Revocable Identity-based Signcryption Scheme Without Random Oracles

Xiangsong Zhang[1], Zhenhua Liu[2,3], Yupu Hu[4] and Tsuyoshi Takagi[5]
(Corresponding author: Zhenhua Liu)

School of Science, Xi'an Technological University, Xi'an, Shaanxi 710032, China[1]
School of Mathematics and Statistics, Xidian University, Xi'an, Shaanxi 710071, China[2]
Guangxi Experiment Center of Information Science, Guilin University of Electronic Technology, Guilin, Guangxi 541004, China[3]
State Key Laboratory of Integrated Services Network, Xidian University, Xi'an, Shaanxi 710071, China[4]
Faculty of Mathematics, Kyushu University, Fukuoka, 819-0395, Japan[5]
(Email: zhualiu@hotmail.com)
(Received Oct. 14, 2013; revised and accepted Nov, 25, 2014)

Abstract

Revocation functionality is crucial for the practicality of the public key cryptosystems including signcryption. When a user's private key is corrupted by hacking or the period of a contract expires, the cryptosystems must provide a revocation method to revoke the misbehaving/compromised user. However, little work has been published on key revocation in identity-based signcryption. We propose a revocable identity-based signcryption scheme. In the scheme, the master key is randomly divided into two parts: one is used to construct the initial key, the other is used to generate the updated key. Furthermore, they are used to periodically and re-randomly generate full private keys for non-revoked users. Thus, the proposed scheme can revoke users and resist key exposure. In the standard model, we prove the proposed scheme with IND-CCA2 security under the DBDH hardness assumption and EUF-CMA security under the CDH hardness assumption.

Keywords: Bilinear pairings, identity-based cryptography, provable security, revocation, signcryption

1 Introduction

Confidentiality, integrity, non-repudiation and authentication are the important requirements for many cryptographic applications. A traditional approach to achieve these requirements is to sign-then-encrypt the message. Signcryption [31] combines the functionality of digital signature and that of public-key encryption in a logical step, and provides the improvements on efficiency over traditional cryptographic mechanisms. The performance advantage makes signcryption useful in many applications, such as shared secret key authentication, resource-constrained network environments and electronic commerce [8, 10, 13, 14].

In an identity-based cryptosystem [23], the public key of a user can be arbitrary strings, such as an email address that uniquely identifies the user. The private key corresponding to the public key or identity is generated by a trusted key authority called key generation center (KGC). Compared with traditional public key cryptosystems using public key infrastructure (PKI), identity-based cryptosystem simplifies the key management problem by avoiding public key certificates. Since then, a large number of papers have been published in this area, including identity-based encryption schemes [3, 27], identity-based signature schemes [1, 9, 12, 20, 26] and identity-based signcryption schemes [1, 4, 6, 11, 15, 16, 17, 29, 30].

Key revocation is critical for the practicality of any public key cryptosystems including identity-based cryptosystem. For example, the private key corresponding to the public key has been stolen, the user has lost her private key, or the user is no longer a legitimate system user. In these cases, it is important that the public/private key pair be revoked or replaced by new keys. In the traditional PKI setting, a certification authority informs the senders about expired or revoked keys of the users via publicly available digital certificates and certificate revocation lists. Many efficient way to revoke users has been studied in numerous studies. However, there are only a few studies in the identity-based cryptosystem setting. To solve the problem of key revocation in the identity-based cryptosystem, Boneh and Franklin [3] suggested that the public key of a user be composed of identity information and time information (called BF revocation technique). Let u_i be a receiver's identity, and T be the current time index. The user's public key is denoted as $\mathrm{u}_i||T$, and the private key $sk_{\mathrm{u}_i,T}$ for non-revoked user u_i on each time

index T is issued by KGC. This means that all users, regardless of whether their keys have been exposed or not, have to periodically get in connect with the KGC, prove their identity and get new private keys. Tseng et al. used the BF revocation technique to propose fully secure revocable identity-based identity-based signature (RIBS) scheme [24] and encryption (RIBE) scheme [25] in the standard model. By the BF revocation technique, the key update complexity at each time index is $\mathcal{O}(n-r)$, with n the number of users and r the number of revoked users. Thus, their solution introduces huge overheads for the KGC that linearly increased in the number of users.

Furthermore, Boldyreva, Goyal and Kumar [2] proposed a new revocable identity-based encryption scheme which used a binary-tree data structure to settle the revocation problem (called BGK revocation technique) in 2008. BGK revocation technique reduces the KGC's periodic key update workload to $\mathcal{O}(r \log \frac{n}{r})$, and their scheme is proved to be selective-identity secure in the standard model. By making use of BGK's binary-tree data structure, Libert and Vergnaud (LV) [18] described an adaptive-identity secure and revocable identity-based encryption scheme, and Chen et al. [7] proposed selective-identity secure and revocable identity-based encryption scheme from lattices. The two schemes share the same key update complexity with the BGK scheme. Liu et al. [19] proposed a low-complexity key updating algorithm, which reduced the binary tree structure of BGK scheme to a tree of depth one, and constructed an efficient revocable identity-based encryption scheme. Most recently, Seo and Emura [21] showed all prior RIBE schemes except for using the BF technique were vulnerable to decryption key exposure attack, where an adversary, who has decryption key $dk_{\mathrm{u}^*,T}$ and key update ku_T, can always recover a part $(D_{x^*,0}, D_{x^*,1})$ of initial private key sk_{u^*} for some x^* if the challenged user u^* is not revoked in time T, and can always obtain a decryption key $dk_{\mathrm{u}^*,T^*} = (D_{x^*,0}, \tilde{D}_{x^*,0}, D_{x^*,1}, \tilde{D}_{x^*,1})$ by combination of the parts $(D_{x^*,0}, D_{x^*,1})$ of sk_{u^*} and $(\tilde{D}_{x^*,0}, \tilde{D}_{x^*,1})$ of ku_{T^*} if u^* is still not revoked in the challenge time T^*. For further details, please read this reference [21]. Then they revisited the Boldyreva et al. security model and proposed the first scalable and efficient RIBE scheme with decryption key exposure resistance. Furthermore, Seo and Emura [22] extended the revocation functionality to the hierarchical identity-based encryption (HIBE).

Signcryption is an important cryptographic primitive. However, little work has been published on revocable identity-based signcryption (RIBSC) schemes. Wu et al. [28] formalized the security model of identity-based signcryption with revocation functionality and proposed the first revocable identity-based signcryption scheme in 2012. Nevertheless, their scheme makes use of the BF revocation technique. Thus this requires the KGC to do work linear in the number of users, and does not scale well as the number of users grows. Moreover, the security of their scheme is demonstrated in the random oracle model. As shown in [5], a proof in the random oracle model can only serve as a heuristic argument and does not necessarily imply the security in the real implementation. Hence, the revocable identity-based signcryption scheme in [28] is not practically secure. In this paper, we focus on efficient identity-based signcryption schemes with revocation functionality without using the random oracles.

The rest of this paper is organized as follows. Some preliminaries are presented in Section 2. The formal model of revocable identity-based signcryption scheme and a concrete construction are detailed in Sections 3 and 4, respectively. We analyze the proposed scheme in Section 5. Finally, some concluding remarks are given in Section 6.

2 Preliminaries

In this section, we briefly review bilinear maps and some complexity assumptions. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of order p for some large prime p, and g be a generator of $\mathbb{G}$. A bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ should satisfy the following properties:

1) Bilinear: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$;

2) Non-degenerate: $e(g, g) \neq 1_{\mathbb{G}_T}$;

3) Computable: it is efficient to compute $e(u, v)$ for any $u, v \in \mathbb{G}$.

We say that $(\mathbb{G}, \mathbb{G}_T)$ are bilinear map groups if they satisfy these requirements above. In such groups, we describe the following intractability assumptions related to the security of our scheme.

Definition 1. *The challenger chooses $a, b, c, z \in \mathbb{Z}_p$ at random and then flips a fair binary coin $\beta \in \{0, 1\}$. If $\beta = 1$, it outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = e(g, g)^{abc})$. Otherwise, if $\beta = 0$, the challenger outputs the tuple $(g, A = g^a, B = g^b, C = g^c, Z = e(g, g)^z)$. The decisional bilinear Diffie-Hellman (DBDH) problem is to guess the value of β.*

An adversary, $\mathcal{C}$, has at least an ϵ advantage in solving the DBDH problem if

$$|Pr[\mathcal{C}(g, g^a, g^b, g^c, e(g, g)^{abc}) = 1] - Pr[\mathcal{C}(g, g^a, g^b, g^c, e(g, g)^z) = 1]| \geq 2\epsilon,$$

where the probability is oven the randomly chosen a, b, c, z and the random bits consumed by $\mathcal{C}$.

The (ϵ, t)-DBDH intractability assumption holds if no t-time adversary $\mathcal{C}$ has at least ϵ advantage in solving the DBDH problem.

Definition 2. *The challenger chooses $a, b \in \mathbb{Z}_p$ at random and outputs (g, g^a, g^b). The computational Diffie-Hellman (CDH) problem is to compute g^{ab}.*

An adversary, $\mathcal{C}$, has at least an ϵ advantage in solving the CDH problem if

$$Pr[\mathcal{C}(g, g^a, g^b) = g^{ab}] \geq \epsilon.$$

The (ϵ, t)-CDH intractability assumption holds if no t-time algorithm has the advantage at least ϵ in solving the CDH problem.

3 Formal Model of RIBSC Scheme

In this section, we define the formal definition of the syntax and the security notions of RIBSC scheme. Our syntax of RIBSC scheme is slightly different from Wu et al. [28]. The main differences are: (1) our key update (KeyUp) algorithm does not bind the identity with the time; (2) our full private key generation (FPKG) algorithm is probabilistic and supports key re-randomization, whereas Wu et al.'s one is deterministic and does not support key re-ranomization; (3) we increase a Revocation algorithm.

