Identity-Based Encryption with Outsourced Revocation in Cloud Computing

Jin Li, Jingwei Li, Xiaofeng Chen, Chunfu Jia, and Wenjing Lou, *Senior Member*, *IEEE*

Abstract—Identity-Based Encryption (IBE) which simplifies the public key and certificate management at Public Key Infrastructure (PKI) is an important alternative to public key encryption. However, one of the main efficiency drawbacks of IBE is the overhead computation at Private Key Generator (PKG) during user revocation. Efficient revocation has been well studied in traditional PKI setting, but the cumbersome management of certificates is precisely the burden that IBE strives to alleviate. In this paper, aiming at tackling the critical issue of identity revocation, we introduce outsourcing computation into IBE for the first time and propose a revocable IBE scheme in the server-aided setting. Our scheme offloads most of the key generation related operations during key-issuing and key-update processes to a Key Update Cloud Service Provider, leaving only a constant number of simple operations for PKG and users to perform locally. This goal is achieved by utilizing a novel collusion-resistant technique: we employ a hybrid private key for each user, in which an AND gate is involved to connect and bound the identity component and the time component. Furthermore, we propose another construction which is provable secure under the recently formulized Refereed Delegation of Computation model. Finally, we provide extensive experimental results to demonstrate the efficiency of our proposed construction.

Index Terms—Identity-based encryption (IBE), revocation, outsourcing, cloud computing

1 Introduction

Identity-Based Encryption (IBE) is an interesting alternative to public key encryption, which is proposed to simplify key management in a certificate-based Public Key Infrastructure (PKI) by using human-intelligible identities (e.g., unique name, email address, IP address, etc) as public keys. Therefore, sender using IBE does not need to look up public key and certificate, but directly encrypts message with receiver's identity. Accordingly, receiver obtaining the private key associated with the corresponding identity from Private Key Generator (PKG) is able to decrypt such ciphertext.

Though IBE allows an arbitrary string as the public key which is considered as an appealing advantages over PKI, it demands an efficient revocation mechanism. Specifically, if the private keys of some users get compromised, we must provide a mean to revoke such users from system. In PKI setting, revocation mechanism is realized by appending validity periods to certificates or using involved combinations of techniques [1]–[3]. Nevertheless, the cumbersome management of certificates is precisely the burden that IBE strives to alleviate.

As far as we know, though revocation has been thoroughly studied in PKI, few revocation mechanisms are known in IBE setting. In [4], Boneh and Franklin suggested that users renew their private keys periodically and senders use the receivers' identities concatenated with current time period. But this mechanism would result in an overhead load at PKG. In another word, all the users regardless of whether their keys have been revoked or not, have to contact with PKG periodically to prove their identities and update new private keys. It requires that PKG is online and the secure channel must be maintained for all transactions, which will become a bottleneck for IBE system as the number of users grows.

In 2008, Boldyreva, Goyal and Kumar [5] presented a revocable IBE scheme. Their scheme is built on the idea of fuzzy IBE primitive [6] but utilizing a binary tree data structure to record users' identities at leaf nodes. Therefore, key-update efficiency at PKG is able to be significantly reduced from linear to the height of such binary tree (i.e. logarithmic in the number of users). Nevertheless, we point out that though the binary tree introduction is able to achieve a relative high performance, it will result in other problems: 1) PKG has to generate a key pair for all the nodes on the path from the identity leaf node to the root node, which results in complexity logarithmic in the number of users in system for issuing a single private key. 2) The size of private key grows in logarithmic in the number of users in system, which makes it difficult in private key storage for users. 3) As the number of users in system grows, PKG has to maintain a binary tree with a large amount of nodes, which introduces another bottleneck for the global system.

In tandem with the development of cloud computing, there has emerged the ability for users to buy on-demand computing from cloud-based services such as Amazon's EC2 and Microsoft's Windows Azure. Thus it desires a new working

- *J. Li is with the School of Computer Science, Guangzhou University, Guangzhou, 510500 China and also with the State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an, 700071 China. E-mail: lijin@gzhu.edu.cn.*
- *J. Li and C. Jia are with the College of Information Technical Science, Nankai University, Nankai, Tianjin 300071, China. E-mail: lijw@mail.nankai.edu.cn, cfjia@nankai.edu.cn.*
- *X. Chen is with the State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an 700071, China. E-mail: xfchen@xidian.edu.cn.*
- *W. Lou is with Virginia Polytechnic Institute and State University, Blacksburg 22043, VA. E-mail: wjlou@vt.edu.*

Manuscript received 25 Dec. 2012; revised 25 July 2013; accepted 10 Oct. 2013. Date of publication 20 Oct. 2013; date of current version 16 Jan. 2015.
Recommended for acceptance by M. Parashar.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2013.208

paradigm for introducing such cloud services into IBE revocation to fix the issue of efficiency and storage overhead described above. A naive approach would be to simply hand over the PKG's master key to the Cloud Service Providers (CSPs). The CSPs could then simply update all the private keys by using the traditional key update technique [4] and transmit the private keys back to unrevoked users. However, the naive approach is based on an unrealistic assumption that the CSPs are fully trusted and is allowed to access the master key for IBE system. On the contrary, in practice the public clouds are likely outside of the same trusted domain of users and are curious for users' individual privacy. For this reason, a challenge on how to design a secure revocable IBE scheme to reduce the overhead computation at PKG with an untrusted CSP is raised.

In this paper, we introduce outsourcing computation into IBE revocation, and formalize the security definition of outsourced revocable IBE for the first time to the best of our knowledge. We propose a scheme to offload all the key generation related operations during key-issuing and key-update, leaving only a constant number of simple operations for PKG and eligible users to perform locally. In our scheme, as with the suggestion in [4], we realize revocation through updating the private keys of the unrevoked users. But unlike that work [4] which trivially concatenates time period with identity for key generation/update and requires to re-issue the whole private key for unrevoked users, we propose a novel collusion-resistant key issuing technique: we employ a hybrid private key for each user, in which an AND gate is involved to connect and bound two sub-components, namely the identity component and the time component. At first, user is able to obtain the identity component and a default time component (i.e., for current time period) from PKG as his/her private key in key-issuing. Afterwards, in order to maintain decryptability, unrevoked users needs to periodically request on key-update for time component to a newly introduced entity named Key Update Cloud Service Provider (KU-CSP).

Compared with the previous work [4], our scheme does not have to re-issue the whole private keys, but just need to update a lightweight component of it at a specialized entity KU-CSP. We also specify that 1) with the aid of KU-CSP, user needs not to contact with PKG in key-update, in other words, PKG is allowed to be offline after sending the revocation list to KU-CSP. 2) No secure channel or user authentication is required during key-update between user and KU-CSP.

Furthermore, we consider to realize revocable IBE with a semi-honest KU-CSP. To achieve this goal, we present a security enhanced construction under the recently formalized Refereed Delegation of Computation (RDoC) model [7]. Finally, we provide extensive experimental results to demonstrate the efficiency of our proposed construction.

This paper is organized as follows. In Section 2, we describe the preliminaries of our scheme. In Section 3, we present the system model and security definition of our scheme. The proposed construction, and its security analysis are presented in Section 4. In Section 5, we propose a security enhanced construction under RDoC model. An extensive experimental result is presented in Section 6 to demonstrate the efficiency of our proposed constructions. Finally, After revisiting the related work in Section 7, we draw conclusion in Section 8.

2 Preliminary

In this section, we give a brief review on some cryptographic background and identity based encryption.

2.1 Cryptographic Background

Definition 1. *(Bilinear map) Let $\mathbb{G}, \mathbb{G}_T$ be cyclic groups of prime order q, writing the group action multiplicatively. g is a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a map with the following properties:*

- *Bilinearity: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}$, and $a, b \in_R \mathbb{Z}_q$;*
- *Non-degeneracy: There exists $g_1, g_2 \in \mathbb{G}$ with $e(g_1, g_2) \neq 1$, in other words, the map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$;*
- *Computability: There is an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}$.*

Definition 2. *(DBDH problem) The decision Bilinear Diffie-Hellman (DBDH) problem is that, given g, g^x, g^y, $g^z \in \mathbb{G}$ for unknown random value $x, y, z \in_R \mathbb{Z}_q$, and $T \in_R \mathbb{G}_T$, to decide if $T = e(g, g)^{xyz}$.*

We say that the (t, ϵ)-DBDH assumption holds in $\mathbb{G}$ if no t-time algorithm has probability at least $\frac{1}{2} + \epsilon$ in solving the DBDH problem for non-negligible ϵ.

2.2 Identity-Based Encryption

An IBE scheme which typically involves two entities, PKG and users (including sender and receiver) is consisted of the following four algorithms.

- Setup(λ): The setup algorithm takes as input a security parameter λ and outputs the public key PK and the master key MK. Note that the master key is kept secret at PKG.
- KeyGen(MK, ID): The private key generation algorithm is run by PKG, which takes as input the master key MK and user's identity $\mathrm{ID} \in \{0, 1\}^*$. It returns a private key SK_{ID} corresponding to the identity ID.
- Encrypt(M, ID'): The encryption algorithm is run by sender, which takes as input the receiver's identity ID' and a message M to be encrypted. It outputs the ciphertext CT.
- Decrypt$(CT, SK_{\mathrm{ID}'})$: The decryption algorithm is run by receiver, which takes as input the ciphertext CT and his/her private key $SK_{\mathrm{ID}'}$. It returns a message M or an error $\perp$.

An IBE scheme must satisfy the definition of consistency. Specifically, when the private key SK_{ID} generated by algorithm KeyGen when it is given ID as the input, then Decrypt$(CT, SK_{\mathrm{ID}}) = M$ where $CT =$ Encrypt(M, ID).

The motivation of IBE is to simplify certificate management. For example, when Alice sends an email to Bob at bob@company.com, she simply encrypts her message using Bob's email address "bob@company.com", but does not need to obtain Bob's public key certificate. When Bob receives the encrypted email he authenticate himself at PKG to obtain his private key, and read his email with such a private key.

3 Problem Statement

3.1 System Model

We present system model for outsourced revocable IBE in Fig. 1. Compared with that for typical IBE scheme, a KU-CSP is involved to realize revocation for compromised users. Actually, the KU-CSP can be envisioned as a public cloud run by a third party to deliver basic computing capabilities to PKG as standardized services over the network. Typically, KU-CSP is hosted away from either users or PKG, but provides a way to reduce PKG computation and storage cost by providing a flexible, even temporary extension to infrastructure. When revocation is triggered, instead of re-requesting private keys from PKG in [4], unrevoked users have to ask the KU-CSP for updating a lightweight component of their private keys. Though many details are involved in KU-CSP's deployment, in this paper we just logically envision it as a computing service provider, and concern how to design secure scheme with an untrust KU-CSP.

