by intercepting  $\langle \text{ID}, b_i P, b_i Q_{\text{ID}} \rangle$  and sending  $\langle \text{ID}, n_a b_i P, n_a b_i Q_{\text{ID}} \rangle$  instead using a random value  $n_a \in \mathbb{Z}_q^*$ . Although each KPA cannot detect this modification using the Eq. 1, the user can still check the correctness of the resulting LTK share by using the Eq. 2. Therefore, the user cannot be disturbed by this attack during the LTK issuance. Similarly, an adversary may hinder the STK issuance by intercepting  $\langle \text{ID}, T, X, Y, Z \rangle$  and sending  $\langle \text{ID}, T, n_a X, n_a Y, n_a Z \rangle$  instead using a random value  $n_a \in \mathbb{Z}_q^*$ . Even though the KGC cannot detect this change using the Eq. 3 and 4, the user can still verify the validity of resulting STK using the Eq. 6. Therefore, the user cannot be disturbed by this attack during the STK issuance.

By summing up the discussions mentioned above, we can conclude that each entity can check the correctness of messages which they have received from others, even though there are two exceptions which make no harm to our protocol. A user can also check the validity of LTK and STK which he/she received through the key issuing model. Therefore, we can conclude that the lemma holds.  $\square$

**Lemma 2.** Proposed key issuing model satisfies both the unforgeability and privacy requirements.

*Proof.* In our key issuing model, a user and the KGC may collude to compute any private key of another user. In this case, since the KGC can compute any STK it wants, forging the LTK is sufficient to compute a private key of a user. However, by the Assumption 1 ~ 3, it is computationally infeasible for a user to compute any LTK or STK corresponding to  $\langle \text{ID}, T \rangle$  pair. More specifically, to obtain a LTK, the KGC and a user can try to do the following using the publicly available information: 1) extract the LTK, 2) extract the master key of KPAs, and 3) collect more than or equal to  $t$  LTK shares.

However, since all the following are infeasible by the Assumption 2, it is infeasible for a user or the KGC to acquire any information that results in a legal LTK. Moreover, it is also infeasible even if a user and the KGC collude.

- Extracting  $x$  from  $P_{\text{KPAs}} = x P$  or any  $x_i$  from  $P_{\text{KPA}_i} = x_i P$ .
- Extracting  $d_{\text{ID}}^{(i)}$  from  $b_i d_{\text{ID}}^{(i)} = b_i x_i Q_{\text{ID}}$ , which is analogous to extracting  $b_i$  from  $b_i d_{\text{ID}}^{(i)}$
- Extracting  $d_{\text{ID}}$  from  $b d_{\text{ID}}$ .
- Extracting  $x$  from  $d_{\text{ID}} = x Q_{\text{ID}}$ .

A user may try to compute an illegal STK on his/her own. However, this is impossible because the followings are infeasible by Assumption 2.

- Extracting  $s$  from  $P_{KGC} = sP$ .
- Extracting  $s$  from previously issued STK  $d_{(ID,T)} = sQ_{(ID,T)}$ .

Additionally, a user may try to compute another user's STK. Even if the user is successful, it cannot cause any harm if the user do not have the corresponding LTK. Anyway, this is also infeasible because of the above and the followings are infeasible by Assumption 2.

- Computing  $d_{(ID,T)}$  from  $sZ = bd_{(ID,T)}$ .
- Computing  $\bar{b}xP$  from  $P_{KPAs} = xP$  and  $\bar{V} = \bar{b}P$ .

From these arguments, we can conclude that a LTK can only be computed by collaboration of more than or equal to  $t$  KPAs. Moreover, a STK can only be computed by the KGC or by cooperation of more than or equal to  $t$  KPAs. As a result, a private key of a user is only known to the legitimate owner assuming that more than or equal to  $t$  KPAs do not collude, and privacy property holds. As a result, the lemma holds.  $\square$

**Theorem 1.** *Our key issuing model satisfies all of our security requirements against every possible adversary models.*

*Proof.* By Lemma 1 and 2, the key issuing model satisfies every security requirements. Therefore, the theorem holds.  $\square$

### 4.3 Security against Private Key Leakage

In our scheme, a private key of a user is a bit concatenation of LTK and STK. As our scheme exploits simple blinding technique in order to provide secure channel between users and authorities, each secret value is only known to each responsible authority and appropriate user. In this reason, even though one of private keys of a user is revealed, the other keys of the user will be safe. Therefore an adversary, who knows a private key of a user  $D_{(ID,T)} = d_{ID} + d_{(ID,T)}$ , knows neither LTK  $d_{ID}$  nor STK  $d_{(ID,T)}$  and cannot compute other private keys of the user,  $D_{(ID,T')} = d_{ID} + d_{(ID,T')}$ . However, if the adversary collude with the KGC, they can compute other private keys of the user. This is because the KGC can compute any STK  $d_{(ID,T)}$  it wants. However, as the KGC is trusted authority, the probability that this kind of attack happens is negligible.

## 5 Conclusion

In this paper, we have proposed a new efficient model for issuing private keys in IBE scheme based on the Weil pairing. In our scheme, a private key is constructed using two values that are issued by two different parties, namely KPAs and the KGC. The KPAs issue LTK using the identity string of the user in a threshold manner, and the KGC issues STK using the same information used to construct the LTK plus some dynamic information.

When issuing the initial private key, our scheme provides similar level of efficiency to other schemes. However, re-issuing the private key by changing the dynamic information is much more efficient than others. Therefore, our scheme outperforms others when the user requires multiple private keys. In addition, we have eliminated the secure channel assumption by using a simple blinding technique. Moreover, we can adapt Gentry's time-slot based private key revocation approach more efficiently than others by using the time-slot information as the dynamic information included in a STK. We believe that this would be a better way to solve the private key revocation problem in IBCs than conventional ways. This is due to the fact that third-party queries used in traditional PKI would offset the advantage of IBCs.

## References

- [1] S. Al-Riyami and K. Paterson. Certificateless public key cryptography. In *Advances in Cryptology, Asiacrypt 2003*, volume 2894 of *LNCS*, pages 452–473. Springer-Verlag, 2003.
- [2] M. Bellare and S. Holdwasser. Verifiable partial key escrow. In *Proc. of the 4th ACM Conf. on Computer and Communications Security*, pages 78–91, 1997.
- [3] D. Boneh and M. Franklin. Identity-based encryption from weil pairing. In *Advances in Cryptology, Crypto 2001*, volume 2139 of *LNCS*, pages 213–229. Springer-Verlag, 2001.
- [4] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. A new RFID authentication protocol using keyed hash function. In *Advances in Cryptology, Eurocrypt 1999*, volume 1592 of *LNCS*, pages 295–310, 1999.
- [5] C. Gentry. Certificate-based encryption and the certificate revocation problem. In *Advances in Cryptology, Eurocrypt 2003*, volume 2656 of *LNCS*, pages 490–497, 2003.
- [6] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo. Secure key issuing in id-based cryptography. In *Proc. of the 2nd Australasian Information Security Workshop, AISW 2004*, volume 32 of *CRPIT*, pages 69–74, 2004.
- [7] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Crypto 1984*, volume 196 of *LNCS*, pages 47–53. Springer-Verlag, 1985.