3.1 Generic Scheme

Let $\mathcal{M}, \mathcal{I}$ and $\mathcal{T}$ be a message space, an identity space, and a time index space, respectively. A RIBSC scheme consists of seven algorithms as follows.

- **Setup:** This is the (stateful) setup algorithm which takes as input the security parameter λ and the number of users N, and outputs public parameters mpk, a master secret key msk, an initial revocation list $RL = \phi$, and a state st.
- **Initial Private Key Generation:** This is the (stateful) initial private key generation (**IPKG**) algorithm which takes as input mpk, msk, an identity $\mathfrak{u} \in \mathcal{I}$, and outputs a secret key $sk_{\mathfrak{u}}$ associated with $\mathfrak{u}$ and an updated state st.
- **Key Update Generation:** This is the key update generation (**KeyUp**) algorithm which takes as input mpk, msk, the key update time $T \in \mathcal{T}$, the current revocation list RL, and st, and outputs a key update ku_T.
- **Full Private Key Generation:** This is the probabilistic full private key generation (**FPKG**) algorithm which takes as input $mpk, sk_{\mathfrak{u}}$, and ku_T, and outputs a decryption key $dk_{\mathfrak{u},T}$, or $\perp$ if $\mathfrak{u}$ has been revoked.
- **Signcryption:** This is the probabilistic signcryption (**SC**) algorithm which takes as input $mpk, T \in \mathcal{T}$, a sender's identity $\mathfrak{u}_s \in \mathcal{I}$ and decryption key $dk_{\mathfrak{u}_s,T}$, a receiver's identity $\mathfrak{u}_r \in \mathcal{I}$, and a message $M \in \mathcal{M}$, and outputs a ciphertext σ.
- **Designcryption:** This is the deterministic designcryption (**DSC**) algorithm which takes as input $mpk, T \in \mathcal{T}$, a sender's identity $\mathfrak{u}_s \in \mathcal{I}$, a receiver's identity $\mathfrak{u}_r \in \mathcal{I}$ and decryption key $dk_{\mathfrak{u}_r,T}$, and a ciphertext σ, and outputs M or $\perp$ if σ is an invalid ciphertext.
- **Revocation:** This is the stateful revocation (**REV**) algorithm which takes as input an identity to be revoked $\mathfrak{u} \in \mathcal{I}$, a revocation time $T \in \mathcal{T}$, the current revocation list RL, and a state st, and outputs an updated RL.

Every RIBSC scheme should satisfy the following consistency constraint that if

$$\sigma = \mathbf{SC}(mpk, \mathfrak{u}_s, \mathfrak{u}_r, T, dk_{\mathfrak{u}_s,T}, M),$$

then

$$\mathbf{DSC}(mpk, \mathfrak{u}_s, \mathfrak{u}_r, T, dk_{\mathfrak{u}_r,T}, \sigma) = M$$

holds. Next, we provide a security definition of RIBSC scheme that captures realistic threats including decryption key exposure.

3.2 Security Notions

Wu et al. [28] gave the security notions for a RIBSC scheme including the indistinguishability under adaptive chosen-ciphertext attack (*IND-RIBSC-CCA2*) and the existential unforgeablilty under adaptive chosen-message attack (*EUF-RIBSC-CMA*). This model is a natural extension of the security notions of the ordinary identity-based signcryption schemes [4, 16, 17, 30]. According to the generic scheme in Subsection 3.1, we will revise the extended security notions by allowing the adversary to access *full private key generation query* and *revocation query*.

For the *IND-RIBSC-CCA2* property, we consider the following game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

– **Initial.** $\mathcal{C}$ runs the algorithm **Setup** and obtains both the master public key parameters mpk and the master secret key msk. The adversary $\mathcal{A}$ is given mpk but the master secret is kept by the challenger.

– **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the challenger $\mathcal{C}$, in an adaptive fashion (i.e., one at time, with knowledge of the previous replies). The following queries are allowed:

  - *Initial private key generation query.* Upon receiving this query with identity $\mathfrak{u} \in \mathcal{I}$, the challenger $\mathcal{C}$ runs $\mathbf{IPKG}(mpk, msk, \mathfrak{u}, \mathsf{st}) \rightarrow sk_{\mathfrak{u}}$ and returns $sk_{\mathfrak{u}}$.
  - *Key update query.* Upon receiving this query with time index $T \in \mathcal{T}$, $\mathcal{C}$ runs $\mathbf{KeyUp}(mpk, msk, T, RL, \mathsf{st}) \rightarrow ku_T$ and returns ku_T.
  - *Revocation query.* Upon receiving this query with $\mathfrak{u} \in \mathcal{I}$ and $T \in \mathcal{T}$, $\mathcal{C}$ runs $\mathbf{REV}(mpk, \mathfrak{u}, T, RL, \mathsf{st}) \rightarrow RL$ and returns the updated revocation list RL.
  - *Full private key generation query.* Upon receiving this query with $\mathfrak{u} \in \mathcal{I}$ and $T \in$

$\mathcal{T}$, $\mathcal{C}$ runs $\mathbf{IPKG}(mpk, msk, \mathtt{u}, \mathtt{st}) \rightarrow sk_{\mathtt{u}}$, $\mathbf{KeyUp}(mpk, msk, T, RL, \mathtt{st}) \rightarrow ku_T$, and $\mathbf{FPKG}(mpk, \mathtt{u}, T, sk_{\mathtt{u}}, ku_T) \rightarrow dk_{\mathtt{u},T}$, and returns $dk_{\mathtt{u},T}$.

- *Signcryption query.* Upon receiving this query for a message $M \in \mathcal{M}$, a sender's identity $\mathtt{u}_s \in \mathcal{I}$, a receiver's identity $\mathtt{u}_r \in \mathcal{I}$, and time index $T \in \mathcal{T}$, $\mathcal{C}$ computes the sender's decryption key $dk_{\mathtt{u}_s,T} = \mathbf{FPKG}(mpk, \mathtt{u}_s, T, sk_{\mathtt{u}_s}, ku_T)$ (if necessary, first need to compute secret key $sk_{\mathtt{u}_s} = \mathbf{IPKG}(mpk, msk, \mathtt{u}_s, \mathtt{st})$ and key update $ku_T = \mathbf{KeyUp}(mpk, msk, T, RL, \mathtt{st})$), runs $\mathbf{SC}(mpk, \mathtt{u}_s, \mathtt{u}_r, T, dk_{\mathtt{u}_s,T}, M) \rightarrow \sigma$, and then returns the ciphertext σ.
- *Designcryption query.* Upon receiving this query for a ciphertext σ, a receiver's identity $\mathtt{u}_r \in \mathcal{I}$, a sender's identity $\mathtt{u}_s \in \mathcal{I}$, and time index $T \in \mathcal{T}$, $\mathcal{C}$ computes the receiver's decryption key $dk_{\mathtt{u}_r,T} = \mathbf{FPKG}(mpk, \mathtt{u}_r, T, sk_{\mathtt{u}_r}, ku_T)$ (if necessary, first need to compute secret key $sk_{\mathtt{u}_r} = \mathbf{IPKG}(mpk, msk, \mathtt{u}_r, \mathtt{st})$ and key update $ku_T = \mathbf{KeyUp}(mpk, msk, T, RL, \mathtt{st})$), runs $\mathbf{DSC}(mpk, \mathtt{u}_s, \mathtt{u}_r, T, dk_{\mathtt{u}_r,T}, \sigma)$, and returns its result to $\mathcal{A}$ (This result can be $\perp$ if σ is an invalid ciphertext).

– **Challenge.** At the end of Phase 1, $\mathcal{A}$ outputs two equal length plaintexts M_0^* and M_1^*, a time index T^*, and two identities $\mathtt{u}_s^*$ and $\mathtt{u}_r^*$, on which it wants to be challenged. $\mathcal{C}$ takes a random bit β from $\{0,1\}$ and runs signcryption algorithm on $(mpk, \mathtt{u}_s^*, \mathtt{u}_r^*, T^*, dk_{\mathtt{u}_s^*,T^*}, M_\beta^*)$ to obtain a ciphertext σ^* which is sent to $\mathcal{A}$.

– **Phase 2.** $\mathcal{A}$ can ask a polynomially bounded number of queries adaptively again as in Phase 1.

– **Guess.** $\mathcal{A}$ produces a bit β' and wins the IND-RIBSC-CCA2 game if $\beta' = \beta$ and the following restrictions are satisfied:

1) *Key update query* and *Revoke query* can be queried on time which is greater than or equal to the time of all previous queries.
2) *Revocation query* cannot be queried on time index T if *Key update query* was queried on T.
3) If *Initial private key generation query* was queried on the challenged identity $\mathtt{u}_r^*$, then *Revocation query* must be queried on $\mathtt{u}_r^*$ for $T \leq T^*$.
4) *Full private key generation query* cannot be queried on time index T before *Key update query* was queried on T.
5) *Full private generation query* cannot be queried on the challenged identity $\mathtt{u}_r^*$ and time index T^*.
6) (σ^*, T^*) was not returned by *Signcryption query* on input $(\mathtt{u}_s^*, \mathtt{u}_r^*, T^*, M_\beta^*)$ for $\beta \in \{0,1\}$.
7) *Designcryption query* cannot be queried on $(\mathtt{u}_s^*, \mathtt{u}_r^*, T^*, \sigma^*)$ to obtain the corresponding plaintext.

The advantage of $\mathcal{A}$ is defined as

$$Adv_{\mathcal{A}}^{IND-RIBSC-CCA2} = |2\Pr[\beta' = \beta] - 1|,$$

where $\Pr[\beta' = \beta]$ denotes the probability that $\beta' = \beta$.

Definition 3. *A RIBSC scheme is said to have the IND-RIBSC-CCA2 property if no polynomially bounded adversary has non-negligible advantage in the above IND-RIBSC-CCA2 game.*

For the *EUF-RIBSC-CMA* property, we consider the following game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

– **Initial.** The phase is the same one defined in the *IND-RIBSC-CCA2* game.