Based on the system model proposed, we are able to define the outsourced revocable IBE scheme. Compared with the traditional IBE definition, the KeyGen, Encrypt and Decrypt algorithms are redefined as follows to integrate time component. Note that two lists RL and TL are utilized in our definition, where RL records the identities of revoked users and TL is a linked list for past and current time period.

- KeyGen$(MK, \mathrm{ID}, RL, TL)$: The key generation algorithm run by PKG takes as input–a master key MK, an identity ID, a revocation list RL and a time list TL. If $\mathrm{ID} \in RL$, the algorithm is aborted. Otherwise, it sends the private key $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ to user where $IK[\mathrm{ID}]$ is the identity component for private key SK_{ID} and $TK[\mathrm{ID}]_{T_i}$ is its time component for current time period T_i. Additionally, the algorithm sends an outsourcing key OK_{ID} to KU-CSP.
- Encrypt$(M, \mathrm{ID}, T_i, PK)$: The encryption algorithm run by sender takes as input–a message M, an identity ID and a time period T_i. It outputs the ciphertext CT.
- Decrypt$(CT, SK_{\mathrm{ID}'})$: The decryption algorithm run by receiver takes as input–a ciphertext CT encrypted under identity ID and time period T_i and a private key $SK_{\mathrm{ID}'} = (IK[\mathrm{ID}'], TK[\mathrm{ID}']_{T_j})$. It outputs the original message M if $\mathrm{ID} = \mathrm{ID}'$ and $T_i = T_j$, otherwise outputs $\perp$.

In addition, two algorithms are defined to realize revocation at KU-CSP through updating the private keys of unrevoked users.

- Revoke$(RL, TL, \{\mathrm{ID}_{i_1}, \ldots, \mathrm{ID}_{i_k}\})$: The revocation algorithm run by PKG takes as input–a revocation list RL, a time list TL and the set of identities to be revoked $\{\mathrm{ID}_{i_1}, \mathrm{ID}_{i_2}, \ldots, \mathrm{ID}_{i_k}\}$. It outputs an updated time period T_{i+1} as well as the updated revocation list RL' and time list TL'.
- KeyUpdate$(RL, \mathrm{ID}, T_{i+1}, OK_{\mathrm{ID}})$: The key update algorithm run by KU-CSP takes as input–a revocation list RL, an identity ID, a time period T_{i+1} and the outsourcing key OK_{ID} for identity ID. It outputs user's updated time component in private key $TK[\mathrm{ID}]_{T_{i+1}}$ if his identity ID does not belong to RL, otherwise, outputs $\perp$.

In this paper, we discuss user revocation, that is how to deprive users of decryptability even if they have been issued their private keys. To this end, we embed a time period into private key in a clever manner for revocation. Specifically, in the same example illustrated in Section 2.2, Alice in our setting not only encrypts message with Bob's email address "bob@company.com" but also with current time period (e.g., "Thu Jul 18 2013"). When receives the encrypted email, Bob then obtains his private key consisting of an identity component and a time period component from PKG. With the both appropriate components, the email can be read.

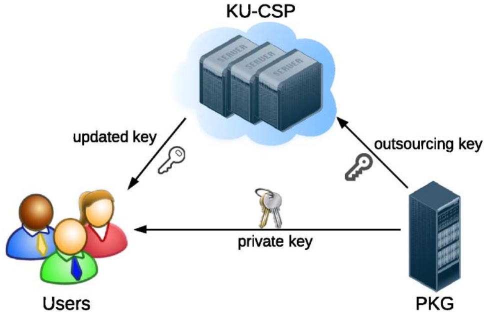


Fig. 1. System model for IBE with outsourced revocation.

Suppose Bob is compromised. Then, the time components of all the other users are updated by KU-CSP with a new time period (e.g., "Fri Jul 19 2013"). From then on, the message sent to Bob should be encrypted with Bob's email address and the updated time period. Since Bob does not have the time component corresponding to the updated time period, the following encrypted messages can not be decrypted by Bob even if they are intended for him.

The challenge in designing the outsourced revocable IBE scheme is how to prevent a collusion between Bob and other unrevoked dishonest users. Specifically, a dishonest user (named Eve) can share her updated time component (i.e., "Fri Jul 19 2013") with Bob, and help Bob decrypt ciphertext even if Bob just has the previous one (i.e., "Thu Jul 18 2013"). We will show how to avoid such a collusion later.

3.2 Security Definition

We assume that KU-CSP in the proposed system model is semi-trusted. Specifically, it will follow our protocol but try to find out as much secret information as possible based on its possession. Therefore, two types of adversaries are to be considered as follows.

- *Type-I adversary*. It is defined as a curious user with identity ID but revoked before time period T_i. Such adversary tries to obtain useful information from ciphertext intended for him/her at or after T_i (e.g. time period $T_i, T_{i+1}, \ldots$) through colluding with other users even if they are unrevoked. Therefore, it is allowed to ask for private key including identity component and updated time component for cooperative users. We specify that under the assumption that KU-CSP is semi-trusted, type-I adversary cannot get outsourcing key for any users.
- *Type-II adversary*. It is defined as a curious KU-CSP which aims to obtain useful information from ciphertext intended for some target identity ID at time period T_i. Such adversary not only possess of outsourcing keys for all users in the system, but also is able to get user's private key through colluding with any other user with identity ID$'$. It is noted that to make such attack reasonable, we must restrict $\mathrm{ID}' \neq \mathrm{ID}$.

CCA Security Game for Type-I Adversary

Setup: Challenger runs $\mathsf{Setup}(\lambda)$ to obtain the key pair (PK, MK) and output PK.

Phase 1: Challenger initializes an empty table list L and an empty set S. Adversary is provided the following oracles.

- *Private key extraction oracle*. Upon receiving ID, run KeyGen to obtain $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ and OK_{ID}. After adding the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ into L, output $IK[\mathrm{ID}]$.
- *Updated key extraction oracle*. Upon receiving ID and T_j, if there exists an entry of $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, set $S = S \cup \{(\mathrm{ID}, T_j)\}$. Accordingly, run $\mathsf{KeyUpdate}$ and output the updated key $TK[\mathrm{ID}]_{T_j}$.
- *Decryption oracle*. Upon receiving ID, T_j and CT, run $\mathsf{Decrypt}$ and output the resulting plaintext M.

Challenge: Adversary outputs two equal-length plaintexts M_0, M_1, T^* and ID^* with the restriction that $(\mathrm{ID}^*, T^*) \notin S$. Challenger picks a random bit $b_{\mathrm{I}} \in \{0,1\}$ and sets $CT^* = \mathsf{Encrypt}(M_{b_{\mathrm{I}}}, \mathrm{ID}^*, T^*, PK)$.

Phase 2: Adversary adaptively issues more queries as in phase 1 with the restriction that (ID^*, T^*) cannot be queried in updated key extraction oracle.

Guess: Finally, adversary outputs a guess $b'_{\mathrm{I}} \in \{0,1\}$ and wins the game if $b'_{\mathrm{I}} = b_{\mathrm{I}}$.

CCA Security Game for Type-II Adversary

Setup: It is identical to the setup phase in the CCA security game for type-I adversary.

Phase 1: Challenger initializes an empty table list L, and two empty sets U and I. Then, adversary is provided the following oracles.

- *Outsourcing key extraction oracle*. Upon receiving ID, challenger runs KeyGen to obtain $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ and OK_{ID}. After adding the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ into L, output OK_{ID}.
- *Private key extraction oracle*. Upon receiving ID, if there exists an entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, set $U = U \cup \{\mathrm{ID}\}$ and return $IK[\mathrm{ID}]$ back to adversary.
- *Updated key extraction oracle*. Upon receiving ID and T_j, if there exists an entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, check on whether $\mathrm{ID} \in U$, if not set $I = I \cup \{T_j\}$. Then, run $\mathsf{KeyUpdate}$ and output $TK[\mathrm{ID}]_{T_j}$.
- *Decryption oracle*. It is identical to the decryption oracle in the CCA security game for type-I adversary.

Challenge: Adversary outputs two equal-length plaintexts M_0, M_1, T^* and ID^* with the restriction that $T^* \notin I$ and $\mathrm{ID}^* \notin U$. Challenger picks a random bit $b_{\mathrm{II}} \in \{0,1\}$ and sets $CT^* = \mathsf{Encrypt}(M_{b_{\mathrm{II}}}, \mathrm{ID}^*, T^*, PK)$.

Phase 2: Adversary adaptively issues more queries as in phase 1 with the restrictions that i) ID^* cannot be queried in private key extraction oracle; ii) (ID^*, T^*) cannot be queried in updated key extraction oracle.

Guess: Finally, adversary outputs a guess $b'_{\mathrm{II}} \in \{0,1\}$ and wins the game if $b'_{\mathrm{II}} = b_{\mathrm{II}}$.

Fig. 2. CCA security game for type-I and type-II adversary.

Having the intuitions above, we are able to define CCA security game for type-I and type-II adversary respectively for our setting in Fig. 2. Suppose $\mathcal{A}_i$ is the type-i adversary for $i = \mathrm{I}, \mathrm{II}$. Then, its advantage in attacking the IBE with outsourced revocation scheme $\mathcal{E}$ is defined as $Adv_{\mathcal{E},\mathcal{A}_i}(\lambda) = |\Pr[b_i = b'_i] - \frac{1}{2}|$.

Definition 3. *An identity-based encryption with outsourced revocation scheme is semantically secure against adaptive chosen-ciphertext attack (IND-ID-CCA) if no polynomially bounded adversary has a non-negligible advantage against challenger in security game for both type-I and type-II adversary.*

Finally, beyond the CCA security, we also specify that 1) An IBE with outsourced revocation scheme is IND-ID-CPA secure (or semantically secure against chosen-plaintext attack) if no polynomial time adversary has non-negligible advantage in modified games for both type-I and type-II adversary, in which the decryption oracle in both phase 1 and phase 2 is removed; 2) An IBE with outsourced revocation scheme is secure in selective model if no polynomial time adversary has non-negligible advantage in modified games for both type-I and type-II adversary, in which the challenge identity and time period is submitted before setup.

4 Efficient IBE with Outsourced Revocation

4.1 Intuition

In order to achieve efficient revocation, we introduce the idea of "partial private key update" into the proposed construction, which operates on two sides: 1) We utilize a "hybrid private key" for each user in our system, which employs an AND gate connecting two sub-components namely the identity component IK and the time component TK respectively. IK is generated by PKG in key-issuing but TK is updated by the newly introduced KU-CSP in key-update; 2) In encryption, we take as input user's identity ID as well as the time period T to restrict decryption, more precisely, a user is allowed to perform successful decryption if and only if the identity and time period embedded in his/her private key are identical to that associated with the ciphertext. Using such skill, we are able to revoke user's decryptability through updating the time component for private key by KU-CSP.

