An Improved Cloud-Based Revocable Identity-Based Proxy Re-encryption Scheme

Changji Wang[1,2](✉), Jian Fang[2,3], and Yuan Li[2,3]

[1] School of Software, Yunnan University, Kunming 650500, China
wchangji@gmail.com
[2] Guangdong Key Laboratory of Information Security Technology, Sun Yat-sen University, Guangzhou 510275, China
[3] School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510275, China

Abstract. Key revocation and ciphertext update are two prominent security requirements for identity-based encryption systems from a practical view. Several solutions to offer efficient key revocation or ciphertext update for identity-based encryption systems have been proposed in the literature. However, how to achieve both key revocation and ciphertext update functionalities simultaneously in identity-based encryption systems is still an open problem. Recently, Liang et al. introduce the notion of cloud-based revocable identity-based proxy re-encryption (CR-IB-PRE) scheme with the aim to achieve both ciphertext update and key revocation functionalities, and present a CR-IB-PRE scheme from bilinear pairings. In this paper, we first showed Liang et al.'s scheme has serious security pitfalls such as re-encryption key forgery and collusion attack, which lead to revoked users can decrypt any ciphertext regarding their identities at any time period. We then redefined the syntax and security model of CR-IB-PRE scheme and proposed an improved CR-IB-PRE scheme from bilinear pairings. The improved scheme not only achieves collusion resistance, but also takes lower decryption computation and achieves constant size re-encrypted ciphtertext. Finally, we proved the improved CR-IB-PRE scheme is adaptively secure in the standard model under DBDH assumption.

Keywords: Identity-based encryption · Proxy re-encryption · Key revocation · Ciphertext update · Cloud computing

1 Introduction

The concept of identity-based public key cryptography (ID-PKC) was originally introduced by Shamir [1] to avoid cumbersome certificate management. In an identity-based crypto-system, users do not need to pre-compute public key and private key pairs and obtain certificates for their public keys. Instead, users' identifiers information such as email addresses, telephone numbers or social security numbers can be used as users' public keys, while private keys are derived at any time by a trusted third party, called private key generator (PKG), upon request by the designated users. Since Boneh and Franklin [2] proposed the first practical and provable secure identity-based encryption (IBE) scheme in 2001, research on ID-PKC has become a hot topic in cryptography [3–6].

Revocation capability is indispensable to IBE systems from a practical point of view [2]. Suppose that a user Alice whose private key is compromised or stolen, or she has left the organization, the PKG should revoke Alice's private key in time to mitigate the damage that an adversary with Alice's compromised private key to access confidential data encrypted under her identity. Note that revocable IBE only assures that revoked users cannot decrypt ciphertexts generated after revocation, however, it cannot prevent a revoked user from accessing ciphertexts which were created before the revocation, since the old private key of the revoked user is enough to decrypt these ciphertexts. Thus, ciphertext update or re-encryption is necessary and crucial to IBE systems [7].

Several solutions to offer efficient revocation functionality or ciphertext update functionality for IBE systems have been proposed in the literatures [8–16]. However, how to achieve both key revocation and ciphertext update functionalities simultaneously in IBE systems is still an open problem. Recently, Liang et al. [17] introduce the notion of cloud-based revocable identity-based proxy re-encryption (CR-IB-PRE) scheme with the aim to achieve both ciphertext update and revocation functionalities for IBE systems. In a CR-IB-PRE scheme, ciphertexts are encrypted under a certain identity and time period and stored in the cloud. At the end of a given time period, the cloud service provider (CSP), acting as a semi-trust proxy, will re-encrypt all ciphertexts of the user under the current time period to the next time period, no matter a user is revoked or not. If a user Alice is revoked in the forthcoming time period, she cannot decrypt the ciphertexts by using her expired private key anymore.

In this paper, we first showed that Liang et al.'s scheme has serious security pitfalls such as re-encryption key forgery and collusion attack, which lead to revoked users can decrypt any ciphertext regarding their identities at any time period. Then, we refined the syntax definition and security model for CR-IB-PRE scheme. The refined syntax for CR-IB-PRE scheme is similar to that of self-updatable encryption scheme recently proposed by Lee [18], where the CSP can update stored ciphertexts without any interaction with data owners as long as the revocation event happens. In our refined security model for CR-IB-PRE scheme, an adversary can choose an original ciphertext or a re-encrypted ciphertext as the challenge ciphertext. In particular, we consider the decryption key exposure attack [10], which means an adversary can obtain long-term private keys and decryption keys corresponding to identities and some time periods of his choice. Next, we proposed an improved CR-IB-PRE scheme from bilinear pairings. The improved scheme not only achieves collusion resistance, but also takes lower decryption computation and achieves constant size re-encrypted ciphtertext. Finally, we proved the improved CR-IB-PRE scheme is adaptively secure in the standard model under DBDH assumption.