– **Queries.** $\mathcal{A}$ makes a polynomially bounded number of queries to the challenger $\mathcal{C}$. The queries are the same as ones defined in the *IND-RIBSC-CCA2* game.

– **Forge.** $\mathcal{A}$ outputs a new tuple $(\mathtt{u}_s^*, \mathtt{u}_r^*, T^*, \sigma^*)$, where T^* is a time index, $\mathtt{u}_s^*$ is a sender's identity, $\mathtt{u}_r^*$ is a receiver's identity, and σ^* is a ciphertext. We say that $\mathcal{A}$ wins the *EUF-RIBSC-CMA* game if the following restrictions are satisfied:

1) *Key update query* and *Revoke query* can be queried on time which is greater than or equal to the time of all previous queries.
2) *Revocation query* cannot be queried on time index T if *Key update query* was queried on T.
3) If *Initial private key generation query* was queried on the challenged identity $\mathtt{u}_s^*$, then *Revocation query* must be queried on $\mathtt{u}_s^*$ for $T \leq T^*$.
4) *Full private key generation query* cannot be queried on time index T before *Key update query* was queried on T.
5) *Full private key generation query* cannot be queried on the challenged identity $\mathtt{u}_s^*$ and time index T^*.
6) The new tuple $(\mathtt{u}_s^*, \mathtt{u}_r^*, T^*, \sigma^*)$ was not produced by *Signcryption query*.
7) The result of $\mathbf{DSC}(\mathtt{u}_s^*, \mathtt{u}_r^*, T^*, \sigma^*)$ is not the $\perp$ symbol.

The advantage of $\mathcal{A}$ is defined as the probability that it wins.

Definition 4. *A RIBSC scheme is said to have the EUF-RIBSC-CMA property if no polynomially bounded adversary has non-negligible advantage in the above EUF-RIBSC-CMA game.*

4 The Proposed Scheme

4.1 KUNode Algorithm

In the revocation process, we follow the KUNode algorithm and Boldyreva et al.'s idea to reduce the key update costs. In the actual schemes, the algorithm is used in a black-box manner.

Definition 5. *(KUNode Algorithm [2]). This algorithm takes as input a binary tree BT, revocation list RL, and time period index T, and outputs a set of nodes. A formal description of this algorithm is as follows: If η is a non-leaf node, then η_{left} and η_{right} denote the left and right child of η, respectively. Each user is assigned to a leaf node. If a user (assigned to η) is revoked on time index T, then $(\eta, T) \in RL$. Path(η) denotes the set of nodes on the path from η to root. The description of KUNode is given as follows.*

$\mathsf{KUNode}(\mathsf{BT}, RL, T)$:
$\mathsf{X}, \mathsf{Y} \leftarrow \emptyset$
$\forall(\eta_i, T_i) \in RL$
  If $T_i \leq T$ *then add* $\mathsf{Path}(\eta_i)$ *to* X
$\forall x \in \mathsf{X}$
  If $x_{left} \notin \mathsf{X}$ *then add* x_{left} *to* Y
  If $x_{right} \notin \mathsf{X}$ *then add* x_{right} *to* Y
If $\mathsf{Y} = \emptyset$ *then add* root *to* Y
Return Y.

This KUNode algorithm can be used to compute the minimal set of nodes for which key update needs to be published so that only non-revoked users at time index T are able to generate full private key. Please see a simple example in [21] to easily understand KUNode(BT,RL, T). When a user joins the system, the key authority assigns it to the leaf node η of a complete binary tree, and issues a set of keys, wherein each key is associated with each node on Path(η). At time index T, the key authority KGC publishes key updates for a set KUNode(BT, RL, T). Then, only non-revoked users have at least one key corresponding to a node in KUNode(BT, RL, T) and are able to generate decryption keys on time index T.

4.2 Our Construction

The new revocable identity-based signcryption can be described as the following algorithms.

- **Setup**(λ, N): On input (λ, N), the key authority does the followings:
  1) Generate two cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p with a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and g, g_2 the generators of $\mathbb{G}$;
  2) Choose a secret $\alpha \in \mathbb{Z}_p$, compute $g_1 = g^\alpha$ and pick up $u', m', v', v \in \mathbb{G}$ and two vectors $\overrightarrow{u} = (u_i), \overrightarrow{m} = (m_j)$ of length n_u and n_m respectively, where u_i, m_j are chosen from $\mathbb{G}$ randomly.
  3) Choose a collision-resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^{n_m}$;
  4) Set master public parameter $mpk = \{g, g_1, g_2, u', m', \overrightarrow{u}, \overrightarrow{m}, v', v\}$, master secret key $msk = \alpha$, $RL = \emptyset$, and st = BT, where BT is a binary tree with N leaves.
- **Initial Private Key Generation**$(mpk, msk, \mathtt{u}, \mathsf{st})$: Randomly choose an unassigned leaf η from BT, and store $\mathtt{u}$ in the node η. Let $\mathcal{U} \subset \{1, 2, \cdots, n_u\}$ be the set of indices such that $\mathtt{u}[i] = 1$, where $\mathtt{u}[i]$ is the i-th bit of $\mathtt{u}$. For each node $\theta \in \mathsf{Path}(\eta)$,
  1) Recall g_θ if it was defined. Otherwise, $g_\theta \xleftarrow{\$} \mathbb{G}$ and store $(g_\theta, \tilde{g}_\theta = g_2/g_\theta)$ in the node θ.
  2) Choose $r_\theta \xleftarrow{\$} \mathbb{Z}_p$.
  3) Compute $D_{\theta,0} \leftarrow g_\theta^\alpha (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, D_{\theta,1} \leftarrow g^{r_\theta}$.
  4) Output secret key $sk_{\mathtt{u}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathsf{Path}(\eta)}$.
- **Key Update Generation**$(mpk, msk, T, RL, \mathsf{st})$: Parse st = BT. For each node $\theta \in$ KUNode (BT, RL, T),
  1) Retrieve $\tilde{g}_\theta$ (note that $\tilde{g}_\theta$ is always pre-defined in the **Initial Private Key Generation** algorithm).
  2) Choose $s_\theta \xleftarrow{\$} \mathbb{Z}_p$.
  3) Compute $\tilde{D}_{\theta,0} \leftarrow \tilde{g}_\theta^\alpha (v' v^T)^{s_\theta}, \tilde{D}_{\theta,1} \leftarrow g^{s_\theta}$.
  4) Output key update $ku_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathsf{KUNode}(\mathrm{BT}, RL, T)}$.
- **Full Private Key Generation**$(mpk, sk_{\mathtt{u}}, ku_T)$: Parse $sk_{\mathtt{u}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathtt{I}}$ and $ku_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathtt{J}}$, where $\mathtt{I}$ denotes Path(η) and $\mathtt{J}$ denotes KUNode(BT, RL, T). If $\mathtt{I} \cap \mathtt{J} = \emptyset$, then return $\perp$. Otherwise, choose $\theta \in \mathtt{I} \cap \mathtt{J}$ and $r, s \xleftarrow{\$} \mathbb{Z}_p$ and return full private/decryption key

$$dk_{\mathtt{u},T} = (D_{\theta,0}\tilde{D}_{\theta,0}(u' \prod_{i \in \mathcal{U}} u_i)^r (v' v^T)^s, D_{\theta,1} g^r, \tilde{D}_{\theta,1} g^s)$$
$$= (g_2^\alpha (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta + r} (v' v^T)^{s_\theta + s}, g^{r_\theta + r}, g^{s_\theta + s}).$$

Let $\mathtt{u}_A$ be sender Alice's identity and $\mathtt{u}_B$ receiver Bob's identity. Then the full private key of Alice at some time period index T is

$$dk_{\mathtt{u}_A,T} = (g_2^\alpha (u' \prod_{i \in \mathcal{U}_A} u_i)^{r_{\theta_A} + r_A} (v' v^T)^{s_{\theta_A} + s_A},$$
$$g^{r_{\theta_A} + r_A}, g^{s_{\theta_A} + s_A}).$$

And the full private key of Bob at some time period index T is

$$dk_{\mathfrak{u}_B,T} = (g_2^{\alpha}(u' \prod_{i\in\mathcal{U}_B} u_i)^{r_{\theta_B}+r_B}(v'v^T)^{s_{\theta_B}+s_B}, g^{r_{\theta_B}+r_B}, g^{s_{\theta_B}+s_B}).$$

- **Signcryption**$(mpk, \mathfrak{u}_A, \mathfrak{u}_B, T, dk_{\mathfrak{u}_A,T}, M)$: On input $M \in \mathbb{G}_T$, the receiver Bob's identity $\mathfrak{u}_B$, the sender Alice's identity $\mathfrak{u}_A$ and full private key $dk_{\mathfrak{u}_A,T} = (dk_{\mathfrak{u}_A,T,1}, dk_{\mathfrak{u}_A,T,2}, dk_{\mathfrak{u}_A,T,3})$, and the current time index T, the algorithm does the following:

  1) Randomly choose a random integer $k \in \mathbb{Z}_p$.
  2) Compute $\sigma_0 = M \cdot e(g_1, g_2)^k$, $\sigma_1 = g^{-k}$, $\sigma_2 = (u' \prod_{i\in\mathcal{U}_B} u_i)^k$, $\sigma_3 = (v'v^T)^k$, $\sigma_4 = dk_{\mathfrak{u}_A,T,2}$, and $\sigma_5 = dk_{\mathfrak{u}_A,T,3}$.
  3) Compute $\mathfrak{m} = H_1(\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \mathfrak{u}_A, \mathfrak{u}_B)$, and let $\mathcal{M} \subset \{1, \cdots, n_m\}$ be the set of indices j such that $\mathfrak{m}[j] = 1$.
  4) Compute $\sigma_6 = dk_{\mathfrak{u}_A,T,1} \cdot (m' \prod_{j\in\mathcal{M}} m_j)^k$.
  5) Output $\sigma = (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6)$.