Moreover, we remark that it cannot trivially utilize an identical updated time component for all users because revoked user is able to re-construct his/her ability through colluding with unrevoked users. To eliminate such collusion, we randomly generate an outsourcing key for each identity ID, which essentially decides a "matching relationship" for the two sub-components. Furthermore, we let KU-CSP maintain a list UL to record user's identity and its corresponding outsourcing key. In key-update, we can use OK_{ID} to update the time component $TK[\mathrm{ID}]_T$ for identity ID. Suppose a user with identity ID is revoked at T_i. Even if he/she is able to obtain $TK[\mathrm{ID}']_{T_{i+1}}$ for identity ID$'$, the revoked user still cannot decrypt ciphertext encrypted under T_{i+1}.

4.2 Proposed Construction

We present our construction based on [6] as follows.

- Setup(λ): The setup algorithm is run by PKG. It selects a random generator $g \in_R \mathbb{G}$ as well as a random integer $x \in_R \mathbb{Z}_q$, and sets $g_1 = g^x$. Then, PKG picks a random element $g_2 \in_R \mathbb{G}$ and two hash functions H_1, H_2: $\{0,1\}^* \to \mathbb{G}_T$. Finally, output the public key $PK = (g, g_1, g_2, H_1, H_2)$ and the master key $MK = x$.
- KeyGen$(MK, \mathrm{ID}, RL, TL, PK)$: For each user's private key request on identity ID, PKG firstly checks whether the request identity ID exists in RL, if so the key generation algorithm is aborted. Next, PKG randomly selects $x_1 \in_R \mathbb{Z}_q$ and sets $x_2 = x - x_1 \bmod q$. It randomly chooses $r_{\mathrm{ID}} \in_R \mathbb{Z}_q$, and computes $IK[\mathrm{ID}] = (g_2^{x_1} \cdot (H_1(\mathrm{ID}))^{r_{\mathrm{ID}}}, g^{r_{\mathrm{ID}}})$. Then, PKG reads the current time period T_i from TL (we require that PKG should create current time period firstly if TL is empty). Accordingly, it randomly selects $r_{T_i} \in_R \mathbb{Z}_q$ and computes $TK[\mathrm{ID}]_{T_i} = (d_{T_i 0}, d_{T_i 1})$, where $d_{T_i 0} = g_2^{x_2} \cdot (H_2(T_i))^{r_{T_i}}$ and $d_{T_i 1} = g^{r_{T_i}}$. Finally, output $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ and $OK_{\mathrm{ID}} = x_2$.
- Encrypt$(M, \mathrm{ID}, T_i, PK)$: Suppose a user wishes to encrypt a message M under identity ID and time period T_i. He/She selects a random value $s \in_R \mathbb{Z}_q$ and computes $C_0 = Me(g_1, g_2)^s$, $C_1 = g^s$, $E_{\mathrm{ID}} = (H_1(\mathrm{ID}))^s$ and $E_{T_i} = (H_2(T_i))^s$. Finally, publish the ciphertext as $CT = (C_0, C_1, E_{\mathrm{ID}}, E_{T_i})$.
- Decrypt$(CT, SK_{\mathrm{ID}}, PK)$: Suppose that the ciphertext CT is encrypted under ID and T_i, and the user has a private key $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$, where $IK[\mathrm{ID}] = (d_0, d_1)$ and $TK[\mathrm{ID}]_{T_i} = (d_{T_i 0}, d_{T_i 1})$. He/She computes

$$\begin{aligned} M &= \frac{C_0 e(d_1, E_{\mathrm{ID}}) e(d_{T_i 1}, E_{T_i})}{e(C_1, d_0) e(C_1, d_{T_i 0})} \\ &= \frac{Me(g_1, g_2)^s}{e(g, g_2)^{x_2 s} e(g, g_2)^{x_1 s}} \\ &= M. \end{aligned}$$

- Revoke$(RL, TL, \{\mathrm{ID}_{i_1}, \mathrm{ID}_{i_2}, \ldots, \mathrm{ID}_{i_k}\})$: If users with identities in the set $\{\mathrm{ID}_{i_1}, \mathrm{ID}_{i_2}, \ldots, \mathrm{ID}_{i_k}\}$ are to be revoked at time period T_i, PKG updates the revocation list as $RL' = RL \cup \{\mathrm{ID}_{i_1}, \mathrm{ID}_{i_2}, \ldots, \mathrm{ID}_{i_k}\}$ as well as the time list through linking the newly created time period T_{i+1} onto original list TL. Finally send a copy for the updated revocation list RL' as well as the new time period T_{i+1} to KU-CSP.
- KeyUpdate$(RL, \mathrm{ID}, T_{i+1}, OK_{\mathrm{ID}})$: Upon receiving a key-update request on ID, KU-CSP firstly checks whether ID exists in the revocation list RL, if so KU-CSP returns $\perp$ and key-update is aborted. Otherwise, KU-CSP fetches the corresponding entry $(\mathrm{ID}, OK_{\mathrm{ID}} = x_2)$ in the user list UL. Then, it randomly selects $r_{T_{i+1}} \in_R \mathbb{Z}_q$, and computes $d_{T_{i+1} 0} = g_2^{x_2} \cdot (H_2(T_{i+1}))^{r_{T_{i+1}}}$ and $d_{T_{i+1} 1} = g^{r_{T_{i+1}}}$. Finally, output $TK[\mathrm{ID}]_{T_{i+1}} = (d_{T_{i+1} 0}, d_{T_{i+1} 1})$.[1]

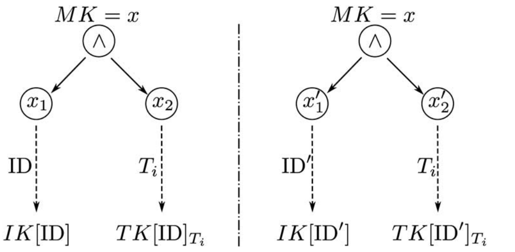


Fig. 3. A comparison on generating private key for two different users.

Finally, we emphasize that the idea behind our construction is to realize revocation through updating the time component in private key. Therefore, the key point is to prevent revoked user from colluding with other users to re-construct his/her private key. As declaring in intuition, such collusion attack is resistant in our proposed construction due to the random split on x for each user. Specifically, as shown in Fig. 3 in which $\wedge$ is an AND gate connecting two sub-components, if two different users call for their private keys, PKG will obtain two randomly splits (x_1, x_2) and (x'_1, x'_2) with the complementary that $x_1 + x_2 = x \bmod q$ and $x'_1 + x'_2 = x \bmod q$. x_1 and x'_1 are used to produce the identity component for ID and ID$'$ respectively, while the time component is separately generated from x_2 and x'_2. By the reason that the complementary exists between x_1 and x_2 as well as x'_1 and x'_2, the identity component and time component should accordingly have a "verification" in private key. With such "verification", even if a curious user obtains time component of other users, he/she cannot forge a valid private key for himself to perform decryption successfully.

4.3 Key Service Procedures

Based on our algorithm construction, as shown in Fig. 4, the key service procedures including key-issuing, key-update and revocation in proposed IBE scheme with outsourced revocation work as follows.

- *Key-issuing.* We require that PKG maintains a revocation list RL and a time list TL locally. Upon receiving a private key request on ID, PKG runs KeyGen$(MK, \mathrm{ID}, RL, TL, PK)$ to obtain private key SK_{ID} and outsourcing key OK_{ID}. Finally, it sends SK_{ID} to user and $(\mathrm{ID}, OK_{\mathrm{ID}})$ to KU-CSP respectively. As described in intuition, for each entry $(\mathrm{ID}, OK_{\mathrm{ID}})$ sent from PKG, KU-CSP should add it into a locally maintained user list UL.
- *Key-update.* If some users have been revoked at time period T_i, each unrevoked user needs to send key-update request to KU-CSP to maintain decryptability. Upon receiving the request on identity ID, KU-CSP runs KeyUpdate$(RL, \mathrm{ID}, T_{i+1}, OK_{\mathrm{ID}})$ to obtain $TK[\mathrm{ID}]_{T_{i+1}}$. Finally, it sends such time component back to user who is able to update his/her private key as $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_{i+1}})$.
- *Revocation.* Similar to key-update, if a revoked user sends a key-update request on identity ID, KU-CSP runs

[1] No secure communication channel is required between user and KU-CSP. Furthermore, it is no need for the identity authentication which relieves the computational overhead at user side.

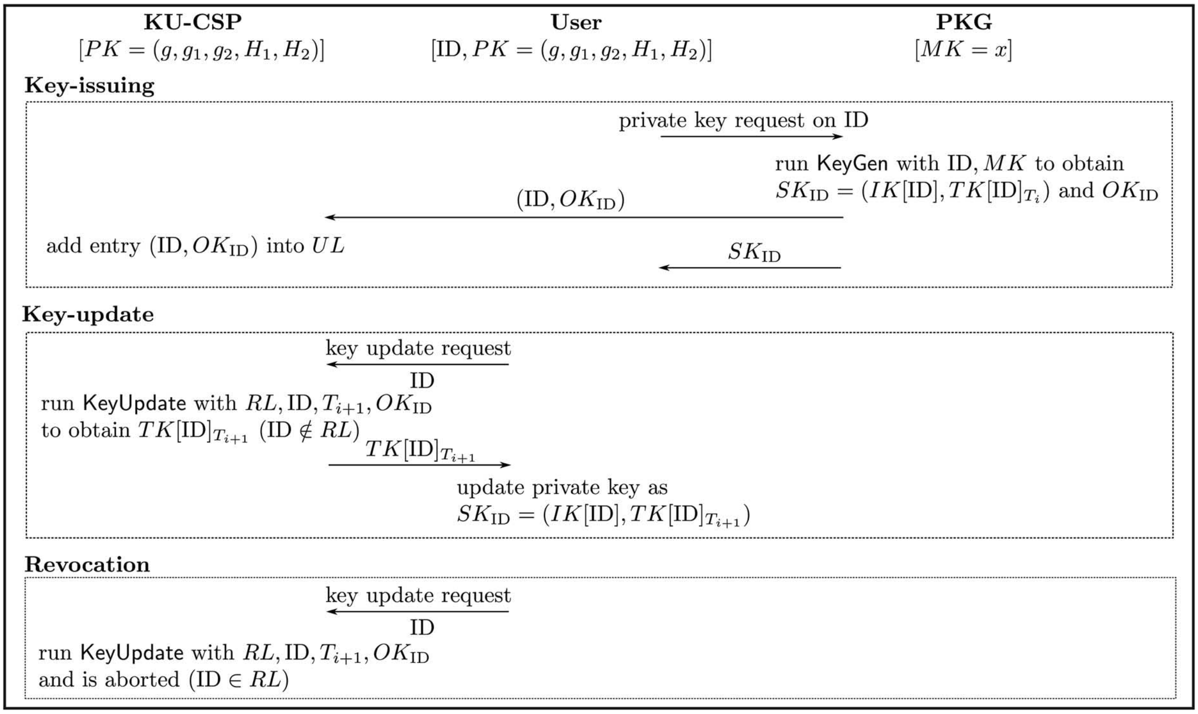

KU-CSP
[PK = (g, g1, g2, H1, H2)]
User
[ID, PK = (g, g1, g2, H1, H2)]
PKG
[MK = x]
Key-issuing
private key request on ID
run KeyGen with ID, MK to obtain
SK_ID = (IK[ID], TK[ID]_Ti) and OK_ID
(ID, OK_ID)
add entry (ID, OK_ID) into UL
SK_ID
Key-update
key update request
ID
run KeyUpdate with RL, ID, T_i+1, OK_ID
to obtain TK[ID]_Ti+1 (ID ∉ RL)
TK[ID]_Ti+1
update private key as
SK_ID = (IK[ID], TK[ID]_Ti+1)
Revocation
key update request
ID
run KeyUpdate with RL, ID, T_i+1, OK_ID
and is aborted (ID ∈ RL)