2 Preliminaries

We denote by $x \xleftarrow{\$} \mathbf{S}$ the operation of picking an element x uniformly at random from the set $\mathbf{S}$, by $\mathrm{Enc}(k, m)$ and $\mathrm{Dec}(k, c)$ the operation of encrypting and decrypting with respect to a semantically secure symmetric cipher Γ under the session key k, respectively.

A bilinear group generator $\mathcal{G}$ is an algorithm that takes as input a security parameter κ and outputs a bilinear group $(q, \mathbf{G}, \mathbf{G}_T, \hat{e}, g)$, where $\mathbf{G}$ and $\mathbf{G}_T$ are cyclic groups of prime order q, g is a generator of $\mathbf{G}$, and $\hat{e}$: $\mathbf{G} \times \mathbf{G} \rightarrow \mathbf{G}_T$ is a bilinear map with the following properties:

- Bilinearity: $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for $g_1, g_2 \xleftarrow{\$} \mathbf{G}$ and $a, b \xleftarrow{\$} \mathbf{Z}_q^*$.
- Non-degeneracy: There exists $g_1, g_2 \in \mathbf{G}$ such that $\hat{e}(g_1, g_2) \neq 1$.
- Computability: There is an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1, g_2 \in \mathbf{G}$.

The Decisional Bilinear Diffie-Hellman (DBDH) assumption in a prime order bilinear group $(q, \mathbf{G}, \mathbf{G}_T, \hat{e}, g)$ states that, given $(g, g^a, g^b, g^c, \hat{e}(g, g)^z)$, it is computationally intractable to determine whether $\hat{e}(g, g)^z = \hat{e}(g, g)^{abc}$, where $a, b, c, z \xleftarrow{\$} \mathbf{Z}_q^*$.

To achieve efficient revocation for IBE schemes, Boldyreva et al. [8] introduced the **KUNode** algorithm, which is described in Algorithm 1. Denote by `root` the root node of the tree $\mathbb{T}$, by $\mathrm{Path}(\eta)$ the set of nodes on the path from η to `root` for a leaf node η, by ζ_L and ζ_R the left and right child of a non-leaf node ζ, respectively. The **KUNode** algorithm determines the smallest subset $\mathbf{Y}$ of nodes that contains an ancestor of all leaves corresponding to non-revoked users at each time period.

Algorithm 1. KUNode Algorithm

```
1: Input ($\mathbb{T}, \mathbf{RL}, \mathsf{T}$);
2: For $\forall(\eta_i, \mathsf{T}_i) \in \mathbf{RL}$, $\mathbf{X}, \mathbf{Y} \leftarrow \emptyset$;
3: if $\mathsf{T}_i \leq \mathsf{T}$ then
4:     Add $\mathrm{Path}(\eta_i)$ to $\mathbf{X}$;
5: end if
6: $\forall x \in \mathbf{X}$;
7: if $x_L \notin \mathbf{X}$ then
8:     Add $x_L$ to $\mathbf{Y}$;
9: end if
10: if $x_R \notin \mathbf{X}$ then
11:     Add $x_R$ to $\mathbf{Y}$;
12: end if
13: if $\mathbf{Y} = \emptyset$ then
14:     Add root to $\mathbf{Y}$;
15: end if
16: Output $\mathbf{Y}$;
```

Upon registration, the PKG assigns a leaf node η of a complete binary tree $\mathbb{T}$ 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 Security Analysis of Liang et al.'s CR-IB-PRE Scheme

Denote Waters' identity hash function $F_{\mathrm{Wat}}(\mathsf{id}) = u_0 \prod_{i=1}^{n} u_i^{\mathsf{id}_i}$ [6], where $\mathsf{id} = \{\mathsf{id}_i\}_{i=1}^{n} \in \{0,1\}^n$, and $u_0, u_1, \cdots, u_n \in \mathbf{G}$. Denote Boneh and Boyen's hash function $F_{\mathrm{BB}}(T) = v_1 v_2^T$ [19], where $v_1, v_2 \in \mathbf{G}$. Liang et al.'s CR-IB-PRE scheme [17] is described as follows.