- **Designcryption**$(mpk, \mathfrak{u}_A, \mathfrak{u}_B, T, dk_{\mathfrak{u}_B,T}, \sigma)$: On input $\sigma = (\sigma_0, \cdots, \sigma_6)$, the time index T, the receiver's full private key $dk_{\mathfrak{u}_B,T} = (dk_{\mathfrak{u}_B,T,1}, dk_{\mathfrak{u}_B,T,2}, dk_{\mathfrak{u}_B,T,3})$ and the sender's identity $\mathfrak{u}_A$, the algorithm outputs M, or $\perp$ (if the signcryptext is not valid) as follows:

  1) Compute $\mathfrak{m} = H_1(\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \mathfrak{u}_A, \mathfrak{u}_B)$, and let $\mathcal{M} \subset \{1, \cdots, n_m\}$ be the set of indices j such that $\mathfrak{m}[j] = 1$.
  2) Check if the following equation holds:

$$e(\sigma_6, g) \stackrel{?}{=} e(g_1, g_2)e(u' \prod_{i\in\mathcal{U}_A} u_i, \sigma_4)e(v'v^T, \sigma_5) \cdot e(m' \prod_{j\in\mathcal{M}} m_j, \sigma_1^{-1}). \quad (1)$$

  if Equation (1) holds, output

$$M = \sigma_0 \cdot \prod_{i=1}^{3} e(dk_{\mathfrak{u}_B,T,i}, \sigma_i). \quad (2)$$

- **Revocation**$(mpk, \mathfrak{u}, T, RL, \mathsf{st})$: Let η be the leaf node associated with $\mathfrak{u}$. Update the revocation list by $RL \leftarrow RL \cup \{\eta, T\}$ and return the updated revocation list.

5 Security Analysis

5.1 Consistency

Now we verify the consistency of our scheme. For Equation (1), we have

$$\begin{aligned}
&e(\sigma_6, g)\\
=&e(dk_{\mathfrak{u}_A,T,1}(m' \prod_{j\in\mathcal{M}} m_j)^k, g)\\
=&e(g_2^{\alpha}(u' \prod_{i\in\mathcal{U}_A} u_i)^{r_{\theta_A}+r_A}(v'v^T)^{s_{\theta_A}+s_A}(m' \prod_{j\in\mathcal{M}} m_j)^k, g)\\
=&e(g_2^{\alpha}, g)e((u' \prod_{i\in\mathcal{U}_A} u_i)^{r_{\theta_A}+r_A}, g)e((v'v^T)^{s_{\theta_A}+s_A}, g)\\
&\cdot e((m' \prod_{j\in\mathcal{M}} m_j)^k, g)\\
=&e(g_1, g_2)e(u' \prod_{i\in\mathcal{U}_A} u_i, \sigma_4)e(v'v^T, \sigma_5)e(m' \prod_{j\in\mathcal{M}} m_j, \sigma_1^{-1}).
\end{aligned}$$

For Equation (2), we have

$$\begin{aligned}
&\sigma_0 \prod_{i=1}^{3} e(dk_{\mathfrak{u}_B,T,i}, \sigma_i)\\
=&Me(g_1, g_2)^k \frac{e(g^{r_{\theta_B}+r_B}, (u' \prod_{i\in\mathcal{U}_B} u_i)^k)e(g^{s_{\theta_B}+s_B}, (v'v^T)^k)}{e(g_2^{\alpha}(u' \prod_{i\in\mathcal{U}_B} u_i)^{r_{\theta_B}+r_B}(v'v^T)^{s_{\theta_B}+s_B}, g^k)}\\
=&\frac{Me(g_1, g_2)^k \cdot e(g^{r_{\theta_B}+r_B}, (u' \prod_{i\in\mathcal{U}_B} u_i)^k)e(g^{s_{\theta_B}+s_B}, (v'v^T)^k)}{e(g_2^{\alpha}, g^k)e((u' \prod_{i\in\mathcal{U}_B} u_i)^{r_{\theta_B}+r_B}, g^k)e((v'v^T)^{s_{\theta_B}+s_B}, g^k)}\\
=&\frac{Me(g_1, g_2)^k \cdot e(g^{r_{\theta_B}+r_B}, (u' \prod_{i\in\mathcal{U}_B} u_i)^k)e(g^{s_{\theta_B}+s_B}, (v'v^T)^k)}{e(g_2, g_1)^k e((u' \prod_{i\in\mathcal{U}_B} u_i)^k, g^{r_{\theta_B}+r_B})e((v'v^T)^k, g^{s_{\theta_B}+s_B})}\\
=&M.
\end{aligned}$$

5.2 Security

Next, we reduce the IND-RIBSC-CCA2 property to the DBDH hardness assumption and the EUF-RIBSC-CMA property to the CDH hardness assumption.

Theorem 1. *If there exists an adversary $\mathcal{A}$ attacking IND-RIBSC-CCA security of the proposed RIBSC scheme, then there exists a challenger $\mathcal{C}$ breaking a DBDH problem instance.*

Proof. We suppose that an $(\epsilon, t, q_{ipk}, q_{ku}, q_{fpk}, q_r, q_s, q_d)$ adversary $\mathcal{A}$ for our scheme exists, where it has advantage at least ϵ, runs in time at most t, and makes at most q_{ipk} initial private key queries, q_{ku} key update queries, q_{fpk} full private key queries, q_r revocation queries, q_s signcryption queries, and q_d designcryption queries. From the adversary, we construct a simulator $\mathcal{C}$, which makes use of $\mathcal{A}$ to solve DBDH game with a probability at least ϵ' and in time at most t', contradicting the (ϵ', t')-DBDH

assumption. Our approach is based on Waters' idea such as [16, 17, 20, 21, 30].

$\mathcal{C}$ will take DBDH challenge $(g, A = g^a, B = g^b, C = g^c, Z)$ and output a guess, β', as to whether the challenge is a DBDH tuple. In order to use $\mathcal{A}$ to solve the problem, $\mathcal{C}$ needs to simulate a challenger and all queries for $\mathcal{A}$. $\mathcal{C}$ then simulates the queries of $\mathcal{A}$ as follows.

Setup: $\mathcal{C}$ randomly guesses the challenge time $T^* \in \mathcal{T}$. We assume that $\mathcal{C}$'s guess is right. (It holds with $1/|\mathcal{T}|$ and this is a loss of polynomial in λ.) Let $l_u = 2(q_{ipk} + q_{fpk} + q_s + q_d)$ and $l_m = 2(q_s + q_d)$.

1) $\mathcal{C}$ randomly chooses two integers k_u and k_m ($0 \leq k_u \leq n_u, 0 \leq k_m \leq n_m$). We assume that $l_u(n_u + 1) < p$ and $l_m(n_m + 1) < p$ for the given values of $q_{ipk}, q_{fpk}, q_s, q_d, n_u$ and n_m.

2) $\mathcal{C}$ picks an integer $x' \in \mathbb{Z}_{l_u}$ and a vector $\mathbf{X} = (x_i)_{n_u} (x_i \in \mathbb{Z}_{l_u})$ at random.

3) $\mathcal{C}$ randomly selects an integer $z' \in \mathbb{Z}_{l_m}$ and a vector $\mathbf{Z} = (z_j)_{n_m} (z_j \in \mathbb{Z}_{l_m})$.

4) $\mathcal{C}$ randomly picks two integers $y', w' \in \mathbb{Z}_p$ and two vectors $\mathbf{Y} = (y_i)_{n_u} (y_i \in \mathbb{Z}_p)$ and $\mathbf{W} = (w_j)_{n_m} (w_j \in \mathbb{Z}_p)$.

5) $\mathcal{C}$ randomly chooses $\nu, \nu' \in \mathbb{Z}_p$.

For convenience, we define the two pairs of functions for binary identity string $\mathfrak{u}$ and message string $\mathfrak{m}$ as follows:

$$
\begin{aligned}
F(\mathfrak{u}) &= (p - l_u k_u) + x' + \sum_{i \in \mathcal{U}} x_i, \\
J(\mathfrak{u}) &= y' + \sum_{i \in \mathcal{U}} y_i, \\
K(\mathfrak{m}) &= (p - l_m k_m) + z' + \sum_{j \in \mathcal{M}} z_j \\
L(\mathfrak{m}) &= w' + \sum_{j \in \mathcal{M}} w_j,
\end{aligned}
$$

where $\mathcal{U} \subset \{1, \cdots, n_u\}$ denotes the set of indices i such that $\mathfrak{u}[i] = 1$ and $\mathcal{M} \subset \{1, \cdots, n_m\}$ denotes the set of indices j such that $\mathfrak{m}[j] = 1$. Then the challenger assigns a set of public parameters as follows:

$$
\begin{aligned}
&g_1 = g^a, && g_2 = g^b, \\
&u' = g_2^{(p - l_u k_u) + x'} g^{y'}, && u_i = g_2^{x_i} g^{y_i} (1 \leq i \leq n_u), \\
&m' = g_2^{(p - l_m k_m) + z'} g^{w'}, && m_j = g_2^{z_j} g^{w_j} (1 \leq j \leq n_m), \\
&v' = g_1^{-T^*} \cdot g^{\nu'}, && v = g_1 \cdot g^{\nu}.
\end{aligned}
$$

Note that the master secret key is $g_2^{\alpha} = g_1^b = g^{ab}$ and the following equations hold for an identity $\mathfrak{u}$ and a message $\mathfrak{m}$:

$$
u' \prod_{i \in \mathcal{U}} u_i = g_2^{F(\mathfrak{u})} g^{J(\mathfrak{u})}, \qquad m' \prod_{j \in \mathcal{M}} m_j = g_2^{K(\mathfrak{m})} g^{L(\mathfrak{m})}.
$$