Fig. 4. Protocol for key-issuing, key update and revocation.

$\mathsf{KeyUpdate}(RL, \mathrm{ID}, T_{i+1}, OK_{\mathrm{ID}})$ as well. Nevertheless, since $\mathrm{ID} \in RL$, KU-CSP will return $\perp$. Therefore, such key-update request is aborted.

4.4 Security Analysis

Theorem 1. *Suppose that the (t,ϵ)–DBDH assumption holds in $\mathbb{G}$ and hash functions H_1 and H_2 are random oracles. Suppose the adversary makes at most q_{H_1}, q_{H_2}, q_P, q_U and q_O queries to hash functions H_1, H_2, private key, updated key and outsourcing key extraction oracles respectively. We use t_{EXP} to denote time cost for single multi-based exponentiation operation in $\mathbb{G}$. Then the proposed IBE with outsourced revocation scheme is (t', ϵ') secure in the sense of IND-ID-CPA where $t' \approx t + (q_{H_1} + q_{H_2} + 3q_P + 3q_U)t_{\mathrm{EXP}}$ and $\epsilon' = \frac{1}{q_{H_1} q_{H_2}}\epsilon$*

Proof. Assume that an adversary $\mathcal{A}_{\mathrm{I}}$ and $\mathcal{A}_{\mathrm{II}}$ have advantage ϵ_{I} and ϵ_{II} in attacking the proposed IBE scheme in the sense of IND-ID-CPA security for type-I and type-II adversary respectively. We will build two simulators $\mathcal{S}_{\mathrm{I}}$ and $\mathcal{S}_{\mathrm{II}}$ that respectively uses $\mathcal{A}_{\mathrm{I}}$ and $\mathcal{A}_{\mathrm{II}}$ as a sub-algorithm to solve the decisional BDH problem with a non-negligible probability.

Suppose challenger in DBDH problem flips a fair binary coin μ outside of $\mathcal{S}_{\mathrm{I}}$ and $\mathcal{S}_{\mathrm{II}}$'s view. If $\mu = 0$, then $\mathcal{S}_{\mathrm{I}}$ and $\mathcal{S}_{\mathrm{II}}$ are given $(X = g^x, Y = g^y, Z = g^z, P = e(g,g)^{xyz})$; otherwise, $(X = g^x, Y = g^y, Z = g^z, P = e(g,g)^v)$ for random $x, y, z, v \xleftarrow{R} \mathbb{Z}_q$. $\mathcal{S}_{\mathrm{I}}$ and $\mathcal{S}_{\mathrm{II}}$ are asked to output a value μ' as the guess for μ. Then we provide simulations as follows.

Simulation of $\mathcal{S}_{\mathrm{I}}$ against Type-I Adversary

Setup: $\mathcal{S}_{\mathrm{I}}$ sets $g_1 = X, g_2 = Y$ and sends the public key $PK = (g, g_1, g_2)$ to $\mathcal{A}_{\mathrm{I}}$.

Phase 1: $\mathcal{S}_{\mathrm{I}}$ initializes an empty table list L, and an empty set S. $\mathcal{A}_{\mathrm{I}}$ is allowed to issue queries in the following types.

- *H_1-query.* $\mathcal{S}_{\mathrm{I}}$ randomly picks $\kappa \in_R \{1, 2, \ldots, q_{H_1}\}$ and maintains a list $\mathcal{L}_1$ to store the answers to the hash oracle H_1. Upon receiving ID_i for $1 \le i \le q_{H_1}$, $\mathcal{S}_{\mathrm{I}}$ performs a check on $\mathcal{L}_1$. If an entry for the query is found, the same answer will be returned. Otherwise, $\mathcal{S}_{\mathrm{I}}$ randomly selects $u_i \in_R \mathbb{Z}_q$ and computes
$$H_1(\mathrm{ID}_i) = \begin{cases} g_1^{u_i}, & i \neq \kappa, \\ g^{u_\kappa}, & i = \kappa. \end{cases}$$
After storing the entry (ID_i, u_i) in $\mathcal{L}_1$, $\mathcal{S}_{\mathrm{I}}$ returns $H_1(\mathrm{ID}_i)$.
- *H_2-query.* $\mathcal{S}_{\mathrm{I}}$ randomly picks $\eta \in_R \{1, 2, \ldots, q_{H_2}\}$ and maintains a list $\mathcal{L}_2$ to store the answers to the hash oracle H_2. Upon receiving T_j for $1 \le j \le q_{H_2}$, $\mathcal{S}_{\mathrm{I}}$ performs a check on $\mathcal{L}_2$. If an entry for the query is found, the same answer will be returned. Otherwise, $\mathcal{S}_{\mathrm{I}}$ randomly selects $v_j \in_R \mathbb{Z}_q$ and computes
$$H_2(T_j) = \begin{cases} g_1^{v_j}, & j \neq \eta, \\ g^{v_\eta}, & i = \eta. \end{cases}$$
After storing the entry (T_j, v_j) in $\mathcal{L}_2$, $\mathcal{S}_{\mathrm{I}}$ returns $H_2(T_j)$.
- *Private key query.* Upon receiving ID_i, $\mathcal{S}_{\mathrm{I}}$ responses in one of the following two ways.
  - If $i \neq \kappa$, $\mathcal{S}_{\mathrm{I}}$ randomly selects $x_2 \in_R \mathbb{Z}_q$ and attempts to simulate $IK[\mathrm{ID}_i]$ by setting $r_{\mathrm{ID}_i} = -\frac{y}{u_i} + r'_i$ where $r'_i \in_R \mathbb{Z}_q$. Therefore, $IK[\mathrm{ID}_i] = (d_0, d_1)$ where $d_0 = g_2^{-x_2} g_1^{u_i r'_i}$ and $d_1 = g_2^{-\frac{1}{u_i} + r'_i}$. Moreover, $\mathcal{S}_{\mathrm{I}}$ sets $OK_{\mathrm{ID}_i} = x_2$ and sends it back to $\mathcal{S}_{\mathrm{I}}$.
  - If $i = \kappa$, $\mathcal{S}_{\mathrm{I}}$ randomly selects $x_1 \in_R \mathbb{Z}_q$ and computes $IK[\mathrm{ID}_\kappa] = (d_0, d_1)$ where $d_0 = g_2^{x_1} g^{u_\kappa r_{\mathrm{ID}_\kappa}}$ and $d_1 = g^{r_{\mathrm{ID}_\kappa}}$ for $r_{\mathrm{ID}_\kappa} \in_R \mathbb{Z}_q$. Moreover, $\mathcal{S}_{\mathrm{I}}$ sets $OK_{\mathrm{ID}_\kappa} = \perp$.

  After adding the entry $(\mathrm{ID}_i, SK_{\mathrm{ID}_i} = (IK[\mathrm{ID}_i], \perp), OK_{\mathrm{ID}_i})$ into L, $\mathcal{S}_{\mathrm{I}}$ returns $IK[\mathrm{ID}_i]$.
- *Updated key query.* Upon receiving (ID_i, T_j), $\mathcal{S}_{\mathrm{I}}$ checks whether there exists an entry $(\mathrm{ID}_i, SK_{\mathrm{ID}_i}, OK_{\mathrm{ID}_i})$ in L: if not, $\mathcal{S}_{\mathrm{I}}$ aborts; otherwise, $\mathcal{S}_{\mathrm{I}}$ fetches such entry and responds in the following two cases.

- If $i \neq \kappa$, $\mathcal{S}_{\mathrm{I}}$ selects $r_{T_j} \in_R \mathbb{Z}_q$ and after setting $S = S \cup \{(\mathrm{ID}_i, T_j)\}$ returns $TK[\mathrm{ID}_i]_{T_j} = (d_{T_j0}, d_{T_j1})$ where $d_{T_j0} = g_2^{x_2} H_2(T_j)^{r_{T_j}}$ and $d_{T1} = g^{r_{T_j}}$.
- If $i = \kappa$, $\mathcal{S}_{\mathrm{I}}$ checks whether $j = \eta$: if so, $\mathcal{S}_{\mathrm{I}}$ aborts. Otherwise, set $r_{T_j} = -\frac{y}{v_j} + r'_{T_j}$ for random $r'_{T_j} \in_R \mathbb{Z}_q$ and after setting $S = S \cup \{(\mathrm{ID}_\kappa, T_j)\}$ returns $TK[\mathrm{ID}_i]_{T_j} = (d_{T_j0}, d_{T_j1})$ where $d_{T_j0} = g_2^{-x_1} g_1^{v_j r'_{T_j}}$ and $d_{T_j1} = g_2^{-\frac{1}{v_j} + r'_{T_j}}$.

