Efficient Pairing-Free Provably Secure Scalable Revocable Identity-Based Signature Scheme

Chang-Ji Wang[1], Hui Huang[2], Yuan Yuan[3]

[1] School of Information Science and Technology, Guangdong University of Foreign Studies, China
[2] School of Statistics and Mathematics, Guangdong University of Finance and Economics, China
[3] School of Mathematics and Statistics, Guangdong University of Foreign Studies, China
wchangji@126.com, xxxhuanghui@163.com, 200711647@oamail.gdufs.edu.cn

Abstract

Revocation functionality is essential for the practical deployment of identity-based cryptosystems because a user's private key may be corrupted by hacking or the period of a contract expires. Many researchers are focusing on revocable identity-based encryption scheme, while revocable identity-based signature scheme has received limited concentration. Recently, several revocable identity-based signature schemes have been proposed. However, these schemes are not scalable and are vulnerable to signing key exposure attack. In this paper, we first refine the security model of revocable identity-based signature scheme by considering the signing key exposure attack. Then, we propose a pairing - free revocable identity-based signature scheme with signing key exposure resistance. The proposed scheme is more efficient and practical than existing schemes in terms of update cost, sign cost, verification cost and scalability. Finally, we prove the proposed scheme is existentially unforgeable against adaptively chosen message and identity attacks under the standard discrete logarithm assumption in the random oracle model.

Keywords: Revocable identity-based signature, KUNode algorithm, Random oracle model

1 Introduction

The concept of identity-based cryptography was first proposed by Shamir to simplify certificate management in tradition public key infrastructure [1]. In an identity-based cryptosystem, an entity's public key can be publicly computed from arbitrary strings that uniquely identifies the entity, such as a complete name or an email address. A trusted third party named as private key generator (PKG) generates the private key for the entity and sends it to the entity through a secure channel. Since Boneh and Franklin proposed the first practical and secure identity-based encryption (IBE) scheme [2], various identity-based cryptosystems have been proposed and widely applied [3].

Key revocation mechanism is necessary for the practical deployment of any public key cryptosystems. For example, a user is no longer qualified for the public key, or a user's private key has been corrupted by hacking. In these cases, it is crucial for the cryptosystems to revoke the misbehaving or compromised user. In the tradition certificate setting, two popular solutions have been proposed, i.e., certificate revocation list and online certificate status protocol [4]. In the identity-based setting, however, key revocation is non-trivial because a user's identity is itself a public key, one cannot simply change his public key, as this changes his identity as well.

Boneh and Franklin proposed a naive revocation method for IBE scheme [2], which requires all users, regardless of whether their private keys have been exposed or not, have to regularly get in contact with the PKG, prove their identities and get new private keys. The PKG must be online for all such transactions, and a secure channel must be established between the PKG and each user to transmit the private key. Obliviously, this will lead to huge computation and communication overhead for the PKG that are linearly increased in the number of non-revoked users. Tseng and Tasi proposed an improved revocable IBE scheme with a public channel [5], where each user's decryption key consists of a fixed initial private key and a time update key, and the time update key is changed along with time period. The PKG periodically generates new time update keys and sends them to the non-revoked users via a public channel.

Boldyreva et al. proposed the first selective-ID secure, scalable revocable IBE scheme by combining the complete subtree method with a fuzzy IBE scheme [6], where the PKG is only required to perform logarithmic work in the number of users and no secure channel is required between users and the PKG. Then, Libert and Vergnaud proposed the first adaptive-ID secure, scalable revocable IBE scheme by combining the complete subtree method with a black-box accountable authority IBE scheme [7]. Later, Seo and

*Corresponding Author: Chang-Ji Wang; E-mail: wchangji@126.com
DOI: 10.3966/160792642020032102018

Emura considered the decryption key exposure attack, and proposed an adaptive-ID secure, scalable revocable IBE scheme with decryption key exposure resistance [8]. Recently, Wang et al. proposed an adaptive-ID secure, scalable revocable IBE scheme with constant size public system parameters and decryption key exposure resilience [9].

Although revocable IBE scheme has attracted researchers' attention in recent years, revocable identity-based signature (RIBS) scheme has received limited concentration. There are only a few RIBS schemes in the literature ([10-11]). In existing RIBS schemes, each user's signing key consists of a fixed initial private key and a time update key, and the time update key is changed along with time period. For non-revoked users, the PKG periodically generates new time update keys and sends them to the non-revoked users via a public channel. Thus, they are not scalable because the PKG has to generate time update key for all non-revoked users in each time period. Furthermore, exiting security model for RIBS scheme is a natural extension of the security model for the ordinary IBS scheme. It allows an adversary to obtain any private keys of a chosen identity, the only one restriction is that if the adversary obtains a private key of the challenge identity ID^*, then ID^* should be revoked before the challenge time T^*. However, it does not consider signing key exposure attack, wherein an adversary may obtain a signer's private key d_{ID^*} from a compromised signing key $sk_{\mathsf{ID}^*,\mathsf{T}}$, thus the adversary can forge signatures by combining it with subsequent updated keys. It is important and challenging task to construct a scalable RIBS scheme with signing key exposure resistance.

In this paper, we first refine the security model of RIBS scheme by considering the signing key exposure attack. Then, we propose a scalable RIBS scheme with signing key exposure resistance by combining the complete subtree method with Galindo and Garcia's IBS scheme without pairings [12]. Finally, we proved the proposed RIBS scheme is existentially unforgeable against chosen message and identity attacks under the standard discrete logarithm assumption in the random oracle model.

2 Preliminaries

We denote by $x \xleftarrow{\$} \mathbf{S}$ the operation of picking an element x uniformly at random from the set $\mathbf{S}$.

2.1 DLP Assumption

Let $\mathbf{G}_p$ be a prime p order group with a generator g. The Discrete Logarithm Problem (DLP) in $\mathbf{G}_p$ is defined as: given $g^x \in \mathbf{G}_p$ for unknown $x \xleftarrow{\$} \mathbf{Z}_p^*$, to compute x.

An adversary $\mathcal{A}$ has advantage ε in solving the DLP in $\mathbf{G}_p$ if $\Pr[\mathcal{A}(g, g^x) = x] \geq \varepsilon$, where the probability is over the random choice of x in $\mathbf{Z}_p^*$, and the random bits consumed by $\mathcal{A}$. The DLP assumption in $\mathbf{G}_p$ holds if all probabilistic polynomial time (PPT) adversaries have at most a negligible advantage in solving the DLP.