Then, it publishes $mpk = \{g, g_1, g_2, u', \overrightarrow{u} = (u_i), m', \overrightarrow{m} = (m_j), v', v\}$. The corresponding master secret key is g_2^{α}. Although $\mathcal{C}$ does not know the master secret key, it still can construct a private key (d_0, d_1) for an identity $\mathfrak{u}$ by assuming $F(\mathfrak{u}) \neq 0 \mod p$, which is the private key generation oracle $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ of the Waters IBE scheme [27]. $\mathcal{C}$ randomly chooses $r_u \in \mathbb{Z}_p$ and computes:

$$
(d_0, d_1) = (g_1^{-J(\mathfrak{u})/F(\mathfrak{u})} (u' \prod_{i \in \mathcal{U}} u_i)^{r_u}, g_1^{-1/F(\mathfrak{u})} g^{r_u}).
$$

By writing $\hat{r}_u = r_u - a/F(\mathfrak{u})$, we can show that (d_0, d_1) is a valid private key for the identity $\mathfrak{u}$ as follows. The challenger $\mathcal{C}$ can generate such a private key (d_0, d_1) if and only if $F(\mathfrak{u}) \neq 0 \mod l_u$, which suffices to have $F(\mathfrak{u}) \neq 0 \mod p$. The simulation is perfect since

$$
\begin{aligned}
d_0 &= g_1^{-J(\mathfrak{u})/F(\mathfrak{u})} (u' \prod_{i \in \mathcal{U}} u_i)^{r_u} \\
&= g_2^a (g_2^{F(\mathfrak{u})} g^{J(\mathfrak{u})})^{-a/F(\mathfrak{u})} (g_2^{F(\mathfrak{u})} g^{J(\mathfrak{u})})^{r_u} \\
&= g_2^a (g_2^{F(\mathfrak{u})} g^{J(\mathfrak{u})})^{r_u - a/F(\mathfrak{u})} \\
&= g_2^a (u' \prod_{i \in \mathcal{U}} u_i)^{\hat{r}_u},
\end{aligned}
$$

and $d_1 = g_1^{-1/F(\mathfrak{u})} g^{r_u} = g^{r_u - a/F(\mathfrak{u})} = g^{\hat{r}_u}$. If, on the other hand, $F(\mathfrak{u}) = 0 \mod p$, $\mathcal{C}$ aborts.

Let $\mathfrak{u}^*$ be the challenge identity. $\mathcal{C}$ guesses an adversarial type among the following two types:

1. Type-1 adversary: $\mathcal{A}$ issues an initial private key generation query for $sk_{\mathfrak{u}^*}$, and so $\mathfrak{u}^*$ should be revoked before T^*. (For $T \neq T^*$, $\mathcal{A}$ may query $dk_{\mathfrak{u}^*,T}$.)

2. Type-2 adversary: $\mathcal{A}$ does not query $sk_{\mathfrak{u}^*}$, but $\mathcal{A}$ may issue $dk_{\mathfrak{u}^*,T}$ for $T \neq T^*$.

We assume that $\mathcal{C}$'s guess is right. (It holds with $1/2$ probability.) We separately describe $\mathcal{C}$'s other process according to its guess.

Type-1 Adversary. Let q be the maximum number of queries regarding initial private key generation queries, full private key generation queries, signcryption queries or designcryption queries. $\mathcal{C}$ randomly guesses $i^* \in [1, q]$ such that $\mathcal{A}$'s i^*-th query is the first query regarding $\mathfrak{u}^*$ among initial private key generation queries, full private key generation queries, signcryption queries and designcryption queries. We assume that $\mathcal{C}$'s guess is right. (It holds with $1/q$ and this is a loss of polynomial in λ.) $\mathcal{C}$ randomly choose a leaf node η^* that will be used for $\mathfrak{u}^*$ (this is not a security loss, but just a pre-assignment for $\mathfrak{u}^*$.) $\mathcal{C}$ marks η^* as a defined node. $\mathcal{C}$ keeps an integer count to count the number of queries for initial private key generation, full private key generation, signcryption or designcryption up to the current time.

Key Update Queries: For all nodes $\theta \in \mathsf{KUNode}(\mathsf{BT}, RL, T)$, $\mathcal{C}$ recalls S_θ from the node θ if it is defined. Otherwise, $\mathcal{C}$ chooses $S_\theta \xleftarrow{\$} \mathbb{G}$ and stores it in the node θ. $\mathcal{C}$ computes $\tilde{D}_{\theta,0}$ and $\tilde{D}_{\theta,1}$ as follows: if $\theta \notin \mathsf{Path}(\eta^*)$, then

$$
(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1}) = (S_\theta^{-1} (v' v^T)^{s_\theta}, g^{s_\theta}),
$$

otherwise

$$(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})$$
$$=(S_\theta^{-1} g_2^{-\frac{\nu'+\nu T^*}{T-T^*}} g_1^{s_\theta(T-T^*)} g^{s_\theta(\nu'+\nu T^*)}, g_2^{-\frac{1}{T-T^*}} g^{s_\theta}),$$

where $s_\theta \xleftarrow{\$} \mathbb{Z}_p$. In fact, if $\theta \in \mathsf{Path}(\eta^*)$, then

$$(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})$$
$$=(S_\theta^{-1} g_2^{-\frac{\nu'+\nu T^*}{T-T^*}} g_1^{s_\theta(T-T^*)} g^{s_\theta(\nu'+\nu T^*)}, g_2^{-\frac{1}{T-T^*}} g^{s_\theta})$$
$$=(S_\theta^{-1} g_2^a (g_1^{T-T^*} g^{\nu'+\nu T^*})^{-\frac{b}{T-T^*}+s_\theta}, g^{-\frac{b}{T-T^*}+s_\theta})$$
$$=(S_\theta^{-1} g_2^a (v'v^T)^{s'_\theta}, g^{s'_\theta})$$

where $s'_\theta = -\frac{b}{T-T^*} + s_\theta$. Output

$$ku_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T)}.$$

When $T = T^*$, $\mathfrak{u}^*$ should be in the revocation list RL so that $\mathcal{C}$ performs the above computation for only $\theta \notin \mathsf{Path}(\eta^*)$.

Revocation Queries: Upon receiving this query on $(\mathfrak{u}, T)$, $\mathcal{C}$ runs algorithm $\mathbf{REV}(mpk, \mathfrak{u}, T, RL, \mathsf{st}) \rightarrow RL$ and returns the updated revocation list RL.

From now, we explain how $\mathcal{C}$ responds to initial private key generation queries, full private key generation queries, signcryption queries and designcryption queries according to count.

Case $\mathsf{count} < i^*$: Whenever $\mathcal{C}$ receives either initial private key generation query for $\mathfrak{u}$, full private key generation query for $(\mathfrak{u}, T)$, signcryption query for $(\mathfrak{u}, \mathfrak{u}_r, M, T)$, or designcryption query for $(\mathfrak{u}_s, \mathfrak{u}, \sigma, T)$, $\mathcal{C}$ firstly sends $\mathfrak{u}$ to $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle and obtains (d_0, d_1), and then randomly chooses an undefined leaf node η and store $\mathfrak{u}$ in η.

- **Initial Private Key Generation Queries**: For $\theta \in \mathsf{Path}(\eta^*)$, $\mathcal{C}$ recalls S_θ if it is defined. Otherwise, $S_\theta \xleftarrow{\$} \mathbb{G}$ and store it in the node θ. Compute

$$(D_{\theta,0}, D_{\theta,1})$$
$$= \begin{cases} (S_\theta (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, g^{r_\theta}), \text{if } \theta \in \mathsf{Path}(\eta^*), \\ (S_\theta d_0 (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, d_1 g^{r_\theta}), \text{otherwise}, \end{cases}$$

  where $r_\theta \xleftarrow{\$} \mathbb{Z}_p$. Return the secret key $sk_{\mathfrak{u}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_\theta$.
- **Full Private Key Generation Queries**: Run key update query and initial private key generation query, and then run full private key generation algorithm as follows (Since $\mathsf{count} < i^*$, $\mathfrak{u} \neq \mathfrak{u}^*$ holds. So, $\mathcal{C}$ can query $\mathfrak{u}$ to $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle).

1) For the case of $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T) \cap \neg\, \mathsf{Path}(\eta^*)$, $\mathcal{C}$ runs initial private key generation query and key update query to obtain secret key

$$sk_{\mathfrak{u}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_\theta$$
$$= \{(\theta, S_\theta \cdot d_0 \cdot (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, d_1 \cdot g^{r_\theta})\}_\theta,$$

and update key

$$ku_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_\theta$$
$$= \{(\theta, S_\theta^{-1} (v'v^T)^{s_\theta}, g^{s_\theta})\}_\theta.$$

If $\mathsf{KUNode}(\mathtt{BT}, RL, T) \cap \neg\mathsf{Path}(\eta^*) = \emptyset$, then return $\perp$. Otherwise, choose $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T) \cap \neg\mathsf{Path}(\eta^*)$ and $r, s \xleftarrow{\$} \mathbb{Z}_p$ and return the decryption key

$$dk_{\mathfrak{u},T} = (D_{\theta,0} \cdot \tilde{D}_{\theta,0} \cdot (u' \prod_{i \in \mathcal{U}} u_i)^r (v'v^T)^s,$$
$$D_{\theta,1} \cdot g^r, \tilde{D}_{\theta,1} \cdot g^s)$$
$$= (g_2^a (u' \prod_{i \in \mathcal{U}} u_i)^{\hat{r}_u + r_\theta + r} (v'v^T)^{s_\theta + s},$$
$$g^{\hat{r}_u + r_\theta + r}, g^{s_\theta + s}).$$

2) For the case of $\theta \in \mathsf{KUNode}(\mathtt{BT}, RL, T) \cap \mathsf{Path}(\eta^*)$, then $\mathcal{C}$ does the similar process as above and returns the decryption key

$$dk_{\mathfrak{u},T} = (g_2^a (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta + r} (v'v^T)^{s'_\theta + s},$$
$$g^{r_\theta + r}, g^{s'_\theta + s}).$$