Challenge: $\mathcal{A}_{\mathrm{I}}$ will submit two challenge messages M_0 and M_1 as well as ID^* and T^* with $(\mathrm{ID}^*, T^*) \notin S$. $\mathcal{S}_{\mathrm{I}}$ checks that whether $\mathrm{ID}^* \neq \mathrm{ID}_\kappa$ or $T^* \neq T_\eta$, if so the security game is aborted. Otherwise, $\mathcal{S}_{\mathrm{I}}$ flips a fair binary coin $\nu \in \{0,1\}$ and returns an encryption of M_ν. The ciphertext is simulated as $CT^* = (M_\nu P, g^s, H_1(\mathrm{ID}^*)^s, H_2(T^*)^s)$. We note that if $P = e(g,g)^{xyz}$, if we let $s = z$, then $C_0 = M_\nu e(g,g)^{xyz} = M_\nu e(g_1, g_2)^z$, $C_1 = g^z, E_{\mathrm{ID}^*} = H_1(\mathrm{ID})^z = g^{u_\kappa z}, E_{T^*} = H_2(T^*)^z = g^{v_\eta z}$.

Phase 2: Phase 1 is repeated.

Guess: $\mathcal{A}_{\mathrm{I}}$ will submit a guess ν' of ν. If $\nu' = \nu$ $\mathcal{S}_{\mathrm{I}}$ outputs $\mu' = 0$, otherwise outputs $\mu' = 1$.

We note that since $\mathcal{A}_{\mathrm{I}}$ has the possibility of $\frac{1}{q_{H_1} q_{H_2}}$ in submitting $(\mathrm{ID}_\kappa, T_\eta)$ for challenge, the security game is finished successfully with the probability of $\frac{1}{q_{H_1} q_{H_2}}$ as well. Thus, we have $Pr[\nu \neq \nu' | \mu = 1] = \frac{1}{2} \frac{1}{q_{H_1} q_{H_2}}$. Since $\mathcal{S}_{\mathrm{I}}$ guesses $\mu' = 1$ when $\nu \neq \nu'$, we have $Pr[\mu' = \mu | \mu = 1] = \frac{1}{2} \frac{1}{q_{H_1} q_{H_2}}$. If $\mu = 0$, then $\mathcal{A}_{\mathrm{I}}$ sees an encryption of M_ν in the successful game. Therefore, we have $Pr[\mu' = \mu | \mu = 0] = \frac{1}{q_{H_1} q_{H_2}} (\frac{1}{2} + \epsilon_{\mathrm{I}})$. Finally, we have the overall advantage of $\mathcal{S}_{\mathrm{I}}$ in solving DBDH problem as $\frac{1}{q_{H_1} q_{H_2}} \epsilon_{\mathrm{I}}$.

Simulation of $\mathcal{S}_{\mathrm{II}}$ against Type-II Adversary

Setup: $\mathcal{S}_{\mathrm{II}}$ performs identically to that in $\mathcal{S}_{\mathrm{I}}$.

Phase 1: $\mathcal{S}_{\mathrm{II}}$ initializes an empty table list L, and two empty sets U and I. $\mathcal{A}_{\mathrm{II}}$ is allowed to issue queries in the following types.

- *H_1-query.* $\mathcal{S}_{\mathrm{II}}$ responses identically to that in $\mathcal{S}_{\mathrm{I}}$.
- *H_2-query.* $\mathcal{S}_{\mathrm{II}}$ responses identically to that in $\mathcal{S}_{\mathrm{I}}$.
- *Outsourcing key query.* Upon receiving ID_i, $\mathcal{S}_{\mathrm{II}}$ randomly selects $x_2 \in_R \mathbb{Z}_q$ and returns x_2 after adding $(\mathrm{ID}_i, SK_{\mathrm{ID}_i} = (\perp, \perp), OK_{\mathrm{ID}_i} = x_2)$ into L.
- *Private key query.* Upon receiving ID_i, if there exists such entry $(\mathrm{ID}_i, SK_{\mathrm{ID}_i}, OK_{\mathrm{ID}_i})$ in L, $\mathcal{S}_{\mathrm{II}}$ c hecks whether $i = \kappa$, if so $\mathcal{S}_{\mathrm{II}}$ aborts. Otherwise, $\mathcal{S}_{\mathrm{II}}$ sets $r_{\mathrm{ID}_i} = -\frac{y}{u_i} + r'_{\mathrm{ID}_i}$ for $r'_{\mathrm{ID}_i} \in_R \mathbb{Z}_q$ and computes $IK[\mathrm{ID}_i] = (d_0, d_1)$ where $d_0 = g_2^{-x_2} g_1^{u_i r'_{\mathrm{ID}_i}}$ and $d_1 = g_2^{-\frac{1}{u_i} + r'_{\mathrm{ID}_i}}$. After setting $U = U \cup \{\mathrm{ID}_i\}$, $\mathcal{S}_{\mathrm{II}}$ returns $IK[\mathrm{ID}_i]$.
- *Updated key query.* Upon receiving ID_i and T_j, if there exists such entry $(\mathrm{ID}_i, SK_{\mathrm{ID}_i}, OK_{\mathrm{ID}_i})$ in L, $\mathcal{S}_{\mathrm{II}}$ computes $TK[\mathrm{ID}_i]_{T_j} = (d_{T_j0}, d_{T_j1})$ where $d_{T_j0} = g_2^{x_2} H_2(T_j)^{r_{T_j}}$ and $d_{T_j1} = g^{r_{T_j}}$ for $r_{T_j} \in_R \mathbb{Z}_q$. $\mathcal{S}_{\mathrm{II}}$ continues to check whether $\mathrm{ID}_i \in U$, if not it sets $I = I \cup \{T_j\}$. Finally, return $TK[\mathrm{ID}_i]_{T_j}$.

Challenge: $\mathcal{A}_{\mathrm{I}}$ will submit two challenge messages M_0 and M_1 as well as ID^* and T^* with $\mathrm{ID}^* \notin U$ and $T^* \notin I$. $\mathcal{S}_{\mathrm{II}}$ checks that whether $\mathrm{ID}^* \neq \mathrm{ID}_\kappa$ or $T^* \neq T_\eta$, if so the security game is aborted. Otherwise, $\mathcal{S}_{\mathrm{II}}$ flips a fair binary coin $\nu \in \{0,1\}$ and returns an encryption of M_ν. The ciphertext is simulated as $CT^* = (M_\nu P, g^z, H_1(\mathrm{ID}^*)^z, H_2(T^*)^z)$.

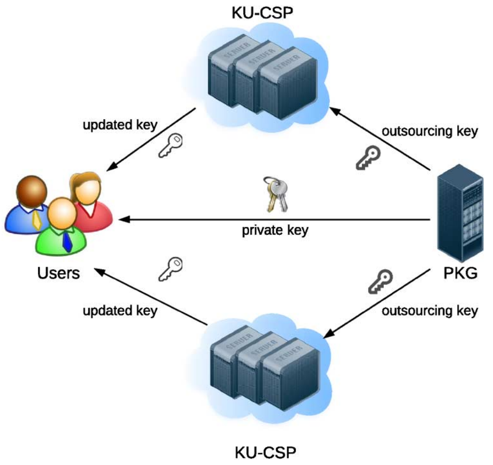


Fig. 5. System model with two KU-CSPs.

Phase 2: Phase 1 is repeated.

Guess: $\mathcal{A}_{\mathrm{II}}$ will submit a guess ν' of ν. If $\nu' = \nu$ $\mathcal{S}_{\mathrm{II}}$ outputs $\mu' = 0$, otherwise outputs $\mu' = 1$.

Similar to the analysis presented in the simulation of $\mathcal{S}_{\mathrm{I}}$ against adversary-I, we have the overall advantage of $\mathcal{S}_{\mathrm{II}}$ in solving DBDH problem as $\frac{1}{q_{H_1} q_{H_2}} \epsilon_{\mathrm{II}}$. □

5 Advanced Construction under Refereed Delegation of Computation Model

In this section, we will attempt to propose a security enhanced construction under the under the recently formalized RDoC model.

5.1 Advanced Construction

RDoC model originates from the model of refereed games in [8], and is later formalized in [9] and [10]. In RDoC model, the client is able to interact with multiple servers and it has a right output as long as there exists one server that follows the proposed protocol. One of the most advantages of RDoC over traditional model with single server is that the security risk on the single server is reduced to multiple servers involved in. As the result of both the practicality and utility, RDoC model recently has been widely utilized in the literature of outsourced computation [9], [10], [11], [7], [12].

In order to apply RDoC to our setting, we introduce another $k-1$ independent KU-CSPs. For simplicity, in the rest of paper, we only focuse on the case that $k = 2$ as shown in Fig. 5. Furthermore, we have three requirements in such model: 1) At least one of the KU-CSPs is honest. 2) Computational complexity at the honest KU-CSP is not much more than the other required to perform revocation. 3) PKG's running time would be much smaller than required to directly perform revocation.

We figure out that the challenge to realize such advanced construction is to demand that OK_{ID} and $IK[\mathrm{ID}]$ cannot be leaked at the same time. To achieve this goal, we randomly split $OK_{\mathrm{ID}} = x_2$ into $OK_{\mathrm{ID}}^{(1)} = x_{21}$ and $OK_{\mathrm{ID}}^{(2)} = x_{22}$ which will be separately used by the two KU-CSPs to produce partial time component $TK[\mathrm{ID}]_T^{(1)}$ and $TK[\mathrm{ID}]_T^{(2)}$. After receiving the two partial time components, user performs a production