2.2 KUNode Algorithm

Denote by $\mathrm{Path}(\eta)$ the set of nodes on the path from a leaf node η to the root node of the binary tree $\mathbb{T}$, by ζ_L and ζ_R the left and right child of a non-leaf node ζ, respectively. The KUNode algorithm takes as input a binary tree $\mathbb{T}$, revocation list $\mathbf{RL}$, and time period T, it outputs a set of nodes. Each user is assigned to a leaf node. If a user (assigned to η) is revoked on time period T, then $(\eta, \mathsf{T}) \in \mathbf{RL}$. The algorithm is described as follows [9].

```
KUNode($\mathbb{T}, \mathbf{RL}, \mathsf{T}$)
    $\mathbf{X}, \mathbf{Y} \leftarrow \emptyset$
    $\forall(\eta_i, \mathsf{T}_i) \in \mathbf{RL}$,
        If $\mathsf{T}_i \leq \mathsf{T}$, then add $\mathrm{Path}(\eta_i)$ to $\mathbf{X}$
    $\forall x \in \mathbf{X}$
        If $x_L \notin \mathbf{X}$, then add $x_L$ to $\mathbf{Y}$
        If $x_R \notin \mathbf{X}$, then add $x_R$ to $\mathbf{Y}$
    If $\mathbf{Y} = \emptyset$, then add root to $\mathbf{Y}$
    Output $\mathbf{Y}$
```

Upon registration, the PKG assigns a leaf node η of $\mathbb{T}$ to the user, and provides the user with a set of distinct private keys, wherein each private key is associated with a node on $\mathrm{Path}(\eta)$. At time period T, the PKG broadcasts key updates for a set $\mathbf{Y} \subset \mathbb{T}$ of nodes which contains no ancestors of revoked users and precisely one ancestor of any non-revoked user.

3 Definitions of RIBS Scheme

We denote by $\mathbf{I}$, $\mathbf{T}$ and $\mathbf{M}$ the identity space, the time space and the message space, respectively.

3.1 Syntax Definition of RIBS Scheme

A RIBS scheme consists of the following seven polynomial time algorithms:

Setup. The setup algorithm takes as input a security parameter κ and a maximal number of users N. It outputs the public parameter mpk, the master secret key msk, the initial revocation list $\mathbf{RL} = \emptyset$, and an initial state st_0.

KeyGen. The key generation algorithm takes as input the public parameter mpk, the master secret key msk, an identity $\mathsf{ID} \in \mathbf{I}$ and a state st. It outputs a private key d_{ID} of ID and an updated state st.

KeyUp. The key update algorithm takes as input the public parameter mpk, the master secret key msk, the key update time $\mathsf{T} \in \mathbf{T}$, the current revocation list $\mathbf{RL}$, and state st. It outputs the update key ku_{T}.

SKG. The signing key generation algorithm takes as input the public parameter mpk, a user's private key d_{ID} and the update key ku_{T}. It outputs a signing key $sk_{\mathsf{ID,T}}$ that is valid on time period T or a special symbol $\perp$ indicating that ID was revoked.

Sign. The signing algorithm takes as input the public parameter mpk, a message m, and the signer's signing key $sk_{\mathsf{ID,T}}$. It outputs the signature $\sigma_{\mathsf{ID,T}}$.

Verify. The verification algorithm takes as input the public parameter mpk and a pair of message and signature $(m, \sigma_{\mathsf{ID,T}})$. It outputs 1 if $\sigma_{\mathsf{ID,T}}$ is valid signature on m for identity ID signed on time period T. Otherwise, it outputs 0.

Revoke. The key revocation algorithm takes as input an identity ID to be revoked, a revocation time period T, revocation list $\mathbf{RL}$, and the state st. It outputs an updated revocation list $\mathbf{RL}'$.

The consistency condition requires that for a message $m \in \mathbf{M}$ and corresponding signature $\sigma_{\mathsf{ID,T}}$ which is signed by a non-revoked user with identity ID on time period T, $\mathbf{Verify}(mpk, m, \sigma_{\mathsf{ID,T}})$ must return 1.

3.2 Security Definition of RIBS Scheme

The existentially unforgeable against adaptively chosen message and identity attacks (EUF-ID-CMA) for RIBS scheme is defined via the following game interacting between a forger $\mathcal{F}$ and a simulator $\mathcal{B}$.

The existentially unforgeable against adaptively chosen message and identity attacks (EUF-ID-CMA) for RIBS scheme is defined via the following game interacting between a forger $\mathcal{F}$ and a simulator $\mathcal{B}$.

Setup. $\mathcal{B}$runs $\mathbf{Setup}(1^\kappa, N) \to (msk, mpk)$, and sends mpk to $\mathcal{F}$, while keeps msk secret.

Queries. $\mathcal{F}$ is allowed to issue the following queries adaptively.

- *KeyGen Oracle*: For an identity $\mathsf{ID} \in \mathbf{I}$, $\mathcal{B}$ forwards d_{ID} and st to $\mathcal{F}$by running $\mathbf{KeyGen}(mpk, msk, \mathsf{ID}, \mathsf{st}) \to (d_{\mathsf{ID}}, \mathsf{st})$.
- *KeyUp Oracle*: For a time period $\mathsf{T} \in \mathbf{T}$, $\mathcal{B}$ runs $\mathbf{KeyUp}(mpk, msk, \mathsf{T}, \mathbf{RL}, \mathsf{st}) \to ku_{\mathsf{T}}$ and forwards the update key ku_{T} to $\mathcal{F}$.
- *SKG Oracle*: For any identity $\mathsf{ID} \in \mathbf{I}$ on any time period $\mathsf{T} \in \mathbf{T}$, and $\mathcal{B}$ forwards $sk_{\mathsf{ID,T}}$ to $\mathcal{F}$ by sequentially running $\mathbf{KeyGen}(mpk, msk, \mathsf{ID}, \mathsf{st}) \to d_{\mathsf{ID}}$, $\mathrm{KeyUp}(mpk, msk, \mathsf{T}, \mathbf{RL}, \mathsf{st}) \to ku_{\mathsf{T}}$ and $\mathbf{SKG}(mpk, d_{\mathsf{ID}}, ku_{\mathsf{T}}) \to sk_{\mathsf{ID,T}}$.
- *Sign Oracle*: For any message $m \in \mathbf{M}$ of any identity $\mathsf{ID} \in \mathbf{I}$ on any time period $\mathsf{T} \in \mathbf{T}$, and $\mathcal{B}$ forwards $\sigma_{\mathsf{ID,T}}$ to $\mathcal{F}$ by sequentially running $\mathbf{KeyGen}(mpk, msk, \mathsf{ID}, \mathsf{st}) \to d_{\mathsf{ID}}$, $\mathbf{KeyUp}(mpk, msk, \mathsf{T}, \mathbf{RL}, \mathsf{st}) \to ku_{\mathsf{T}}$, $\mathbf{SKG}(mpk, d_{\mathsf{ID}}, ku_{\mathsf{T}}) \to sk_{\mathsf{ID,T}}$ and $\mathbf{Sign}(mpk, m, sk_{\mathsf{ID,T}}) \to \sigma_{\mathsf{ID,T}}$.
- *Revoke Oracle*: For the revocation of any identity $\mathsf{ID} \in \mathbf{I}$ on any time period $\mathsf{T} \in \mathbf{T}$, $\mathcal{B}$ forwards $\mathbf{RL}'$ to $\mathcal{F}$ by running $\mathbf{Revoke}(mpk, \mathsf{ID}, \mathsf{T}, \mathbf{RL}, \mathsf{st}) \to \mathbf{RL}'$.