- **Signcryption Queries**: When $\mathcal{A}$ queries the signcrypt oracle for a message M, a time index T, a sender's identity $\mathfrak{u}$ and a receiver's identity $\mathfrak{u}_r$, the challenger $\mathcal{C}$ proceeds as follows:
  1. Computes a decryption key $dk_{\mathfrak{u},T}$ by running a full private key generation query for $\mathfrak{u}$ and T (If it is necessary to query $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle on $\mathfrak{u}$, but $F(\mathfrak{u}) = 0 \mod l_u$, $\mathcal{C}$ will simply abort).
  2. Run the algorithm $\mathbf{SC}(\mathfrak{u}, \mathfrak{u}_r, T, M)$ and return its output as response.
- **Designcryption Queries**: At any time $\mathcal{A}$ can perform a designcryption query for a ciphertext $\sigma = (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6)$ associated with T, $\mathfrak{u}_s$ and $\mathfrak{u}$, $\mathcal{C}$ does the following.
  1. Compute

$$\mathfrak{m} = H_1(\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \mathfrak{u}_s, \mathfrak{u}),$$

2. Set $\mathcal{M} \subset \{1, \cdots, n_m\}$ be the set of indices j such that $\mathfrak{m}[j] = 1$, where $\mathfrak{m}[j]$ is the j-th bit of $\mathfrak{m}$,
3. Check the equation

$$e(\sigma_6, g) \stackrel{?}{=} e(v'v^T, \sigma_5) e(m' \prod_{j \in \mathcal{M}} m_j, \sigma_1^{-1}) \cdot e(g_1, g_2) e(u' \prod_{i \in \mathcal{U}} u_i, \sigma_4). \quad (3)$$

4. Prepare its response according to the following situations.

(i) If Equation (3) does not hold, $\mathcal{C}$ rejects the ciphertext.

(ii) If Equation (3) holds and $F(\mathfrak{u}) = 0 \mod l_u$, but it is necessary to query $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle on $\mathfrak{u}$, then $\mathcal{C}$ will abort.

(iii) If Equation (3) holds and $F(\mathfrak{u}) \neq 0 \mod l_u$, or $F(\mathfrak{u}) = 0 \mod l_u$, but it is not necessary to query $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle on $\mathfrak{u}$, then $\mathcal{C}$ makes a full private key generation query on $\mathfrak{u}$ and T, and obtains the decryption key $dk_{\mathfrak{u},T} = (dk_{\mathfrak{u},T,1}, dk_{\mathfrak{u},T,2}, dk_{\mathfrak{u},T,3})$ and returns the message

$$M = \sigma_0 \cdot \prod_{i=1}^{3} e(dk_{\mathfrak{u},T,i}, \sigma_i).$$

Case $\mathsf{count}{=}i^*$: $\mathcal{C}$ can identify $\mathfrak{u}^*$ and store $\mathfrak{u}^*$ in the pre-assigned leaf node η^*.

- **Initial Private Key Generation Queries**: For $\theta \in \mathsf{Path}(\eta^*)$, $\mathcal{C}$ recalls S_θ if it is defined. Otherwise, $S_\theta \stackrel{\$}{\leftarrow} \mathbb{G}$ and store it in the node θ. Return

$$(S_\theta \cdot (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, g^{r_\theta}), \text{where } r_\theta \stackrel{\$}{\leftarrow} \mathbb{Z}_p.$$

Note that it is not necessary to obtain (d_0, d_1) by sending $\mathfrak{u}$ to $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle. Thus in this case we do not need to consider whether $F(\mathfrak{u}) = 0 \mod l_u$ or not.

- **Full Private Key Generation Queries**: Run initial private key generation query for $\mathfrak{u}$ and key update query on T, and then run full private key generation algorithm for $\mathfrak{u}$ and T.
- **Signcryption Queries**: When $\mathcal{A}$ queries the signcrypt oracle for a message M, a time index T, a sender's identity $\mathfrak{u}$ and a receiver's identity $\mathfrak{u}_r$, the challenger $\mathcal{C}$ will proceeds the same as signcryption query in the case $\mathsf{count} < i^*$.
- **Designcryption Queries**: At any time $\mathcal{A}$ can perform a designcryption query for a ciphertext $\sigma = (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6)$ associated with T, $\mathfrak{u}_s$ and $\mathfrak{u}$, $\mathcal{C}$ does the same as the designcryption query in the case $\mathsf{count} < i^*$.

Case $\mathsf{count} > i^*$: If $\mathfrak{u} \neq \mathfrak{u}^*$, then $\mathcal{C}$ does the same process as the queries in the case $\mathsf{count} < i^*$. Otherwise, $\mathcal{C}$ acts the same as the queries in the case $\mathsf{count} = i^*$.

Challenge: At the end of the first stage, $\mathcal{A}$ outputs two messages M_0^* and M_1^*, a time period index T^*, a receiver's identity $\mathfrak{u}_r^*$, and a sender's identity $\mathfrak{u}^*$ on which it wishes to be challenged. Then, $\mathcal{C}$ chooses a random bit $\beta \in \{0, 1\}$ and fails if $F(\mathfrak{u}^*) = 0 \mod l_u$, or $F(\mathfrak{u}_r^*) \neq 0 \mod l_u$, or $K(\mathfrak{m}^*) \neq 0 \mod l_m$. Otherwise, $\mathcal{C}$ first makes the full private key generation query on $(\mathfrak{u}^*, T^*)$ and obtains the decryption key $dk_{\mathfrak{u}^*,T^*} = (dk_{\mathfrak{u}^*,T^*,1}, dk_{\mathfrak{u}^*,T^*,2}, dk_{\mathfrak{u}^*,T^*,3})$, then sets

$$\begin{array}{ll} \sigma_0^* = M_\beta^* \cdot Z, & \sigma_1^* = C^{-1}, \\ \sigma_2^* = C^{J(\mathfrak{u}_r^*)}, & \sigma_3^* = C^{\nu' + \nu \cdot T^*}, \\ \sigma_4^* = dk_{\mathfrak{u}^*,T^*,2}, & \sigma_5^* = dk_{\mathfrak{u}^*,T^*,3}, \\ \mathfrak{m}^* = H(\sigma_0^*, \cdots, \sigma_5^*, \mathfrak{u}^*, \mathfrak{u}_r^*), & \sigma_6^* = dk_{\mathfrak{u}^*,T^*,1} \cdot C^{L(\mathfrak{m}^*)}. \end{array}$$

Finally, $\mathcal{C}$ sends $\sigma^* = (\sigma_0^*, \cdots, \sigma_6^*)$ to $\mathcal{A}$. It is obvious that along with the assumption that $\mathcal{C}$ does not fail, the signcryptext σ^* can pass the verification equation in the designcryption algorithm.

During the second phase, $\mathcal{A}$ may continue to make the queries to the challenger $\mathcal{C}$ as above, but with the restrictions in Subsection 3.2.

Eventually, $\mathcal{A}$ outputs a bit β'. If $\beta = \beta'$, then $\mathcal{C}$ output 1 (which means that $e(g,g)^{abc} = e(g,g)^z$), and 0 otherwise (which means that $e(g,g)^{abc} \neq e(g,g)^z$).

Type-2 Adversary: Let q be the maximum number of full private key generation queries, signcryption queries or designcryption queries. $\mathcal{C}$ randomly guesses $i^* \in [1, q]$ such that $\mathcal{A}$'s i^*-th query is the first query regarding $\mathfrak{u}^*$ among full private key generation queries, signcryption queries and designcryption queries. We assume that $\mathcal{C}$'s guess is right (It holds with $1/q$ and this is a loss of polynomial in λ). $\mathcal{C}$ keeps an integer count to count the number of full private key generation queries, signcrypiton queries or designcryption queries up to the current time.

Key Update Queries: For all nodes $\theta \in \mathsf{KUNode}(\mathsf{BT}, RL, T)$, $\mathcal{C}$ recalls S_θ from the node θ if it is defined. Otherwise, $\mathcal{C}$ chooses $S_\theta \stackrel{\$}{\leftarrow} \mathbb{G}$ and stores it in the node θ. $\mathcal{C}$ computes

$$(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1}) = (S_\theta^{-1} (v'v^T)^{s_\theta}, g^{s_\theta}),$$

where $s_\theta \stackrel{\$}{\leftarrow} \mathbb{Z}_p$. Output $ku_T = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathsf{KUNode}(\mathsf{BT}, RL, T)}$.

Revocation Queries: Upon receiving this query on $(\mathfrak{u}, T)$, $\mathcal{C}$ runs algorithm $\mathbf{REV}(mpk, \mathfrak{u}, T, RL, \mathsf{st}) \rightarrow RL$ and returns the updated revocation list RL.

Initial Private Key Generation Queries: $\mathcal{C}$ starts with receiving an identity $\mathfrak{u}$, sends it to $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$, and obtains (d_0, d_1) (if $F(\mathfrak{u}) = 0 \mod l_u$, then abort). $\mathcal{C}$ randomly chooses an undefined leaf node η and stores $\mathfrak{u}$ in η. For $\theta \in \mathsf{Path}(\eta)$, $\mathcal{C}$ recalls S_θ if it is defined. Otherwise, $S_\theta \xleftarrow{\$} \mathbb{G}$ and store it in the node θ. Then return the secret key

$$sk_{\mathfrak{u}} = \{(\theta, S_\theta \cdot d_0 \cdot (u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta}, d_1 \cdot g^{r_\theta})\}_{\theta \in \mathsf{Path}(\eta)},$$

where r_θ is randomly chosen from $\mathbb{Z}_p$.