– **Setup**$(1^\kappa, N)$: The PKG generates $(q, \mathbf{G}, \mathbf{G}_T, \hat{e}, g)$, chooses $\gamma, \alpha, \hat{\alpha}, \beta \xleftarrow{\$} \mathbf{Z}_q^*$, $g_2, g_3, v_1, v_2, u_0, u_1, \ldots, u_n \xleftarrow{\$} \mathbf{G}$, two target collision resistant (TCR) hash functions $\mathrm{TCR}_1 : \mathbf{G} \rightarrow \mathbf{Z}_q^*$ and $\mathrm{TCR}_2 : \mathbf{G}_T \rightarrow \{0,1\}^\kappa$. Then, the PKG sets $g_1 = g^\alpha$, $v_0 = g^\gamma$, $\mathbf{RL} = \emptyset$ and $\mathbf{ST} = \mathbf{DB}$. Finally, the PKG publishes system parameter $mpk = (g, g_1, g_2, g_3, v_0, v_1, v_2, u_0, u_1, \cdots, u_n, \mathrm{TCR}_1, \mathrm{TCR}_2, \varGamma)$, and keeps the master secret key $msk = (\gamma, \hat{\alpha}, g_2^\alpha, g_3^\beta)$ secret.

– **KeyGen**(msk, id): The PKG chooses $r_{\mathsf{id}} \xleftarrow{\$} \mathbf{Z}_q^*$, computes $sk_{\mathsf{id}_1} = g_3^\beta F_{\mathrm{Wat}}(\mathsf{id})^{r_{\mathsf{id}}}$, $sk_{\mathsf{id}_2} = g^{r_{\mathsf{id}}}$ and $sk_{\mathsf{id}_3} = g_z^\gamma$. Then, the PKG sets additional public parameters $g_z = g^{\hat{\alpha}^z}$, $g_{z+1} = g^{\hat{\alpha}^{z+1}}$, $g_{\lambda+1-z} = g^{\hat{\alpha}^{\lambda+1-z}}$, $g_{\lambda+1+z} = g^{\hat{\alpha}^{\lambda+1+z}}$ for user id, where z is the index for identity id and $\lambda = N + 1$. Finally, the PKG sets the partial private key $sk_{\mathsf{id}} = (sk_{\mathsf{id}_1}, sk_{\mathsf{id}_2}, sk_{\mathsf{id}_3})$.

– **TokenUp**$(msk, \mathbf{id}, T_i, \mathbf{RL}, \mathbf{ST})$: The PKG chooses $r_{T_i}, \hat{t} \xleftarrow{\$} \mathbf{Z}_q^*$, $K \xleftarrow{\$} \mathbf{G}_T$, sets $E_{\tau_i}^{(1)} = (\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ and $E_{\tau_i}^{(2)} = \mathrm{Enc}(k, \tau_{i,1} \| \tau_{i,2})$, where i is the index for the time period, $\mathbf{id}$ is a set of identities, and

$$\mathcal{T}_1 = K \cdot \hat{e}(g_{\lambda+1}, g)^{\hat{t}}, \quad \mathcal{T}_2 = g^{\hat{t}}, \quad \mathcal{T}_3 = (v_0 \prod_{\omega \in \mathbf{id}} g_{\lambda+1-\omega})^{\hat{t}},$$

$$\tau_{i,1} = (g_2^\alpha / g_3^\beta) F_{\mathrm{BB}}(T_i)^{r_{T_i}}, \quad \tau_{i,2} = g^{r_{T_i}}, \quad k = \mathrm{TCR}_2(K)$$

Finally, the PKG uploads the token $\tau_i = (E_{\tau_i}^{(1)}, E_{\tau_i}^{(2)})$ for a set $\mathbf{id}$ of identities to the CSP.

– **DeKeyGen**$(sk_{\mathsf{id}}, \tau_i)$: A user id chooses $\tilde{r}, r_1, r_2 \xleftarrow{\$} \mathbf{Z}_q^*$, computes

$$K = \mathcal{T}_1 / (\hat{e}(\mathcal{T}_3, g_z) / \hat{e}(sk_{\mathsf{id}_3} \prod_{\omega \in \mathbf{id} \setminus \{z\}} g_{\lambda+1-\omega+z}, \mathcal{T}_2)), \quad k = \mathrm{TCR}_2(K)$$

$$\mathrm{Dec}(k, E_{\tau_i}^{(2)}) = (\tau_{i,1}, \tau_{i,2}), \quad \tau_{i,1} \leftarrow \tau_{i,1} F_{BB}(T_i)^{\tilde{r}}, \quad \tau_{i,2} \leftarrow \tau_{i,2} g^{\tilde{r}}$$