$\mathcal{F}$ is allowed to query above oracles with the following restrictions:

- *SKG Oracle* cannot be queried on time period T before *KeyUp Oracle* was queried.
- *KeyUp Oracle* and *Revoke Oracle* can be queried on time period which is greater than or equal to the time period of all previous queries.
- *Revoke Oracle* cannot be queried on time period T if *KeyUp Oracle* was queried on T.

Note that $\mathcal{F}$ can access the signing key oracle in our security model, which is not given in the existing security model for RIBS schemes [10-11].

Forge. At the end of the game, $\mathcal{F}$ outputs a forgery of a pair of message and signature $(\hat{m}, \sigma_{\hat{\mathsf{ID}},\hat{\mathsf{T}}})$ on behalf of $\hat{\mathsf{ID}}$ on $\hat{\mathsf{T}}$.

We say that $\mathcal{F}$ wins the experiment provided that the following conditions hold:

- $\mathbf{Verify}(mpk, \hat{\mathsf{ID}}, \hat{\mathsf{T}}, \hat{m}, \sigma_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}) = 1$.
- If $\mathbf{KeyGen}(\hat{\mathsf{ID}})$ was queried, then $\mathbf{Revoke}(\hat{\mathsf{ID}}, \mathsf{T})$ must be queried for $\mathsf{T} \le \hat{\mathsf{T}}$.
- *SKG Oracle* cannot be queried with respect to $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- *Sign Oracle* cannot be queried with respect to $(\hat{\mathsf{ID}}, \hat{\mathsf{T}}, \hat{m})$.

The forger's advantage ϵ in the above game is defined by the probability that $\mathcal{F}$ outputs a valid forgery. If there does not exist any PPT forger who has non-negligible advantage in the above game, we say that a RIBS scheme is EUF-ID-CMA secure.

4 The Proposed RIBS Scheme

The proposed RIBS scheme is described as follows.

$\mathbf{Setup}(1^\kappa, N)$. The PKG first chooses a prime $p (2^\kappa \le p < 2^{\kappa+1})$ order group $\mathbf{G}_p$ with a generator g, $x_1, x_2, x_3 \overset{\$}{\leftarrow} \mathbf{Z}_p$, then computes $h_1 = g^{x_1}$, $h_2 = g^{x_2}$ and $h_3 = g^{x_3}$. The PKG also chooses three cryptographic hash functions $H_1 : \mathbf{G}_p \times \mathbf{I} \to \mathbf{Z}_p$, $H_2 : \mathbf{G}_p \times \mathbf{T} \to \mathbf{Z}_p$ and $H_3 : \mathbf{M} \times \mathbf{G}_p \times \mathbf{I} \times \mathbf{T} \to \mathbf{Z}_p$. Finally, the PKG sets $mpk = \langle \mathbf{G}_p, g, p, h_1, h_2, h_3, H_1, H_2, H_3 \rangle$, $msk = \langle x_1, x_2, x_3 \rangle$, $\mathbf{RL} = \emptyset$and $\mathsf{st} = \mathbf{BT}$, where $\mathbf{BT}$ is a binary tree with N leaves.

$\mathbf{KeyGen}(mpk, msk, \mathsf{ID}, \mathsf{st})$. The PKG chooses an unassigned leaf node η from $\mathbf{BT}$ at random, and stores ID in the node η. For each node $\theta \in \mathsf{Path}(\eta)$, the PKG performs as follows.

(1) Recall c_θ if it was defined. Otherwise, choose $c_\theta \overset{\$}{\leftarrow} \mathbf{Z}_p$ and store $(c_\theta, \tilde{c}_\theta = x_3 - c_\theta)$ in the node η.

(2) Choose $r_{1,\theta} \overset{\$}{\leftarrow} \mathbf{Z}_p$, compute $R_{1,\theta} = g^{r_{1,\theta}}$ and $y_{1,\theta} = r_{1,\theta} + x_1 \cdot H_1(R_{1,\theta}, \mathsf{ID}) + c_\theta$, and set $d_{\mathsf{ID},\theta} = (R_{1,\theta}, y_{1,\theta})$.

(3) Finally, the PKG assigns the user with identity ID the private key $d_{\mathsf{ID}} = \{(\theta, d_{\mathsf{ID},\theta})\}_{\theta \in \mathsf{Path}(\eta)}$.

$\mathbf{KeyUp}(mpk, msk, \mathbf{T}, \mathbf{RL}, \mathsf{st})$. The PKG parses $\mathsf{st} = \mathbf{BT}$ and performs the following steps for each node $\theta \in \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$.

(1) Retrieve $\tilde{c}_\theta$ (note that $\tilde{c}_\theta$ is always pre-defined in the **KeyGen** algorithm).

(2) Choose $r_{2,\theta} \xleftarrow{\$} \mathbf{Z}_p$, compute $R_{2,\theta} = g^{r_{2,\theta}}$ and $y_{2,\theta} = r_{2,\theta} + x_2 \cdot H_2(R_{2,\theta}, \mathsf{T}) + \tilde{c}_\theta$, and set $\mathsf{ku}_{\mathsf{T},\theta} = (R_{2,\theta}, y_{2,\theta})$.

(3) Return $ku_{\mathsf{T}} = \{(\theta, ku_{\mathsf{T},\theta})\}$.

$\mathbf{SKG}(mpk, d_{\mathsf{ID}}, ku_{\mathsf{T}})$. The PKG checks whether the intersection of two sets $\mathsf{Path}(\eta)$ and $\mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$ is an empty set or not. If it is an empty set, it returns $\perp$. Otherwise, the PKG performs the following steps for each node $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$.

(1) Choose $\$r_d \xleftarrow{\$} \mathbf{Z}_p$ and compute $h_1' = h_1^{r_d}$, $h_2' = h_2^{r_d}$, $h_3' = h_3^{r_d}$, $R_{1,\theta}' = R_{1,\theta}^{r_d}$, $R_{2,\theta}' = R_{2,\theta}^{r_d}$, $D_\theta = r_d(y_{1,\theta} + y_{2,\theta})$.