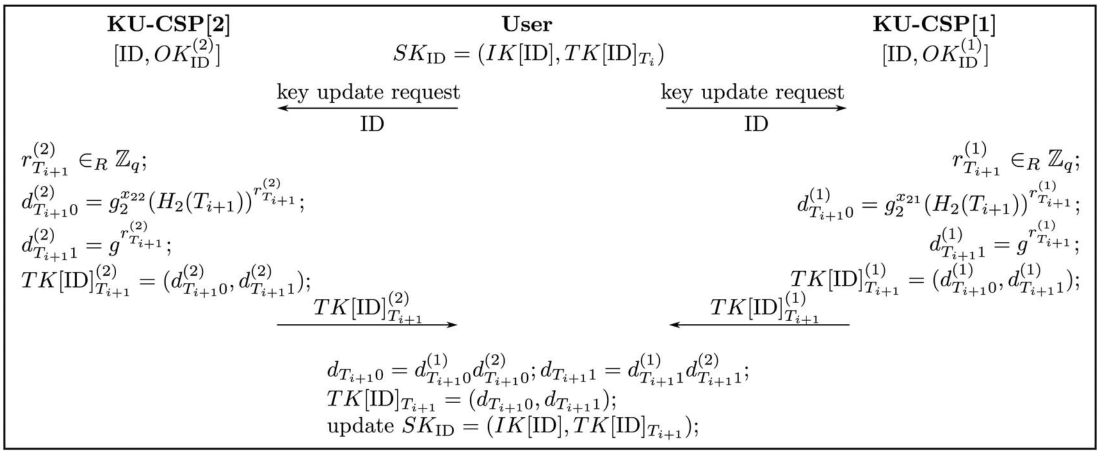

KU-CSP[2]
$[\mathrm{ID}, OK_{\mathrm{ID}}^{(2)}]$
User
$SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$
KU-CSP[1]
$[\mathrm{ID}, OK_{\mathrm{ID}}^{(1)}]$
key update request
ID
key update request
ID
$r_{T_{i+1}}^{(2)} \in_R \mathbb{Z}_q;$
$d_{T_{i+1}0}^{(2)} = g_2^{x_{22}}(H_2(T_{i+1}))^{r_{T_{i+1}}^{(2)}};$
$d_{T_{i+1}1}^{(2)} = g^{r_{T_{i+1}}^{(2)}};$
$TK[\mathrm{ID}]_{T_{i+1}}^{(2)} = (d_{T_{i+1}0}^{(2)}, d_{T_{i+1}1}^{(2)});$
$r_{T_{i+1}}^{(1)} \in_R \mathbb{Z}_q;$
$d_{T_{i+1}0}^{(1)} = g_2^{x_{21}}(H_2(T_{i+1}))^{r_{T_{i+1}}^{(1)}};$
$d_{T_{i+1}1}^{(1)} = g^{r_{T_{i+1}}^{(1)}};$
$TK[\mathrm{ID}]_{T_{i+1}}^{(1)} = (d_{T_{i+1}0}^{(1)}, d_{T_{i+1}1}^{(1)});$
$TK[\mathrm{ID}]_{T_{i+1}}^{(2)}$
$TK[\mathrm{ID}]_{T_{i+1}}^{(1)}$
$d_{T_{i+1}0} = d_{T_{i+1}0}^{(1)} d_{T_{i+1}0}^{(2)}; d_{T_{i+1}1} = d_{T_{i+1}1}^{(1)} d_{T_{i+1}1}^{(2)};$
$TK[\mathrm{ID}]_{T_{i+1}} = (d_{T_{i+1}0}, d_{T_{i+1}1});$
update $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_{i+1}});$

Fig. 6. KeyUpdate and KeyCombine in advanced construction.

to make a combination and obtains the final updated key (i.e. time component for private key).

Since the setup, encryption and decryption phases operate exactly as before, we will introduce the KeyCombine algorithm and only provide the key generation and revocation for the advanced construction as follows.

- KeyGen$(MK, \mathrm{ID}, RL, TL, PK)$: The algorithm is presented similar to that in our proposed construction in Section 4. The only difference is that PKG does not directly send $OK_{\mathrm{ID}} = x_2$ to KU-CSP, but makes a further random split on x_2 to obtain x_{21} and x_{22} with $x_2 = x_{21} + x_{22} \bmod q$. Finally, PKG sends $OK_{\mathrm{ID}}^{(l)} = x_{2l}$ to l-th KU-CSP for $l = 1, 2$.
- KeyUpdate$(RL, \mathrm{ID}, T_{i+1}, OK_{\mathrm{ID}}^{(l)})$: Upon receiving the key-update request on ID, the l-th KU-CSP checks whether ID exists in the revocation list RL, if so the key update is aborted. Otherwise, it fetches the corresponding entry $(\mathrm{ID}, OK_{\mathrm{ID}}^{(l)} = x_{2l})$ in the user list UL and computes $TK[\mathrm{ID}]_{T_{i+1}}^{(l)} = (d_{T_{i+1}0}^{(l)}, d_{T_{i+1}1}^{(l)})$ as shown in Fig. 6. Finally send the updated partial time component back to user.
- KeyCombine$(TK[\mathrm{ID}]_{T_{i+1}}^{(1)}, TK[\mathrm{ID}]_{T_{i+1}}^{(2)})$: Upon receiving $TK[\mathrm{ID}]_{T_{i+1}}^{(1)}$ and $TK[\mathrm{ID}]_{T_{i+1}}^{(2)}$, user performs a key combination by computing $d_{T_{i+1}0}$ and $d_{T_{i+1}1}$ as the paradigm shown in Fig. 6 to obtain $TK[\mathrm{ID}]_{T_{i+1}}$. Finally update SK_{ID} as $(IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_{i+1}})$.

5.2 Security Analysis

As a stronger adversary model, RDoC captures much more meaning beyond the "honest-but-curious" sense, that is curious user is allowed to cooperate with at most $k-1$ servers if k servers are involved. To accomodate to this case, we modify the private key oracle slightly to adapt to a pair of outsourcing keys and introduce another outsourcing key extraction oracle for Type-I adversary as follows. It is noted that the challenger is required to maintain an empty set R to restrict adversary accessing the whole outsourcing key for some identity. This coincides with the assumption that at least one of the KU-CSPs is honest.

- *Private key extraction oracle.* Upon receiving private key request on ID, challenger runs $KeyGen$ to obtain the private key $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_T)$ and a pair of outsourcing keys $OK_{\mathrm{ID}} = (OK_{\mathrm{ID}}^{(1)}, OK_{\mathrm{ID}}^{(2)})$. After adding the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ into L, return $IK[\mathrm{ID}]$.
- *Outsourcing key extraction oracle.* Upon receiving the partial outsourcing key request on ID to the $l-$th KU-CSP, challenger firstly checks whether $(\mathrm{ID}, 2-l+1) \in R$. If so the oracle is aborted. Otherwise, if there exists an entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, after setting $R = R \cup \{(\mathrm{ID}, l)\}$ return $OK_{\mathrm{ID}}^{(l)}$.

Theorem 2. *The advanced construction is secure in the sense of IND-ID-CPA in random oracle under the assumption that DBDH problem is intractable.*

Proof. Since the proof technique is quite similar to that used in the proof of theorem 1, we would only provide a sketch here.

Suppose $\mathcal{A}_{\mathrm{I}}$ and $\mathcal{A}_{\mathrm{II}}$ has the advantage ϵ_{I} and ϵ_{II} in attacking the proposed advanced construction in the sense of IND-ID-CPA for type-I and type-II adversary respectively. Then, we are to provide two simulators $\mathcal{S}_{\mathrm{I}}$ and $\mathcal{S}_{\mathrm{II}}$ to simulate two games (i.e. CPA security game for type-I and type-II adversary) between challenger and adversary.

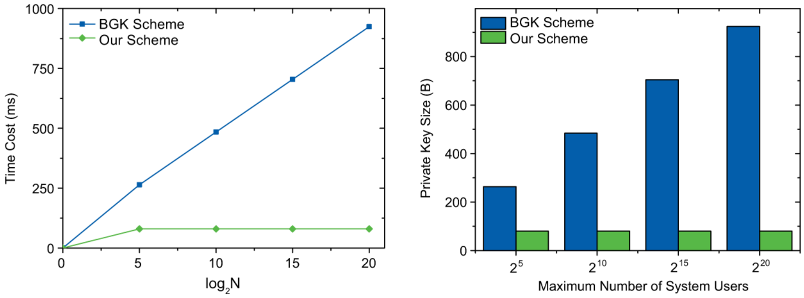
We specify that comparing with single KU-CSP adversary model (in Section 3.2), it allows a collusion between curious user and either of the KU-CSPs here. Correspondingly, comparing with the simulators in the proof of theorem 1, $\mathcal{S}_{\mathrm{II}}$ is identical and $\mathcal{S}_{\mathrm{I}}$ needs to simulate another outsourcing key extraction oracle.

In the additional outsourcing key extraction oracle, upon receiving the input (ID, l), $\mathcal{S}_{\mathrm{I}}$ firstly examines that whether $(\mathrm{ID}, 2-l+1)$ is queried. If so, the oracle is aborted since $\mathcal{A}_{\mathrm{I}}$ is not allowed to collude with both of the KU-CSPs. Otherwise, $\mathcal{A}_{\mathrm{I}}$ fetches the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L and returns $OK_{\mathrm{ID}}^{(l)}$ back to adversary. □

TABLE 1
Efficiency Comparison for Stages in Revocable IBE

	Our Scheme	IBE without Revocation [4]
Setup	83.764 ms	80.233 ms
Key-Issuing	40.369 ms	20.121 ms
Encryption	39.840 ms	24.595 ms
Decryption	21.278 ms	10.285 ms
Key-Update	10.300 ms[1]	—-

[1] This time cost is evaluated at KU-CSP.



(a) Responding Time for Single Key Generation Request

(b) Private Key Size

Fig. 7. Comparisons in key-issuing (N is the maximum number of users in the system).

6 Performance Evaluation

In this section, we will provide a thorough experimental evaluation of the construction proposed in Section 4. We build our testbed by using 64-bit M2 high-memory quadruple extra large Linux servers in Amazon EC2 platform as KU-CSP, and a Linux machine with Intel(R) Core(TM)2 Duo CPU clocked at 2.40 GHz and 2 GB of system memory as the user and PKG. Note that in all the evaluations, the groups $\mathbb{G}$ and $\mathbb{G}_T$ are selected in 160-bit and 512-bit length respectively.

6.1 Performance Evaluation for Overall Scheme

Firstly, we aim to evaluate the efficiency of our outsourced revocable scheme by comparing the total time taken during each stage with the original IBE [4] which does not consider revocation.

In Table 1, we examine the time cost of executing individual stage by the both schemes. It is not surprising to see that our scheme takes more time because we consider the revocability issue. Note that our scheme shares the same setup algorithm with the IBE scheme in [4]. Our key-issuing stage is relative longer than that in the IBE scheme [4]. This is because we embed a time component into each user's private key to allow periodically update for revocation, resulting that some additional computations[2] are needed in our scheme to initialize this component. Our encryption and decryption is slightly longer than the IBE scheme [4], which is also due to the existence of the time component. The user needs to perform an additional encryption/decryption for this component, rather than just encrypt/decrypt the identity component.

To sum up, our revocable scheme achieves both identity-based encryption/decryption and revocability without introducing significant overhead compared to the original IBE scheme [4] (our execution time is still within millisecond).

6.2 Performance Evaluation for Revocation

Secondly, we attempt to simulate the scenario of multi-user revocation, and show an extensive comparison between our outsourced revocation scheme and another revocable IBE scheme–BGK scheme [5]. Note that in this set of experiments, we use a 32-bit integer to identify each node in binary tree which is utilized in BGK scheme [5] for managing users. Our comparison is in terms of the key-issuing stage and the key-update stage.

6.2.1 Key-Issuing Stage

In Fig. 7(a), we vary the maximum number of users in the system and show the responding time for a single key generation request. It is not hard to see that the responding time in BGK scheme [5] is in propotion of $O(log_2(N))$ where N is the maximum number of users in system. This is because a binary tree is utilized to manage all the users, each leaf node of which is assigned to a single user in system. During key-issuing, PKG has to perform computation on all the nodes in the path from the corresponding leaf node to root node. Compared to the logarithmically growing efficiency in [5], our scheme achieves constant efficiency (nearly six modular exponentiation in $\mathbb{G}$) in single key-issuing.