Full Private Key Generation Queries: For $\mathfrak{u} \neq \mathfrak{u}^*$ and all T, run initial private key generation query and key update query, and full private key generation algorithm (When $\mathsf{count} < i^*$, all $\mathfrak{u}$ are not equal to $\mathfrak{u}^*$. And when $\mathsf{count} = i^*$, $\mathcal{C}$ can identify $\mathfrak{u}^*$). If $\mathsf{KUNode}(\mathsf{BT}, RL, T) \cap \mathsf{Path}(\eta) = \emptyset$, then return $\perp$. Otherwise, choose $\theta \in \mathsf{KUNode}(\mathsf{BT}, RL, T) \cap \mathsf{Path}(\eta)$ and $r, s \xleftarrow{\$} \mathbb{Z}_p$ and return the decryption key

$$dk_{\mathfrak{u},T} = (d_0(u' \prod_{i \in \mathcal{U}} u_i)^{r_\theta + r}(v'v^T)^{s_\theta + s}, d_1 g^{r_\theta + r}, g^{s_\theta + s}).$$

For $\mathfrak{u} = \mathfrak{u}^*$ and $T \neq T^*$, $\mathcal{C}$ chooses random integers $r, s \xleftarrow{\$} \mathbb{Z}_p$ and outputs the decryption key

$$\begin{aligned} dk_{\mathfrak{u}^*,T} =& ((u' \prod_{i \in \mathcal{U}^*} u_i)^r g_2^{-\frac{\nu' + \nu T}{T - T^*}} (v'v^T)^s, g^r, g_2^{-\frac{1}{T - T^*}} g^s) \\ =& (g_2^a (u' \prod_{i \in \mathcal{U}^*} u_i)^r (v'v^T)^{s'}, g^r, g^{s'}), \end{aligned}$$

where $s' = -\frac{b}{T - T^*} + s$. Thus the decryption keys for $\mathfrak{u} = \mathfrak{u}^*$ and $T \neq T^*$ are identically distributed to those generated in the real experiment.

Signcryption Queries: When $\mathcal{A}$ queries the signcryption oracle for a message M, a time index T, a sender's identity $\mathfrak{u}$ and a receiver's identity $\mathfrak{u}_r$, the challenger $\mathcal{C}$ proceeds the same as signcryption oracle for Type-1 adversary.

Designcryption Queries: At any time $\mathcal{A}$ can perform a designcryption query for a ciphertext $\sigma = (\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6)$ associated with T, $\mathfrak{u}_s$ and $\mathfrak{u}$, $\mathcal{C}$ does the same as designcryption query for Type-1 adversary.

Challenge: $\mathcal{C}$ acts the same as the **Challenge** for Type-1 adversary.

Note that $\mathcal{C}$ does not query $\mathfrak{u}^*$ to $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle during the simulation. Type-2 adversary does not query the initial private key generation for $\mathfrak{u}^*$, but she may query full private key generation for $\mathfrak{u}^*$ and $T \neq T^*$. For full private key generation query for the challenged identity $\mathfrak{u}^*$ and time index $T \neq T^*$, $\mathcal{C}$ simulates queries without aid of $\mathrm{PKG}_{\mathrm{Wat}}(\cdot)$ oracle. The analysis about the challenge phase is same as the case for Type-1 adversary.

This completes the description of the simulation. It remains to analyze $\mathcal{C}$'s advantage. According to Claims 1 and 2 in [21], the distribution of all transcription between a challenger $\mathcal{C}$ and two types of adversaries $\mathcal{A}$ is identical to the real experiment. Furthermore, if $\mathcal{C}$ correctly guesses and does not abort, $\mathcal{C}$'s advantage is equal to $\mathcal{A}$'s advantage.

In the following, we firstly compute the probability of $\mathcal{C}$'s correct guess. In the setup phase, $\mathcal{C}$ randomly guesses the challenged time $T^* \in \mathcal{T}$, and so $\mathcal{C}$'s correct guess holds with $1/|\mathcal{T}|$. In the queries, $\mathcal{C}$ randomly guesses $i^* \in [1, q]$ such that $\mathcal{A}$'s i^*-th query is the first query regarding $\mathfrak{u}^*$ among initial private key generation queries, full private key generation queries, signcryption queries and designcryption queries, and so $\mathcal{C}$'s correct guess holds with $1/q$, where q is the maximum number of queries. It is obvious that $\mathcal{C}$'s guess T^* is totally independent from its guess i^*.

Then we consider the probability of $\mathcal{C}$'s not aborting. For the simulation to complete without aborting, we require that at most all initial private key generation queries, full private key generation queries on an identity $\mathfrak{u}$ have $F(\mathfrak{u}) \neq 0 \mod l_u$, that at most all signcryption queries $(\mathfrak{u}, \mathfrak{u}_r, M, T)$ have $F(\mathfrak{u}) \neq 0 \mod l_u$, that at most all designcryption queries $(\mathfrak{u}_s, \mathfrak{u}, \sigma, T)$ have $F(\mathfrak{u}) \neq 0 \mod l_u$ and that $F(\mathfrak{u}^*) \neq 0 \mod l_u$, $F(\mathfrak{u}_r^*) = 0 \mod l_u$ and $K(\mathfrak{m}^*) = 0 \mod l_m$. Similarly to the same technique in [16, 17, 20, 21, 30], we can bound the probability that $\mathcal{C}$ succeeds.

When we put the results for two types of adversaries together, we obtain a (polynomial-time) reduction from an adversary breaking IND-RIBSC-CCA security to a challenger against a DBDH instance with $\frac{1}{2q|\mathcal{T}|}$ reduction loss. Thus we obtain the following advantage of $\mathcal{C}$ in solving the DBDH problem:

$$Adv(\mathcal{C}) > \frac{\epsilon}{64q|\mathcal{T}|(q_{ipk} + q_{fpk} + q_s + q_d)^2(n_u + 1)^2(q_s + q_d)(n_m + 1)}.$$

Regarding the running time of $\mathcal{C}$, one can take into account the running time t of $\mathcal{A}$ and the multiplications, the exponentiations and the pairings computation time in the series of queries and the challenge processes above. For simplicity and due to the fact that the pairing is the most dominant component in pairing based cryptosystems, we only count the number of pairing operations required. Thus, we have the time complexity bound of $\mathcal{C}$:

$$t' \leq t + \mathcal{O}((q_s + 8q_d)\tau),$$

where τ is the time of pairing computation. Thus, the theorem follows. □

Theorem 2. *If there exists an adversary $\mathcal{A}$ attacking EUF-RIBSC-CMA security of the proposed RIBSC*

scheme, then there exists a challenger $\mathcal{C}$ breaking a CDH problem instance.

Proof. $\mathcal{C}$ receives a random instance (g, g^a, g^b) of the CDH problem. $\mathcal{C}$ uses $\mathcal{A}$ as a subroutine to solve that instance and plays the role of $\mathcal{A}$' challenger in the game of Definition 4. The simulation process is the same as that described in Theorem 1.

At the end of the game, $\mathcal{A}$ produces a ciphertext $\sigma^* = (\sigma_0^*, \cdots, \sigma_6^*)$ of message M^*, time index T^* and two identities $\mathfrak{u}_s^*$ and $\mathfrak{u}_r^*$. If σ^* is a valid forgery, then $(\sigma_1^*, \sigma_4^*, \sigma_5^*, \sigma_6^*)$ is a valid signature of $\mathfrak{u}_s^*$ on message $\mathfrak{m}^*$, where $\mathfrak{m}^* = H(\sigma_0^*, \cdots, \sigma_5^*, \mathfrak{u}_s^*, \mathfrak{u}_r^*)$. If $F(\mathfrak{u}_s^*) \neq 0 \mod l_u$ and $K(\mathfrak{m}^*) \neq 0 \mod l_m$, then $\mathcal{C}$ fails and stops. Otherwise, $\mathcal{C}$ computes and outputs

$$\frac{\sigma_6^* \cdot (\sigma_1^*)^{L(\mathfrak{m}^*)}}{(\sigma_4^*)^{J(\mathfrak{u}_s^*)} \cdot (\sigma_5^*)^{\nu' + \nu T^*}}$$

$$= \frac{g_2^a (u' \prod_{i \in \mathcal{U}_s} u_i)^{r_s} (v' v^{T^*})^{r_t} (m' \prod_{j \in \mathcal{M}} m_j)^k (g^{-k})^{L(\mathfrak{m}^*)}}{(g^{r_s})^{J(\mathfrak{u}_s^*)} (g^{r_t})^{\nu' + \nu T^*}}$$

$$= \frac{g_2^a (g^{J(\mathfrak{u}_s^*)})^{r_s} (g_1^{-T^*} g^{\nu'} g_1^{T^*} g^{\nu T^*})^{r_t} (g^{L(\mathfrak{m}^*)})^k (g^{-k})^{L(\mathfrak{m}^*)}}{(g^{r_s})^{J(\mathfrak{u}_s^*)} (g^{r_t})^{\nu' + \nu T^*}}$$

$$= g_2^a = g^{ab}$$

which is the solution to the given CDH problem.