(2) Return $sk_{\mathsf{ID},\mathsf{T}}$as $(h_1', h_2', h_3', \{D_\theta, R_{1,\theta}, R_{2,\theta}, R_{1,\theta}', R_{2,\theta}'\})$.

Note: In both Wu et al.'s RIBS scheme [10] and Sun et al.'s RIBS scheme [11], signer does not update his/her signing private key $sk_{\mathsf{ID},\mathsf{T}}$, and just adds long-term private key component d_{ID} with update key component ku_{T} corresponding to the current time period T directly. Consequently, their scheme cannot withstand signing key exposure. However, our scheme re-randomizes the signing private key $sk_{\mathsf{ID},\mathsf{T}}$ in each update process to conquer this kind of attack.

$\mathbf{Sign}(mpk, sk_{\mathsf{ID},\mathsf{T}}, m)$. To sign a message m on the time period T, the signer generates a signature $\sigma_{\mathsf{ID},\mathsf{T}}$ on m using his signing key $sk_{\mathsf{ID},\mathsf{T}}$ by performing the following steps.

(1) Choose $a \xleftarrow{\$} \mathbf{Z}_p$ and compute $A = g^a$.

(2) For $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$, compute $b_\theta = a + D_\theta \cdot H_3(m, A, \mathsf{ID}, \mathsf{T})$.

(3) For $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$, return $\sigma_{\mathsf{ID},\mathsf{T}}$ as $(A, h_1', h_2', h_3', \{b_\theta, R_{1,\theta}, R_{2,\theta}, R_{1,\theta}', R_{2,\theta}'\})$.

$\mathbf{Verify}(mpk, \sigma_{\mathsf{ID},\mathsf{T}}, m)$. Upon receiving a signature $\sigma_{\mathsf{ID},\mathsf{T}}$ on message m for ID on the time period T, a verifier checks the equation $g^{b_\theta} = A(R_{1,\theta}' R_{2,\theta}' {h_1'}^{H_1(\mathsf{ID}, R_{1,\theta})} {h_2'}^{H_2(\mathsf{T}, R_{2,\theta})} h_3')^{H_3(m, A, \mathsf{ID}, \mathsf{T})}$holds or not. If it holds, the algorithm outputs 1; otherwise, the algorithm outputs 0.

$\mathbf{Revoke}(mpk, \mathsf{ID}, \mathsf{T}, \mathbf{RL}, \mathsf{st})$. Let η be the leaf node associated with ID. The PKG updates the revocation list by $\mathbf{RL}' \leftarrow \mathbf{RL} \cup \{(\eta, \mathsf{T})\}$.

5 Efficiency Analysis and Security Proof of the Proposed RIBS Scheme

5.1 Efficiency Analysis

We denote by P a computation of the bilinear pairing $\hat{e}(\mathbf{G}_1, \mathbf{G}_1) \to \mathbf{G}_T$, by M a scalar multiplication in $\mathbf{G}_1$, by H a map-to-point hash function, by E an exponentiation in $\mathbf{G}_p$, by N the maximum number of users in the system, and by r the number of revoked users. Compared to scalar multiplication in $\mathbf{G}_1$, map-to-point hash function, and exponentiation in $\mathbf{G}_p$, bilinear pairing is considered as the most expensive computational operations. The relative computation cost of a bilinear pairing is approximately twenty times higher than that of the scalar multiplication over elliptic curve group [13].

Table 1. Comparison of RIBS schemes

	[7]	[8]	Ours
Scalability	No	No	Yes
Update cost	$O(N-r)$	$O(N-r)$	$O(r\log(N/r))$
Sign cost	$2M$	E	E
Verify cost	$2P+M+2H$	$3E$	$4E$

As shown in the Table 1, our RIBS scheme is more efficient and practical than existing RIBS schemes ([10-11]) in terms of update cost, sign cost, verification cost and scalability.

5.2 Security Proof of the Proposed RIBS Scheme

Theorem 1. Let $\mathcal{F}$ be an $(\epsilon, q_{H_1}, q_{H_2}, q_{H_3})$-forger against RIBS in the EUF-ID-CMA model, where q_{H_1}, q_{H_2} and q_{H_3} are denoted by the upper bound of the number of queries on H_1-oracle, H_2-oracle and H_3-oracle, respectively. If H_1, H_2 and H_3 are modeled as random oracles, we can construct either

- Algorithm $\mathcal{R}$ which ε-breaks the DLP, where $\varepsilon \geq \epsilon^2/q^4 q_{H_3}$.
- Algorithm $\mathcal{R}_1$ which ε_1-breaks the DLP, where $\varepsilon_1 \geq \epsilon^2/q^4(q_{H_3}+q_{H_2})^2 + \epsilon^2/q^4(q_{H_3}+q_{H_2})^6$.
- Algorithm $\mathcal{R}_2$ which ε_2-breaks the DLP, where $\varepsilon_2 \geq \epsilon^2/q^4(q_{H_3}+q_{H_1})^2 + \epsilon^2/q^4(q_{H_3}+q_{H_1})^6$.
- Algorithm $\mathcal{R}_3$ which ε_3-breaks the DLP, where $\varepsilon_3 \geq 4\epsilon^2/q^4(q_{H_3}+q_{H_1}+q_{H_2})^2 + 2\epsilon^2/q^4(q_{H_3}+q_{H_1}+q_{H_2})^6$

Proof: Consider the following events in the case that a forger $\mathcal{F}$ produces successfully a valid forgery $\hat{\sigma}_{\hat{\mathsf{ID}},\hat{\mathsf{T}}} = (\hat{A}, \{\hat{b}_\theta, \hat{R}_{1,\theta}, \hat{R}_{2,\theta}\})$ on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}}, m)$for $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$.

- E: $\mathcal{F}$ makes at least one signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$, and $(\hat{R}_{1,\theta}, \hat{R}_{2,\theta})$ were returned by the simulator $\mathcal{B}$ as part of the output to a signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- E_1: $\mathcal{F}$ makes at least one signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$, and $\hat{R}_{1,\theta}$ was returned by the simulator as part of the output to a signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- E_2: $\mathcal{F}$ makes at least one signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$, and $\hat{R}_{2,\theta}$ was returned by the simulator as part of the output to a signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- E_3: Either $\mathcal{F}$ does not make any signature queries on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$ or both $\hat{R}_{1,\theta}$ and $\hat{R}_{2,\theta}$ were never returned by the simulator as part of the output to a signature query on $(\hat{\mathsf{ID}}, \hat{\mathsf{T}})$.