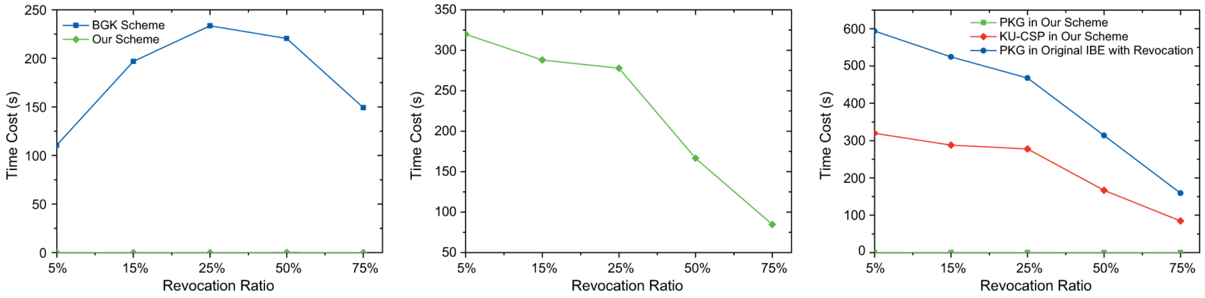
Correspondingly, we show the comparison on private key size in Fig. 7(b). Due to the same reason of demanding for computation on all the nodes in path from leaf node to root node, the previous approach [5] has an increasing private key size, whereas ours achieves constant key size (nearly four element in group $\mathbb{G}$).

Besides the better performance in efficiency and private key size, another advantage of our scheme over the previous work [5] is that it supports dynamic number of users. Specifically, the previous work [5] requires to fix the maximum number of users in system initially to facilitate building the binary tree. Once the maximum number is fixed, it is difficult to add users exceeding this bound. Ours does not have such a drawback, and flexibly supports dynamic management of users.

6.2.2 Key Update Stage

In this experiment, we randomly pick 5% to 75% users and compare the total time of updating private keys for the rest users. For simplicity, we just illustrate an example and compare the key-update time at PKG in revocation in the case of 2^{15} system users in Fig. 8(a). It can be seen that the efficiency curve of BGK scheme [5] shows a parabolic shape, and at the 25% revocation ratio, the efficiency achieves the lowest point in our evaluation. This is because it is the gap that the leaf nodes to be revoked has a large number but low aggregation degree, which requires that we have to update a lot of internal nodes for key-update. However, in our scheme, such a behavior is avoided, and just a negligible constant time is taken at PKG. More generally, this constant key-update efficiency is

[2] In our implementation, we read and hash the system time for current time period, and generate the time componenet in a way similar that for identity



(a) Key Update at PKG (b) Key Update at KU-CSP (c) Key Update Comparison with [4]

Fig. 8. Comparisons in key update (Case: 2^{15} system users).

actually achieved by our scheme with regardless to the number of system users since we delegate the revocation to KU-CSP, but BGK scheme [5] requires an increasing time cost with the number of system users (Table 2 shows the underlying tendency in more cases).

Accordingly, we also show the time cost at KU-CSP in our scheme for updating private keys for all the unrevoked users in the revocation ratio ranging from 5% to 75%. It must be pointed out, though, such a time cost is growing with the number of users in each case as with the performance of PKG in [5], this computation is conducted at cloud, which typically has abundant resources.

Moreover, we evaluate the communication cost for each user's key-update request in Amazon EC2 cloud environment, which is 87 ms. Note that such an overhead includes the time consuming for transmission and authentication at Amazon EC2 cloud platform.

6.3 Performance Evaluation for Outsourcing

Thirdly, we will show the effectiveness of outsourcing computation in our scheme by comparing the efficiency of our scheme with that of the direct revocable scheme [4] which shares a similar revocation idea with ours but does not considers outsourcing. Recall that in [4] Boneh et al. suggested that users renew their private keys periodically and senders use the receivers' identities concatenated with current time period for encryption. Compared with the work [4], we consider outsourcing the overhead computation at PKG to KU-CSP for efficient revocation. The effects of outsourcing is illustrated in Fig. 8(c) in the case of 2^{15} system users.

It can be easily seen that the computation at KU-CSP is close to that in PKG in Boneh's revocable scheme [4]. This is because we build identity component and time component in our scheme, and each component has a similar structure with original private key in [4]. Therefore, in key-update, the computation demanded for updating time component and re-issuing private key are nearly the same. As we emphasize before, such computation is conducted by KU-CSP typically with abundant resources, which will not seriously affect the efficiency of our system.

TABLE 2
Key-Update in BGK Scheme in Varying System Users

Revocation Ratio	2^5	2^{10}	2^{15}	2^{20}
5%	132.646 ms	3.66 s	1.845 min	0.998 h
15%	212.162 ms	6.607 s	3.283 min	1.879 h
25%	194.135 ms	7.232 s	3.894 min	2.15 h
50%	189.851 ms	9.63 s	3.674 min	1.99 h
75%	133.072 ms	4.186 s	2.488 min	1.207 h

7 Related Work

7.1 Revocable IBE

Introduced by [13] and firstly implemented by Boneh and Franklin [4] as well as [14], IBE has been researched intensively in cryptographic community.

On the aspect of construction, these first schemes [4], [14] were proven secure in random oracle. Some subsequent systems achieved provable secure in standard model under selective-ID security [15], [16] or adaptive-ID security [17]–[19]. Recently, there have been multiple lattice-based constructions for IBE systems [20]–[22].

Nevertheless, concerning on revocable IBE, there is little work presented. As mentioned before, Boneh and Franklin's suggestion [4] is more a viable solution but impractical. Hanaoka et al. [23] proposed a way for users to periodically renew their private keys without interacting with PKG. However, the assumption required in their work is that each user needs to possess a tamper-resistant hardware device. Another solution is mediator-aided revocation [24], [25]: In this setting there is a special semi-trusted third party called a mediator who helps users to decrypt each ciphertext. If an identity is revoked then the mediator is instructed to stop helping the user. Obviously, it is impractical since all users are unable to decrypt on their own and they need to communicate with mediator for each decryption. Recently, Lin et al. [26] proposed a space efficient revocable IBE mechanism from non-monotonic Attribute-Based Encryption (ABE), but their construction requires $O(r)$ times bilinear pairing operations for a single decryption where r is the number of revoked users.

As far as we know, the revocable IBE scheme presented by Boldyreva et al. [5] remains the most effective solution right now. Libert and Vergnaud [27] improved Boldyreva's construction [5] to achieve adaptive-ID security. Their work focused on security enhanced, but inherits the similar disadvantage as Boldyreva's original construction [5]. As we mentioned before, they are short in storage for both private key at user and binary tree structure at PKG.

7.2 Other Revocation Technique

Another work related to us originates from Yu et al. [28]. The authors utilized proxy re-encryption to propose a revocable ABE scheme. The trusted authority only needs to update master key according to attribute revocation status in each time period and issue proxy re-encryption key to proxy servers. The proxy servers will then re-encrypt ciphertext using the re-encryption key to make sure all the unrevoked users can perform successful decryption.

We specify that a third party service provider is introduced in both Yu et al. [28] and this work. Differently, Yu et al. [28] utilized the third party (work as a proxy) to realize revocation through re-encrypting ciphertext which is only adapt to the special application that the ciphertext is stored at the third party. However, in our construction the revocation is realized through updating private keys for unrevoked users at cloud service provider which has no limits on the location of ciphertext.

7.3 Outsourcing Computation

The problem that how to securely outsource different kinds of expensive computations has drawn considerable attention from theoretical computer science community for a long time. Chaum and Pedersen [29] firstly introduced the notion of wallets with observers, a piece of secure hardware installed on the client's computer to perform some expensive computations. Atallah et al. [30] presented a framework for secure outsourcing of scientific computations such as matrix multiplication and quadrature. Nevertheless, the solution used the disguise technique and thus leaded to leakage of private information. Hohenberger and Lysyanskaya [9] proposed the first outsource-secure algorithm for modular exponentiations based on pre-computation and server-aided computation. Atallah and Li [31] investigated the problem of computing the edit distance between two sequences and presented an efficient protocol to securely outsource sequence comparison with two servers. Furthermore, Benjamin and Atallah [32] addressed the problem of secure outsourcing for widely applicable linear algebraic computations. Nevertheless, the proposed protocol required the expensive operations of homomorphic encryption. Atallah and Frikken [12] further studied this problem and gave improved protocols based on the so-called weak secret hiding assumption. Chen et al. [11] made an efficiency improvement on the work [9] and proposed a new scheme for outsourcing single/simultaneous modular exponentiations.

7.4 Cloud Computing

Cloud Computing is the latest term encapsulating the delivery of computing resources as a service [33]. It is the current iteration of utility computing and returns to the model of "renting" resources. Leveraging cloud computing is today, the defacto means of deploying internet scale systems and much of the internet is tethered to a large number of cloud service providers.

In this paper, the KU-CSP provides computing service in the Infrastructure as a service (IaaS) model, which provides the raw materials of cloud computing, such as processing, storage and other forms of lower level network and hardware resources in a virtual, on demand manner via the Internet. Differing from traditional hosting services with which physical servers or parts thereof are rented on a monthly or yearly basis, the cloud infrastructure is rented as virtual machines on a per-use basis and can scale in and out dynamically, based on customer needs. Such on-demand scalability is enabled by the recent advancements in virtualisation and network management. IaaS users do not need to manage or control the underlying cloud infrastructure but have control over operating systems, storage, deployed applications, and in some cases limited control of select networking components (e.g. host firewalls) [34]. Typical IaaS examples are Amazon EC2 and S3 where computing and storage infrastructure are open to public access in a utility fashion. We specify that in this work we also aim to utilize outsourcing computation technique to deliver overhead computation to KU-CSP so that PKG is able to be offline in key-update.

Recently, a number of works have been proposed to tackle practical problems in the cloud aided model, which explores a joint point between cloud computing and outsourcing computation. Wang et al. [35] presented efficient mechanisms for secure outsourcing of linear programming computation. Green et al. [36] proposed a new method for efficiently and securely outsourcing decryption of attribute-based encryption ciphertexts. They also showed their performance evaluation in Amazon EC2 platform as the simulation of cloud environment. Some other works about outsourced ABE include [37]–[39]. Especially, [38] outsourced the encryption in ABE with the map-reduce technique in cloud computing. Zhang et al. [40] proposed a novel outsourced image recovery service architecture, which exploits different domain technologies and takes security, efficiency, and design complexity into consideration from the very beginning of the service flow.