$$dk_{\mathsf{id}|i,1} = sk_{\mathsf{id}_1} \tau_{i,1} F_{\mathrm{Wat}}(\mathsf{id})^{r_1} F_{\mathrm{BB}}(T_i)^{r_2} = g_2^\alpha F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1} F_{\mathrm{BB}}(T_i)^{\hat{r}_2},$$

$$dk_{\mathsf{id}|i,2} = sk_{\mathsf{id}_2} g^{r_1} = g^{\hat{r}_1}, \quad dk_{\mathsf{id}|i,3} = \tau_{i,2} g^{r_2} = g^{\hat{r}_2},$$

where $\hat{r}_1 = r_{id} + r_1$ and $\hat{r}_2 = r_{T_i} + \tilde{r} + r_2$. Finally, the user sets the updated secret key $dk_{\mathsf{id}|i} = (dk_{\mathsf{id}|i,1}, dk_{\mathsf{id}|i,2}, dk_{\mathsf{id}|i,3})$ for identity id and time period T_i. Note that the user will share $r_1, r_2, \tilde{r}$ with the PKG such that the PKG can store $(\mathsf{id}|i, \hat{r}_1, \hat{r}_2)$ in a list $\mathbf{List}^{dk_{\mathsf{id}|i}}$ for further use.

- **ReKeyToken**$(msk, T_i, T_{i'})$: If a user with identity id is allowed to update his key to another time period $T_{i'}$, the PKG chooses $\xi \xleftarrow{\$} \mathbf{G}_T$, computes $\varphi^{(1)}_{i\to i'} = F_{\mathrm{BB}}(T_{i'})^{\mathrm{TCR}_1(\xi)}/F_{\mathrm{BB}}(T_i)^{\hat{r}_2}$, $\varphi^{(2)}_{i\to i'} = (\hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3) \leftarrow \mathbf{IBEnc}(\mathsf{id}, T_{i'}, \xi)$, where $\hat{r}_2$ is recovered from $(\mathsf{id}|i', \hat{r}_1, \hat{r}_2)$ which is stored the $\mathbf{List}^{dk_{\mathsf{id}|i}}$. Finally, the PKG sets the re-encryption key token $\varphi_{i\to i'} = (\varphi^{(1)}_{i\to i'}, \varphi^{(2)}_{i\to i'})$.
- **ReKey**$(dk_{\mathsf{id}|i}, \varphi_{i\to i'})$: After receiving $\varphi_{i\to i'}$ from the PKG, a user with identity id chooses $\rho \xleftarrow{\$} \mathbf{Z}_q^*$, sets $rk_1 = dk_{\mathsf{id}|i,1}\varphi^{(1)}_{i\to i'} F_{\mathrm{Wat}}(\mathsf{id})^{\rho}$, $rk_2 = dk_{\mathsf{id}|i,2}g^{\rho}$, and $rk_3 = \varphi^{(2)}_{i\to i'}$. Finally, the user outputs the re-encryption key $rk_{\mathsf{id}|i\to i'} = (rk_1, rk_2, rk_3)$.
- **IBEnc**(id, T_i, m): Given an identity id, a time period T_i, and a message $m \in \mathbf{G}_T$, a sender chooses $t \xleftarrow{\$} \mathbf{Z}_q^*$, computes $C_0 = m \cdot \hat{e}(g_1, g_2)^t$, $C_1 = g^t$, $C_2 = F_{\mathrm{Wat}}(\mathsf{id})^t$ and $C_3 = F_{\mathrm{BB}}(T_i)^t$. The sender then sets the ciphertext $C_{\mathsf{id}\|T_i} = (C_0, C_1, C_2, C_3)$. We assume that the identity id and the time period T_i are implicitly included in the ciphertext.
- **ReEnc**$(rk_{\mathsf{id}|i\to i'}, C_{\mathsf{id}\|T_i})$: The CSP first parses $C_{\mathsf{id}\|T_i} = (C_0, C_1, C_2, C_3)$ and $rk_{\mathsf{id}|i\to i'} = (rk_1, rk_2, rk_3)$, then sets the re-encrypted ciphertext $C_{\mathsf{id}\|T_{i'}} = (C_0, C_1, C_4, rk_3)$, where

$$C_4 = \frac{\hat{e}(C_1, rk_1)}{\hat{e}(C_2, rk_2)} = \hat{e}(g^t, g_2^{\alpha} F_{\mathrm{BB}}(T_{i'})^{\mathrm{TCR}_1(\xi)}),$$