To forge a signature $\hat{\sigma}_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}$ with a non-negligible probability, $\mathcal{F}$ has to issue three random oracle queries: $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$, $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ and $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$.

According to the order in which the adversary issues these queries, we further subdivide the event E_1 into the following events:

- $\mathsf{E}_{1,(2,3)}$: $\mathcal{F}$ issues the query on $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ before the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- $\mathsf{E}_{1,(3,2)}$: $\mathcal{F}$ issues the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ before the query on $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$.

In the similar way, we can subdivide the event E_2 into the following events $\mathsf{E}_{2,(1,3)}$ and $\mathsf{E}_{2,(3,1)}$, and subdivide the event E_3 into the following events $\mathsf{E}_{3,(1,2,3)}$, $\mathsf{E}_{3,(1,3,2)}$, $\mathsf{E}_{3,(2,1,3)}$, $\mathsf{E}_{3,(2,3,1)}$, $\mathsf{E}_{3,(3,1,2)}$ and $\mathsf{E}_{3,(3,2,1)}$, respectively.

- $\mathsf{E}_{2,(1,3)}$: $\mathcal{F}$ issues the query on $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$ before the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$.
- $\mathsf{E}_{2,(3,1)}$: $\mathcal{F}$ issues the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ before the query on $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$.
- $\mathsf{E}_{3,(1,2,3)}$: $\mathcal{F}\ \mathcal{A}$ issues the query on $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$, $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ and $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ in order.
- $\mathsf{E}_{3,(1,3,2)}$: $\mathcal{F}$ issues the query on $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$, $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ and $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ in order.
- $\mathsf{E}_{3,(2,1,3)}$: $\mathcal{F}$ issues the query on $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$, $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$ and $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ in order.
- $\mathsf{E}_{3,(2,3,1)}$: $\mathcal{F}$ issues the query on $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$, $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ and $H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$ in order.
- $\mathsf{E}_{3,(3,1,2)}$: $\mathcal{F}$ issues the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$ and $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ in order.
- $\mathsf{E}_{3,(3,2,1)}$: $\mathcal{F}$ issues the query on $H_3(m, \hat{A}, \hat{\mathsf{ID}}, \hat{\mathsf{T}})$, $H_2(\hat{R}_{2,\theta}, \hat{\mathsf{T}})$ and $\$H_1(\hat{R}_{1,\theta}, \hat{\mathsf{ID}})$ in order.

In the case of the events E, E_1, E_2 and E_3, we give the reductions $\mathcal{R}$, $\mathcal{R}_1$, $\mathcal{R}_2$ and $\mathcal{R}_3$ respectively.

For simplicity, we only give the reduction $\mathcal{R}$ and $\mathcal{R}_1$. We deal with $\mathcal{R}$ using general forking lemma [14], and we deal with $\mathcal{R}_1$ adopting general forking lemma [14] and multiple forking lemma [15]. $\mathcal{R}_2$ and $\mathcal{R}_3$ can be analogously constructed as $\mathcal{R}_1$.

To deal with the challenge identity, we adopt the strategy which is utilized by Seo and Emura in [8]. Let q be the maximum number of queries regarding H_1-oracle and H_3-oracle. Simulator $\mathcal{B}$ randomly guesses $\hat{i} \in [1, q]$ such that $\mathcal{A}$'s $\hat{i}$-th query is the first query regarding $\hat{\mathsf{ID}}$ on H_1-oracle and H_3-oracle. We assume $\mathcal{B}$'s guess is right (It holds with probability $1/q$). As discussed in [7-8], time space $\mathbf{T}$ are widely considered to be polynomial in κ . $\mathcal{B}$ randomly guesses the challenge time $\hat{\mathsf{T}} \in \mathbf{T}$, we also assume that $\mathcal{B}$'s guess is right (It holds with probability $1/|\mathbf{T}|$).

5.2.1 Reduction $\mathcal{R}$

Let $\Pi \triangleq (\mathbf{G}_p, p, g, g^{\alpha})$ be the given DLP instance. $\mathcal{B}$ chooses $x_1, x_2, x_3 \xleftarrow{\$} \mathbf{Z}_p^*$, computes $h_1 = g^{x_1}$, $h_2 = g^{x_2}$ and $h_3 = g^{x_3}$, sets $msk = (x_1, x_2, x_3)$ and $mpk = (\mathbf{G}_p, p, g, h_1, h_2, h_3)$, and sends mpk to $\mathcal{F}$. The hash functions H_1 , H_2 and H_3 are modelled as random oracles. This is done with the aid of three tables L_{H_1}, L_{H_2} and L_{H_3}. $\mathcal{B}$ simulates $\mathcal{F}$'s environment as follows:

- H_1 , H_2 and H_3 *Oracle Query*: $\mathcal{B}$ simulates those oracles by keeping L_{H_1}, L_{H_2} and L_{H_3} containing the queried values together with the answers.
- *KeyGen Oracle Query*: $\mathcal{F}$ issues a private key query on identity ID. If there is a tuple which is related to ID in L_{kgo}, then $\mathcal{B}$ return as the list. Otherwise, $\mathcal{B}$ randomly choose a leaf node η in binary tree BT and assign ID to η . For each node $\theta \in \mathsf{Path}(\eta)$, $\mathcal{B}$ performs the following steps.

(1) Recall $c_\theta \xleftarrow{\$} \mathbf{Z}_p$ and store$(c_\theta, \tilde{c}_\theta = x_3 - c_\theta)$ in the node η.

(2) Choose $r_{1,\theta} \xleftarrow{\$} \mathbf{Z}_p$ and $z_{1,\theta} \xleftarrow{\$} \mathbf{Z}_p$, compute $R_{1,\theta} = g^{r_{1,\theta}}$ and $y_{1,\theta} = r_{1,\theta} + x_1 z_{1,\theta} + c_\theta$.

(3) Add $((\mathsf{ID}, R_{1,\theta}), z_{1,\theta})$ to the list L_{H_1} to the list L_{H_1}.

(4) Return $d_{\mathsf{ID}} = \{\theta, R_{1,\theta}, y_{1,\theta}\}_{\theta \in \mathsf{Path}(\eta)}$.

(5) Add $(\mathsf{ID}, d_{\mathsf{ID}})$ to the list L_{kgo}.

- *KeyUp Oracle Query*: $\mathcal{F}$issues a key update query on time T. If there is a tuple which is related to T in L_{kuo} , then $\mathcal{B}$ return as the list. Otherwise, $\mathcal{B}$ performs the following steps for each node $\theta \in \mathsf{KUNode}(\mathsf{BT}, \mathsf{RL}, \mathsf{T})$.

(1) Retrieve $\tilde{c}_\theta$.