8 Conclusion

In this paper, focusing on the critical issue of identity revocation, we introduce outsourcing computation into IBE and propose a revocable scheme in which the revocation operations are delegated to CSP. With the aid of KU-CSP, the proposed scheme is full-featured: 1) It achieves constant efficiency for both computation at PKG and private key size at user; 2) User needs not to contact with PKG during key-update, in other words, PKG is allowed to be offline after sending the revocation list to KU-CSP; 3) No secure channel or user authentication is required during key-update between user and KU-CSP.

Furthermore, we consider to realize revocable IBE under a stronger adversary model. We present an advanced construction and show it is secure under RDoC model, in which at least one of the KU-CSPs is assumed to be honest. Therefore, even if a revoked user and either of the KU-CSPs collude, it is unable to help such user re-obtain his/her decryptability.

Finally, we provide extensive experimental results to demonstrate the efficiency of our proposed construction.

Acknowledgment

This work was supported by National Natural Science Foundation of China (No. 61100224, No. 61272455 and No. 61272423), the Guangzhou Zhujiang Science and Technology Future Fellow Fund (No. 2012J2200094), Distinguished Young Scholars Fund of Department of Education (No. Yq2013126), Guangdong Province, China 111 Project (No. B08038), Doctoral Fund of Ministry of Education of China

(No. 20130203110004), Program for New Century Excellent Talents in University (No. NCET-13-0946). Besides, Lou's work is supported by US National Science Foundation under grant (CNS-1217889).

References

[1] W. Aiello, S. Lodha, and R. Ostrovsky, "Fast digital identity revocation," in *Advances in Cryptology (CRYPTO'98)*. New York, NY, USA: Springer, 1998, pp. 137–152.

[2] V. Goyal, "Certificate revocation using fine grained certificate space partitioning," in *Financial Cryptography and Data Security*, S. Dietrich and R. Dhamija, Eds. Berlin, Germany: Springer, 2007, vol. 4886, pp. 247–259.

[3] F. Elwailly, C. Gentry, and Z. Ramzan, "Quasimodo: Efficient certificate validation and revocation," in *Public Key Cryptography (PKC'04)*, F. Bao, R. Deng, and J. Zhou, Eds. Berlin, Germany: Springer, 2004, vol. 2947, pp. 375–388.

[4] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology (CRYPTO '01)*, J. Kilian, Ed. Berlin, Germany: Springer, 2001, vol. 2139, pp. 213–229.

[5] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proc. 15th ACM Conf. Comput. Commun. Security (CCS'08)*, 2008, pp. 417–426.

[6] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology (EUROCRYPT'05)*, R. Cramer, Ed. Berlin, Germany: Springer, 2005, vol. 3494, pp. 557–557.

[7] R. Canetti, B. Riva, and G. N. Rothblum, "Two 1-round protocols for delegation of computation," *Cryptology ePrint Archive*, Rep. 2011/518, 2011 [online]. Available: http://eprint.iacr.org/2011/518.

[8] U. Feige and J. Kilian, "Making games short (extended abstract)," in *Proc. 29th Annu. ACM Symp. Theory Comput. (STOC'97)*, 1997, pp. 506–516.

[9] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in *Proc. 2nd Int. Conf. Theory Cryptography (TCC'05)*, 2005, pp. 264–282.

[10] R. Canetti, B. Riva, and G. Rothblum, "Two protocols for delegation of computation," in *Information Theoretic Security*, A. Smith, Ed. Berlin, Germany: Springer, 2012, vol. 7412, pp. 37–61.

[11] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New and secure outsourcing algorithms of modular exponentiations," in *Proc. 17th Eur. Symp. Res. Comput. Security (ESORICS)*, 2012, pp. 541–556.

[12] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proc. 5th ACM Symp. Inf. Comput. Commun. Security (ASIACCS'10)*, 2010, pp. 48–59.

[13] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology (CRYPTO)*, G. Blakley and D. Chaum, Eds. Berlin, Germany: Springer, 1985, vol. 196, pp. 47–53.

[14] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *Cryptography and Coding*, B. Honary, Ed. Berlin/Heidelberg: Springer, 2001, vol. 2260, pp. 360–363.

[15] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Advances in Cryptology (EUROCRYPT'03)*, E. Biham, Ed. Berlin, Germany: Springer, 2003, vol. 2656, pp. 646–646.

[16] D. Boneh and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles," in *Advances in Cryptology (EUROCRYPT'04)*, C. Cachin and J. Camenisch, Eds. Berlin, Germany: Springer, 2004, vol. 3027, pp. 223–238.

[17] D. Boneh and X. Boyen, "Secure identity based encryption without random oracles," in *Advances in Cryptology (CRYPTO'04)*, M. Franklin, Ed. Berlin, Germany: Springer, 2004, vol. 3152, pp. 197–206.

[18] B. Waters, "Efficient identity-based encryption without random oracles," in *Advances in Cryptology (EUROCRYPT'05)*, R. Cramer, Ed. Berlin, Germany: Springer, 2005, vol. 3494, pp. 114–127.

[19] C. Gentry, "Practical identity-based encryption without random oracles," in *Advances in Cryptology (EUROCRYPT'06)*, S. Vaudenay, Ed. Berlin, Germany: Springer, 2006, vol. 4004, pp. 445–464.

[20] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 40th Annu. ACM Symp. Theory Comput. (STOC'08)*, 2008, pp. 197–206.

[21] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (h)ibe in the standard model," in *Advances in Cryptology (EUROCRYPT'10)*, H. Gilbert, Ed. Berlin, Germany: Springer, 2010, vol. 6110, pp. 553–572.

[22] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," in *Advances in Cryptology (EUROCRYPT'10)*, H. Gilbert, Ed. Berlin, Germany: Springer, 2010, vol. 6110, pp. 523–552.

[23] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai, "Identity-based hierarchical strongly key-insulated encryption and its application," in *Advances in Cryptology (ASIACRYPT'05)*, B. Roy, Ed. Berlin, Germany: Springer, 2005, vol. 3788, pp. 495–514.

[24] D. Boneh, X. Ding, G. Tsudik, and C. Wong, "A method for fast revocation of public key certificates and security capabilities," in *Proc. 10th USENIX Security Symp.*, 2001, pp. 297–308.

[25] B. Libert and J.-J. Quisquater, "Efficient revocation and threshold pairing based cryptosystems," in *Proc. 22nd Annu. Symp. Principles Distrib. Comput.*, 2003, pp. 163–171.

[26] H. Lin, Z. Cao, Y. Fang, M. Zhou, and H. Zhu, "How to design space efficient revocable IBE from non-monotonic ABE," in *Proc. 6th ACM Symp. Inf. Comput. Commun. Security (ASIACCS'11)*, 2011, pp. 381–385.

[27] B. Libert and D. Vergnaud, "Adaptive-id secure revocable identity-based encryption," in *Topics in Cryptology (CT-RSA'09)*, M. Fischlin, Ed. Berlin, Germany: Springer, 2009, vol. 5473, pp. 1–15.

[28] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proc. 5th ACM Symp. Inf. Comput. Commun. Security (ASIACCS'10)*, 2010, pp. 261–270.

[29] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *Proc. 12th Annu. Int. Cryptology Conf. Adv. Cryptology (CRYPTO'92)*, 1993, pp. 89–105.

[30] M. J. Atallah, K. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," in *Trends in Software Engineering*, M. V. Zelkowitz, Ed. New York, NY, USA: Elsevier, 2002, vol. 54, pp. 215–272.

[31] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," *Int. J. Inf. Security*, vol. 4, pp. 277–287, 2005.

[32] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proc. 6th Annu. Conf. Privacy Security Trust (PST'08)*, 2008, pp. 240–245.

[33] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[34] P. Mell and T. Grance, "*The NIST Definition of Cloud Computing*," Nat. Inst. Stand. Technol., Tech. Rep. SP 800-145, 2011.

[35] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2011, pp. 820–828.

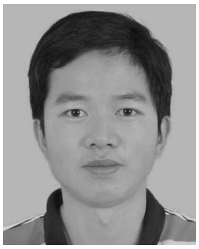
[36] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proc. 20th USENIX Conf. Security (SEC'11)*, 2011, pp. 34–34.

[37] Z. Zhou and D. Huang, "Efficient and secure data storage operations for mobile cloud computing," in *Proc. 8th Int. Conf. Netw. Service manage.*, 2012, pp. 37–45.

[38] J. Li, C. Jia, J. Li, and X. Chen, "Outsourcing encryption of attribute-based encryption with mapreduce," in *Information and Communications Security*. Berlin, Heidelberg: Springer, 2012, vol. 7618, pp. 191–201.

[39] J. Li, X. Chen, J. Li, C. Jia, J. Ma, and W. Lou, "Fine-grained access control system based on outsourced attribute-based encryption," in *Proc. 18th Eur. Symp. Res. Comput. Security (ESORICS)*, 2013, pp. 592–609.

[40] B. Zhang, J. Wang, K. Ren, and C. Wang, "Privacy-assured outsourcing of image reconstruction service in cloud," *IEEE Trans. Emerging Topics Comput.*, vol. 1, no. 1, p. 166–177, Jul./Dec. 2013.



Jin Li received the BS degree in mathematics from Southwest University, Chongqing, China, in 2002. He recieved the PhD degree in information security from Sun Yat-sen University, Guangdong, China, in 2007. Currently, he works at Guangzhou University, China. His research interests include applied cryptography and security in cloud computing.

Jingwei Li received the BS degree in mathematics in 2005 from the Hebei University of Technology, China. Currently, he is working toward the PhD degree in computer technology in Nankai University, China. His current interests include applied cryptography and cloud security.

Xiaofeng Chen received the BS and MS degrees in mathematics in Northwest University, X'ian, China. He received the PhD degree in cryptography in Xidian University, X'ian, China, in 2003. Currently, he works at Xidian University as a professor. His research interests include applied cryptography and cloud security.

Chunfu Jia received the PhD degree in engineering from Nankai University, China, in 1996. He has finished the post-doctoral research from the University of Science and Technology of China, Hefei. Currently, he is a professor in Nankai University, China. His current interests include computer system security, network security, trusted computing, malicious code analysis, etc.

Wenjing Lou received the BS and MS degrees in computer science and engineering from Xi'an Jiaotong University, China, the MASc degree in computer communications from Nanyang Technological University, Singapore, and the PhD degree in electrical and computer engineering, from the University of Florida. She is now an associate professor in the Computer Science Department, Virginia Polytechnic Institute and State University, Blacksburg.