This completes the description of the simulation. It remains to analyze the probability of $\mathcal{C}$ success. Similar to the probability analysis of $\mathcal{C}$ in the Theorem 1, if $\mathcal{C}$ correctly guesses and does not abort, $\mathcal{C}$'s advantage is equal to $\mathcal{A}$'s advantage. The probability of $\mathcal{C}$'s correct guess is $1/(2q|\mathcal{T}|)$. On the other hand, for the simulation to complete without aborting, we require that at most all initial private key generation queries, full private key generation queries on an identity $\mathfrak{u}$ have $F(\mathfrak{u}) \neq 0 \mod l_u$, that at most all signcryption queries $(\mathfrak{u}, \mathfrak{u}_r, M, T)$ have $F(\mathfrak{u}) \neq 0 \mod l_u$, that at most all designcryption queries $(\mathfrak{u}_s, \mathfrak{u}, \sigma, T)$ have $F(\mathfrak{u}) \neq 0 \mod l_u$, and that $F(\mathfrak{u}_s^*) = 0 \mod l_u$ and $K(\mathfrak{m}^*) = 0 \mod l_m$. According to the same technique in [16, 17, 20, 21, 30], we can bound the probability that $\mathcal{C}$ succeeds. Thus we obtain the following advantage of $\mathcal{C}$ in solving the CDH problem instance:

$$Adv(\mathcal{C}) > \frac{\epsilon}{32q|\mathcal{T}|(q_{ipk} + q_{fpk} + q_s + q_d)(n_u + 1)(q_s + q_d)(n_m + 1)}.$$

Regarding the running time of $\mathcal{C}$, we only count the number of pairing operations required and have the time complexity bound of $\mathcal{C}$:

$$t' \leq t + \mathcal{O}((q_s + 8q_d)\tau),$$

where τ is the time of pairing computation. Thus, the theorem follows. □

6 Conclusions

In this paper, we have proposed an identity-based signcryption scheme with revocation functionality. In the proposed scheme, the master key is randomly divided into two parts: one is used to construct the initial key, the other is used to generate the updated key. These keys are used to periodically generate full private/decryption keys for non-revoked users. Thus, our method can revoke users in time and resist key exposure. Furthermore, we prove that our scheme has the IND-CCA2 security under the DBDH hardness assumption and has the EUF-CMA property under the CDH hardness assumption in the standard model. Compared with the previous schemes, our scheme supports key re-randomization, reduces the key update complexity from $\mathcal{O}(n - r)$ to $\mathcal{O}(r \log \frac{n}{r})$ with n the number of users and r the number of revoked users, and is proved to be secure without using the random oracles.

Finally, we remark that some interesting problems remain to be solved. Our RIBSC scheme has long public parameters and loose security reduction. Therefore, constructing efficient and tightly secure RIBSC schemes is an open problem. Furthermore, one natural question is how to construct a generic transformation from IBSC to RIBSC. On the other hand, our scheme is based on bilinear pairings, but it is interesting to construct post-quantum secure schemes based on other mathematical structure such as lattices.

Acknowledgement

We would like to thank the anonymous reviewers for their valuable comments and suggestions. This work is supported by the National Natural Science Foundation of China under Grants No.61472470, 61472309, 61100229 and 61173151, the China Scholarship Council under Grants No. 201208610019, the Natural Science Foundation of Shaanxi Province under Grant No. 2014JM2-6091, the Scientific Research Plan Project of Education Department of Shaanxi Province under Grants No.12JK0852, and the State Key Laboratory of Information Security under Grants No. (GW0704127001).

References

[1] P. Barreto, B. Libert, N. McCullagh, J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Advances in Cryptology-ASIACRYPT'05*, pp. 515-532, Springer-Verlag, 2005.

[2] A. Boldyreva, V. Goyal, and V. Kumar, "Identity based encryption with efficient revocation," in *Proceedings of the 15th ACM Conference on Computer and Communications Security-CCS'08*, pp. 417-426, ACM Press, 2008.

[3] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology-CRYPTO'01*, pp. 213-229, Springer-Verlag, 2001.

[4] X. Boyen, "Multipurpose identity-based signcryption: a swiss army knife for identity-based cryptography," in *Advances in Cryptology-CRYPTO'03*, pp. 383-399, Springer-Verlag, 2003.

[5] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisted," *Journal of the ACM*, vol. 51, no. 4, pp. 557-594, 2004.

[6] H. Chen, Y. Li, and J. Ren, "A practical identity-based signcryption scheme," *International Journal of Network Security*, vol. 15, no. 6, pp. 484-489, 2013.

[7] J. Chen, H. Lim, S. Ling, H. Wang, and K. Nguyen, "Revocable identity-based encryption from lattices," in *17th Australasian Conference on Information Security and Privacy-ACISP'12*, pp. 390-403, Springer-Verlag, 2012.

[8] A. Dent and Y. Zheng, "Practical Signcryption," Berlin: Springer-Verlag, 2010.

[9] D. He, J. Chen, and R. Zhang, "An efficient identity-based blind signature scheme without bilinear pairings," *Computers & Electrical Engineering*, vol. 37, no. 4, pp. 444-450, 2011.

[10] M. S. Hwang, S. T. Hsu, and C. C. Lee, "A new public key encryption with conjunctive field keyword search scheme", *Information Technology and Controlitc*, vol. 43, no. 3, pp. 277-288, 2014.

[11] Z. Jin, Q. Wen, and H. Du, "An improved semantically-secure identity-based signcryption scheme in the standard model," *Computers & Electrical Engineering*, vol. 36, no. 3, pp. 545-552, 2010.

[12] J. Kar, "Provably secure online/off-line identity-based signature scheme for wireless sensor network," *International Journal of Network Security*, vol. 16, no. 1, 2014, pp. 29-39

[13] C. C. Lee, C. H. Liu, and M. S. Hwang, "Guessing attacks on strong-password authentication protocol", *International Journal of Network Security*, vol. 15, no. 1, pp. 64-67, 2013.

[14] W. T. Li, C. H. Ling, and M. S. Hwang, "Group rekeying in wireless sensor networks: a survey", *International Journal of Network Security*, vol. 16, no. 6, pp. 400-410, 2014.

[15] F. Li, Y. Liao, Z. Qin, "Further improvement of an identity-based signcryption scheme in the standard model," *Computers & Electrical Engineering*, vol. 38, no. 2, pp. 413-421, 2012.

[16] F. Li and T. Takagi, "Secure identity-based signcryption in the standard model," *Mathematical and Computer Modelling*, vol. 57, no. 11-12, pp. 2685-2694, 2013.

[17] X. Li, H. Qian, J. Weng, and Y. Yu, "Fully secure identity-based signcryption scheme with shorter signcryptext in the standard model," *Mathematical and Computer Modelling*, vol. 57, no. 3-4, pp. 503-511, 2013.

[18] B. Libert and D. Vergnaud, "Adaptie-ID secure revocable identity based encryption," in *Topics in Cryptology CT-RSA'09*, pp. 1-15. Springer-Verlag, 2009.

[19] S. Liu, Y. Long, and K. Chen, "Key updating technique in identity-based encryption," *Information Sciences*, vol. 181, no. 11, pp. 2436-2440, 2011.

[20] K. Paterson and J. Schuldt, "Efficient identity based signatures secure in the standard model," in *11th Australasian Conference Information Security and Privacy-ACISP'06*, pp. 207-222. Springer-Verlag, 2006.

[21] J. Seo and K. Emura, "Revocable identity-based encryption revisited: security model and construction," in *Public-Key Cryptography-PKC'13*, pp. 216-234. Springer-Verlag, 2013.

[22] J. Seo and K. Emura, "Efficient delegation of key generation and revocation functionalities in identity-based encryption," in *Topics in Cryptology-CT-RSA'13*, pp. 343-358. Springer-Verlag, 2103.

[23] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology-CRYPTO'84*, pp. 47-53. Springer-Verlag, 1985.

[24] Y. Tseng, T. Tsai, and T. Wu, "Provably secure revocable ID-based signature in the standard model," *Security and Communication Networks*, http://dx.doi.org/10.1002/sec.696, 2013.

[25] Y. Tseng, T. Tsai, and T. Wu, "A fully secure revocable ID-based encryption in the standard model," *Informatica*, vol. 23, no. 3, pp. 487-505, 2012.

[26] Z. Wang, L. Wang, S. Zheng, Y. Yang, and Z. Hu, "Provably secure and efficient identity-based signature scheme based on cubic residues," *International Journal of Network Security*, vol. 14, no. 1, pp. 33-38, 2012.

[27] B. Waters, "Efficient identity-based encryption without random oracles," in *Advances in Cryptology-EUROCRYPT'05*, pp. 114-127, Springer-Verlag, 2005.

[28] T. Wu, T. Tsai, and Y. Tseng, "A revocable ID-based signcryption scheme," *Journal of Information Hiding and Multimedia Signal Processing*,vol. 3, no. 2, pp. 240-251, 2012.

[29] H. Xiong, J. Hu, and Z. Chen, "Security flaw of an ECC-based signcryption scheme with anonymity," *International Journal of Network Security*, vol. 15, no. 4, pp. 317-320, 2013.

[30] Y. Yu, B. Yang, Y. Sun, and S Zhu, "Identity based signcryption scheme without random oracles," *Computer Standards & Interfaces*, vol. 31, no. 1, pp. 56-62, 2009.

[31] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost (signature) + cost(encryption)," in *Advances in Cryptology-CRYPTO'97*, pp. 165-179, Springer-Verlag, 1997.

Xiangsong Zhang received her B.S. degree from Henan Normal University, M.S., and Ph.D degrees from Xidian University, China, in 2004, 2007 and 2011, respectively. She is a lecturer at Xi'an Technological University, Xi'an, China. Her research interests include the mathematical problems of cryptography.

Zhenhua Liu received his B.S. degree from Henan Normal University, M.S., and Ph.D degrees from Xidian University, China, in 2000, 2003 and 2009, respectively. He is an associate professor at Xidian University, Xi'an, China. His research interests include public key cryptography and information security.

Yupu Hu received his B.S., M.S., and Ph.D. degrees from Xidian University, China, in 1982, 1987 and 1999, respectively. He is currently a professor at the State Key Laboratory of Integrated Services Network, Xidian University. His mainly research interests include cryptography and information security.

Tsuyoshi Takagi received his B.Sc. and M.Sc. degrees in mathematics from Nagoya University in 1993 and 1995, respectively. He had engaged in the research on network security at NTT Laboratories from 1995 to 2001. He received the Dr.rer.nat degree from Technische University Darmstadt in 2001. He was an Assistant Professor in the Department of Computer Science at Technische University Darmstadt until 2005, and a Professor at the School of Systems Information Science in Future University-Hakodate, Japan until 2009. He is currently a Professor in Graduate School of Mathematics, Kyushu University. His current research interests are information security and cryptography. Dr. Takagi is a memeber of International Association for Cryptologic Research (IACR).