(2) Choose $r_{2,\theta} \xleftarrow{\$} \mathbf{Z}_p$ and $z_{2,\theta} \xleftarrow{\$} \mathbf{Z}_p$.

(3) Compute $R_{2,\theta} = g^{r_{2,\theta}}$ and $y_{2,\theta} = r_{2,\theta} + x_2 z_{2,\theta} + \tilde{c}_\theta$.

(4) Add $((\mathsf{T}, R_{2,\theta}), z_{2,\theta})$ to the list L_{H_2}.

(5) Return $\mathsf{ku}_{\mathsf{T}} = \{\theta, R_{2,\theta}, y_{2,\theta}\}$.

(6) Add $(\mathsf{T}, \mathsf{ku}_{\mathsf{T}})$ to the list L_{kuo}.

- *Sign Oracle Query*: When $\mathcal{F}$ makes a sign oracle query on $(\mathsf{ID}, \mathsf{T}, m)$ with $\mathsf{ID} \neq \hat{\mathsf{ID}} \vee \mathsf{T} \neq \hat{\mathsf{T}}$ and ID is not revoked on time T, $\mathcal{B}$ simply computes the private key for ID and the update key on time period T as described in the previous *KeyGen Oracle Query* and *KeyUp Oracle Query*. Then $\mathcal{B}$ runs the **Sign** algorithm and returns the produced signature to $\mathcal{F}$. In the case of $\mathsf{ID} = \hat{\mathsf{ID}} \wedge \mathsf{T} = \hat{\mathsf{T}}$, for $\theta \in \mathsf{Path}(\eta) \cap \mathsf{KUNode}(\mathbf{BT}, \mathbf{RL}, \mathsf{T})$, $\mathcal{B}$ acts as follows.

(1) Choose $\hat{r}_{1,\theta} \xleftarrow{\$} \mathbf{Z}_p$ and compute $g^{\hat{r}_{2,\theta}} = g^{\alpha}/g^{\hat{r}_{1,\theta}}$.

(2) If no tuple in list L_{H_1} related to $(g^{\hat{r}_{1,\theta}}, \hat{\mathsf{ID}})$, then $\mathcal{B}$ chooses $z_{1,\theta} \xleftarrow{\$} \mathbb{Z}_p$ and adds it to list L_{H_1}. If no tuple in list L_{H_2} related to $(g^{\hat{r}_{2,\theta}}, \hat{\mathsf{T}})$, then $\mathcal{B}$ chooses $z_{2,\theta} \xleftarrow{\$} \mathbb{Z}_p$ and adds it to list L_{H_2}.

(3) Choose $t_\theta \xleftarrow{\$} \mathbb{Z}_p$, $e \xleftarrow{\$} \mathbb{Z}_p$, compute $b_\theta = t_\theta + (\hat{r}_{1,\theta} + x_1 z_{1,\theta} + x_2 z_{2,\theta} + x_3)e$, $A = g^{t_\theta}/(g^{\alpha}/g^{\hat{r}_{1,\theta}})^e$, and add $((m, A, \hat{\mathsf{ID}}, \hat{\mathsf{T}}), e)$ to list L_{H_3}.

(4) Return $\sigma_{\hat{\mathsf{ID}}, \hat{\mathsf{T}}} = (A, \{b_\theta, g^{\hat{r}_{1,\theta}}, g^{\hat{r}_{2,\theta}}\})$ to $\mathcal{F}$.

The correctness of the forged signature can be verified as follows.

$$\begin{aligned} g^{b_\theta} &= g^{t_\theta+(\hat{r}_{1,\theta}+x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)e} \\ &= A\cdot(g^{\alpha}/g^{\hat{r}_{1,\theta}})^e\cdot g^{(\hat{r}_{1,\theta}+x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)e} \\ &= A\cdot g^{(\alpha+x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)e} \\ &= A\cdot(g^{\hat{r}_{1,\theta}}\cdot g^{\hat{r}_{2,\theta}}\cdot g^{x_1z_{1,\theta}+x_2z_{2,\theta}+x_3})^e \end{aligned}$$

Algorithm $\mathcal{R}$ now uses the general forking algorithm $\mathcal{F}^{\mathrm{GF}}$ to solve the DLP challenge [14]. It runs $\mathcal{F}^{\mathrm{GF}}$ on the given DLP instance Π, with the H_3 involved in the replay attack. If $\mathcal{F}^{\mathrm{GF}}$ fails, $\mathcal{R}$ aborts. If $\mathcal{F}^{\mathrm{GF}}$ is successful, $\mathcal{R}$ can get two valid forgeries $\hat{\sigma}^{(i)}_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}=(\hat{A},\{\hat{b}^{(i)}_\theta,\hat{R}^{(i)}_{1,\theta},\hat{R}^{(i)}_{2,\theta}\}_{i=0,1})$ with respect to $(\hat{\mathsf{ID}},\hat{\mathsf{T}},m)$, where $\theta\in\mathsf{Path}(\eta)\cap\mathsf{KUNode}(\mathbf{BT},\mathbf{RL},\mathsf{T})$. Next, we compute the discrete logarithm α as follows.

$$\begin{cases} b^{(0)}_\theta = \hat{a}+(\alpha+x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)e^{(0)} \\ b^{(1)}_\theta = \hat{a}+(\alpha+x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)e^{(1)} \end{cases} \Rightarrow$$

$$\alpha=(b^{(1)}_\theta-b^{(0)}_\theta)/(e^{(1)}-e^{(0)})-(x_1z_{1,\theta}+x_2z_{2,\theta}+x_3)$$

To bound the success probability of $\mathcal{R}$ against the DLP assumption, $\mathcal{F}^{\mathrm{GF}}$ is successful during running if $\mathcal{B}$'s guess for identity and time is right. We denote this probability by acc, and apply the general forking lemma [14] with $|\mathbf{S}|=p$ and $\gamma=q_{H_3}$, we get $\mathsf{suc}\geq\mathsf{acc}(\mathsf{acc}/q_{H_3}-1/p)\geq\varepsilon^2/q^4q_{H_3}$. Thus $\mathcal{R}$ can successfully solve DLP with advantage $\varepsilon\geq\epsilon^2/q^4q_{H_3}$.

This ends the proof.

5.2.2 Reduction $\mathcal{R}_1$

In this case, $\mathcal{F}$ makes at least one signature query on $(\hat{\mathsf{ID}},\hat{\mathsf{T}})$ and $\hat{R}_{1,\theta}$ was returned by $\mathcal{B}$ as part of the output to a signature query on $(\hat{\mathsf{ID}},\hat{\mathsf{T}})$, but $\hat{R}_{2,\theta}$ was not returned.

Let $\Pi\triangleq(\mathbf{G}_p,p,g,g^\alpha)$ be the given DLP instance. $\mathcal{B}$ chooses $x_1,x_3\xleftarrow{\$}\mathbf{Z}_p^*$, computes $h_1=g^{x_1}$ and $h_3=g^{x_3}$, sets $msk=(x_1,x_3)$ and $mpk=(\mathbf{G}_p,p,g,h_1,g^\alpha,h_3)$, and sends mpk to $\mathcal{F}$. In fact, $\mathcal{B}$ sets $x_2=\alpha$ implicitly.

$\mathcal{B}$ adopts the same mechanism to response $\mathcal{F}$'s queries for both $\mathsf{E}_{1,(2,3)}$ and $\mathsf{E}_{1,(3,2)}$.

- H_1, H_2 and H_3 *Oracle Query*: $\mathcal{B}$ performs the same steps as it did in 5.2.1.
- *KeyGen Oracle Query*: $\mathcal{B}$ performs the same steps as it did in 5.2.1.
- *KeyUp Oracle Query*: $\mathcal{F}$ issues a key update query on time T. If there is a tuple which is related to T in L_{kuo}, then $\mathcal{B}$ return as the list. Otherwise, $\mathcal{B}$ performs the following steps for each node $\theta\in\mathsf{KUNode}(\mathsf{BT},\mathsf{RL},\mathsf{T})$.

  (1) Retrieve $\tilde{c}_\theta$.

  (2) Choose $t_{2,\theta}\xleftarrow{\$}\mathbf{Z}_p$ and $z_{2,\theta}\xleftarrow{\$}\mathbf{Z}_p$.

  (3) Compute $R_{2,\theta}=g^{t_{2,\theta}}(g^\alpha)^{-z_{2,\theta}}$ and $y_{2,\theta}=t_{2,\theta}+\tilde{c}_\theta$.

  (4) Add $((\mathsf{T},R_{2,\theta}),z_{2,\theta})$ to the list L_{H_2}.

  (5) Return $\mathsf{ku_T}=\{\theta,R_{2,\theta},y_{2,\theta}\}$.

  (6) Add $(\mathsf{T},\mathsf{ku_T})$ to the list L_{kuo}.
- *Sign Oracle Query*: $\mathcal{B}$ performs the same steps as it did in 5.2.1 when $\mathsf{ID}\neq\hat{\mathsf{ID}}\vee\mathsf{T}\neq\hat{\mathsf{T}}$ and ID is not revoked on time T. If $\mathsf{ID}=\hat{\mathsf{ID}}\wedge\mathsf{T}=\hat{\mathsf{T}}$, for $\theta\in\mathsf{Path}(\eta)\cap\mathsf{KUNode}(\mathbf{BT},\mathbf{RL},\mathsf{T})$, $\mathcal{B}$ acts as follows.

  (1) Choose $\hat{r}_{1,\theta}\xleftarrow{\$}\mathbf{Z}_p$ and $\hat{r}_{2,\theta}\xleftarrow{\$}\mathbf{Z}_p$.

  (2) If no tuple in list L_{H_1} related to $(g^{\hat{r}_{1,\theta}},\hat{\mathsf{ID}})$, then $\mathcal{B}$ chooses $z_{1,\theta}\xleftarrow{\$}\mathbb{Z}_p$ and adds it to list L_{H_1}. If no tuple in list L_{H_2} related to $(g^{\hat{r}_{2,\theta}},\hat{\mathsf{T}})$, then $\mathcal{B}$ chooses $z_{2,\theta}\xleftarrow{\$}\mathbb{Z}_p$ and adds it to list L_{H_2}.

  (3) Choose $t_\theta\xleftarrow{\$}\mathbb{Z}_p$ and $e\xleftarrow{\$}\mathbb{Z}_p$, set $A=g^{t_\theta}/(g^\alpha)^{z_{2,\theta}e}$ and $b_\theta=t_\theta+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+x_3)e$, add $((m,A,\hat{\mathsf{ID}},\hat{\mathsf{T}}),e)$ to list L_{H_3}.

  (4) Return $\sigma_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}=(A,\{b_\theta,g^{\hat{r}_{1,\theta}},g^{\hat{r}_{2,\theta}}\})$ to $\mathcal{F}$.

The correctness of the forged signature can be verified as follows.

$$\begin{aligned} g^{b_\theta} &= g^{t_\theta+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+x_3)e} \\ &= g^{t_\theta-\alpha e z_{2,\theta}}\cdot g^{(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z_{2,\theta}+x_3)e} \\ &= A\cdot(g^{\hat{r}_{1,\theta}}\cdot g^{\hat{r}_{2,\theta}}\cdot g^{x_1z_{1,\theta}+\alpha z_{2,\theta}+x_3})^e \end{aligned}$$

In the event $\mathsf{E}_{1,(3,2)}$, algorithm $\mathcal{R}_1$ applies the general forking algorithm $\mathcal{F}^{\mathrm{GF}}$ to solve the DLP challenge [14]. If $\mathcal{F}^{\mathrm{GF}}$ is successful, $\mathcal{R}_1$ can get two valid forgeries $\hat{\sigma}^{(i)}_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}=(\hat{A},\{\hat{b}^{(i)}_\theta,\hat{R}^{(i)}_{1,\theta},\hat{R}^{(i)}_{2,\theta}\}_{i=0,1})$ with respect to $(\hat{\mathsf{ID}},\hat{\mathsf{T}},m)$, where $\theta\in\mathsf{Path}(\eta)\cap\mathsf{KUNode}(\mathbf{BT},\mathbf{RL},\mathsf{T})$. Next, we compute the discrete logarithm α as follows.

$$\begin{cases} b^{(0)}_\theta = \hat{a}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(0)}_{2,\theta}+x_3)e \\ b^{(1)}_\theta = \hat{a}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(1)}_{2,\theta}+x_3)e \end{cases} \Rightarrow$$

$$\alpha=b^{(1)}_\theta-b^{(0)}_\theta/(z^{(1)}_{2,\theta}-z^{(0)}_{2,\theta})e$$

The advantage that $\mathcal{R}_1$ can successfully solve the DLP is $\varepsilon\geq\epsilon^2/q^4(q_{H_3}+q_{H_2})^2$.

In the event $\mathsf{E}_{1,(2,3)}$, algorithm $\mathcal{R}_1$ applies the multiple-forking algorithm $\mathcal{F}^{\mathrm{MF}}$ to solve the DLP challenge [15]. If $\mathcal{F}^{\mathrm{MF}}$ is successful, $\mathcal{R}_1$ can get four valid forgeries $\hat{\sigma}^{(i)}_{\hat{\mathsf{ID}},\hat{\mathsf{T}}}=(\hat{A},\{\hat{b}^{(i)}_\theta,\hat{R}^{(i)}_{1,\theta},\hat{R}^{(i)}_{2,\theta}\}_{i=0,1,2,3})$ with respect to $(\hat{\mathsf{ID}},\hat{\mathsf{T}},m)$. We have

$$\begin{cases} b^{(0)}_\theta = \hat{a}^{(0)}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(0)}_{2,\theta}+x_3)e^{(0)} \\ b^{(1)}_\theta = \hat{a}^{(1)}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(1)}_{2,\theta}+x_3)e^{(1)} \\ b^{(2)}_\theta = \hat{a}^{(2)}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(2)}_{2,\theta}+x_3)e^{(2)} \\ b^{(3)}_\theta = \hat{a}^{(3)}+(\hat{r}_{1,\theta}+\hat{r}_{2,\theta}+x_1z_{1,\theta}+\alpha z^{(3)}_{2,\theta}+x_3)e^{(3)} \end{cases}$$

where $\hat{a}^{(0)}=\hat{a}^{(1)}$, $\hat{a}^{(2)}=\hat{a}^{(3)}$, $z^{(0)}_{2,\theta}=z^{(1)}_{2,\theta}$ and $z^{(2)}_{2,\theta}=z^{(3)}_{2,\theta}$. Thus, we can compute the discrete logarithm α as follows.

$$\alpha=\frac{(b^{(0)}_\theta-b^{(1)}_\theta)(e^{(2)}-e^{(3)})-(b^{(2)}_\theta-b^{(3)}_\theta)(e^{(0)}-e^{(1)})}{(z^{(0)}_{2,\theta}-z^{(2)}_{2,\theta})(e^{(0)}-e^{(1)})(e^{(2)}-e^{(3)})}$$

The advantage that $\mathcal{R}_1$ can successfully solve the DLP is $\varepsilon \geq \epsilon^2/q^4(q_{H_3} + q_{H_2})^6$

This ends the proof.

6 Conclusion

In this paper, we first refined the security model for revocable identity-based signature scheme by considering a realistic threat, called signing key exposure. Then, we proposed the first scalable revocable identity-based signature scheme without bilinear pairings in the new security model by combining the lightweight Galindo and Garcia's identity-based signature scheme with the complete tree method. Finally, we proved our proposed revocable identity-based signature scheme is existentially unforgeable against adaptively chosen message and identity attacks under the standard discrete logarithm assumption in the random oracle model.

Acknowledgments

This research is funded by Science and Technology Program of Guangzhou (Grant No. 201707010358) and Opening Project of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security (Grant No. AGK201707).

References

[1] A. Shamir, Identity-based Cryptosystems and Signature Schemes, *1984 4th Annual International Cryptology Conference (CRYPTO)*, Santa Barbara, California, USA, 1984, pp. 47-53.

[2] D. Boneh, M. Franklin, Identity-based Encryption from the Weil Pairing, *2001 21st Annual International Cryptology Conference (CRYPTO)*, Santa Barbara, California, USA, 2001, pp. 213-229.

[3] Y. Liu, Z. Y. Cheng, C. C. Chang, Z. J. Zhang, A Secure Dynamic Identity Based Remote User Authentication Scheme Using Secret Sharing, *Journal of Internet Technology*, Vol. 13, No. 3, May, 2012, pp. 463-469.

[4] P. F. Zheng, Tradeoffs in Certificate Revocation Schemes, ACM SIGCOMM Computer Communication Review, Vol. 33, No. 2, 2003, pp. 103-112.

[5] Y. M. Tseng, T. T. Tsai, Efficient Revocable ID-based Encryption with a Public Channel, *The Computer Journal*, Vol. 55, No. pp. 475-486, April, 2012, pp. 475-486.

[6] A. Boldyreva, V. Goyal, V. Kumar, Identity-based Encryption with Efficient Revocation, *2008 15th ACM Conference on Computer and Communications Security (CCS)*, Alexandria, Virginia, USA, 2008, pp. 417-426.

[7] B. Libert, D. Vergnaud, Adaptive-ID Secure Revocable Identity-based Encryption, *2009 Cryptographers' Track at the RSA Conference (CT-RSA)*, San Francisco, CA, USA, 2009, pp. 1-15.

[8] J. H. Seo, K. Emura, Revocable Identity-based Encryption Revisited: Security Model and Construction, *2013 16th International Conference on Practice and Theory in Public-Key Cryptography (PKC)*, Nara, Japan, 2013, pp. 216-234.

[9] C. J. Wang, Y. Li, X. N. Xia, K. J. Zheng, An Efficient and Provable Secure Revocable Identity-based Encryption Scheme, *Plos one*, Vol. 9, No. 9, September, 2014, pp. 1-11.

[10] T. Y. Wu, T. T. Tsai, Y. M. Tseng, Revocable ID-based Signature Scheme with Batch Verifications, *2012 8th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Piraeus, Greece, 2012, pp. 49-54.

[11] Y. X. Sun, F. T. Zhang, L. M. Shen, R. Deng, Revocable Identity-based Signature without Pairing, *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, Xi'an, China, 2013, pp. 363-365.

[12] D. Galindo, F. D. Garcia, A Schnorr-Like Lightweight Identity-based Signature Scheme, *2009 Second International Conference on Cryptology in Africa (AFRICACRYPT)*, Gammarth, Tunisia, 2009, pp. 135-148.

[13] L. Chen, Z. Cheng N. P. Smart, Identity-based Key Agreement Protocols from Pairings, *International Journal of Information Security*, Vol. 6, No. 4, January, 2007, pp. 213-241.

[14] M. Bellare, G. Neven, Multi-signatures in the Plain Public-key Model and a General Forking Lemma, *2006 13th ACM Conference on Computer and Communications Security (CCS)*, Alexandria, VA, USA, 2006, pp. 390-399.

[15] A. Boldyreva, A. Palacio, B. Warinschi, Secure Proxy Signature Schemes for Delegation of Signing Rights, *Journal of Cryptology*, Vol. 25, No. 1, January, 2012, pp. 57-115.

Biographies

Chang-Ji Wang, Ph.D., Professor of School of Information Science and Technology, Guangdong University of Foreign Studies. His main research areas include cryptography theory and applications, security and privacy in cloud computing, etc.

Hui Huang, Ph.D., Associate Professor, School of Statistics and Mathematics, Guangdong University of Finance and Economics. His main research areas include applied mathematics and pattern recognition.

Yuan Yuan, Ph.D., Associate Professor, School of Mathematics and Statistics, Guangdong University of Foreign Studies. Her main research areas include cryptography theory and applications.