Note if $C_{\mathsf{id}\|T_{i'}}$ needs to be further re-encrypted to the time period $T_{i''}$ with a given re-encrypt key $rk_{\mathsf{id}|i'\to i''} = (rk_1', rk_2', rk_3')$, the CSP first parses rk_3 as $(\hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3)$, then sets the ciphertext $C_{\mathsf{id}\|T_{i''}} = (C_0, C_1, C_4, \hat{C}_0, \hat{C}_1, \hat{C}_4, rk_3')$, where

$$C_4' = \frac{\hat{e}(\hat{C}_1, rk_1')}{\hat{e}(\hat{C}_2, rk_2')},$$

- **IBDec**$(dk_{\mathsf{id}|i}, C_{\mathsf{id}\|T_i})$: The decryptor responses as follows with respect to the following three cases:

  Case 1: For the original ciphertext $C_{\mathsf{id}\|T_i} = (C_0, C_1, C_2, C_3)$, the decryptor can recover message by computing

$$\frac{\hat{e}(C_1, dk_{\mathsf{id}|i,1})}{\hat{e}(C_2, dk_{\mathsf{id}|i,2})\hat{e}(C_3, dk_{\mathsf{id}|i,3})} = \hat{e}(g_1, g_2)^t \Rightarrow m = \frac{C_0}{\hat{e}(g_1, g_2)^t}$$

  Case 2: For the re-encrypted ciphertext $C_{\mathsf{id}\|T_i}$ and it is re-encrypted only once, i.e., $C_{\mathsf{id}\|T_i} = (C_0, C_1, C_4, rk_3 = (\hat{C}_0, \hat{C}_1, \hat{C}_2, \hat{C}_3))$, the decryptor recover message by computing

$$\hat{C}_0 \frac{\hat{e}(\hat{C}_2, dk_{\mathsf{id}|i,2}) \cdot \hat{e}(\hat{C}_3, dk_{\mathsf{id}|i,3})}{\hat{e}(C_1, dk_{\mathsf{id}|i,1})} = \xi \Rightarrow m = C_0 \frac{\hat{e}(C_1, F_{\mathrm{BB}}(T_i)^{\mathrm{TCR}_1(\xi)})}{C_4}.$$

Case 3: For the re-encrypted ciphertext $C_{\mathsf{id}\|T_i}$ and it is re-encrypted ℓ times from period T_1 to $T_{\ell+1}$. Denote by $C^{(\ell+1)} = (C_0^{(1)}, C_1^{(1)}, C_4^{(1)}, \ldots, C_0^{(\ell)}, C_1^{(\ell)}, C_4^{(\ell)}, rk^{(\ell+1)})$ the re-encrypted ciphertext, where $C_0^{(1)}$ and $C_1^{(1)}$ are the components of original ciphertext under (id, T_1). For $1 \leq i \leq \ell$, $r_3^{(i+1)} = (C_0^{(i+1)}, C_1^{(i+1)}, C_2^{(i+1)}, C_3^{(i+1)})$ is the ciphertext under (id, T_{i+1}). The decryptor first sets

$$C_0^{(\ell+1)} \frac{\hat{e}(C_2^{(\ell+1)}, dk_{\mathsf{id}|\ell+1,2})\hat{e}(C_3^{(\ell+1)}, dk_{\mathsf{id}|\ell+1,3})}{\hat{e}(C_1^{(\ell+1)}, dk_{\mathsf{id}|\ell+1,1})} = \tilde{m}^{(\ell)}.$$

Then the decryptor sets

$$C_0^{(i)} \frac{\hat{e}(C_1^{(1)}, F_{\mathrm{BB}}(T_{i+1})^{\mathrm{TCR}_1(\tilde{m}^{(i)})})}{C_4^{(i)}} = \tilde{m}^{(i-1)}, \text{ for } i = \ell, \ell-1, \cdots, 2$$

Finally, the decryptor recovers the message by computing

$$m = C_0^{(1)} \frac{\hat{e}(C_1^{(1)}, F_{\mathrm{BB}}(T_2)^{\mathrm{TCR}_1(\tilde{m}^{(1)})})}{C_4^{(1)}}.$$

– $\mathbf{Revoke}(\mathsf{id}, T_i, \mathbf{RL}, \mathbf{ST})$: The PKG updates the revocation list by $\mathbf{RL} \leftarrow \mathbf{RL} \cup \{\mathsf{id}, T_i\}$ and returns the updated revocation list.

Theorem 1. *A revoked user Alice can decrypt any ciphertext regarding her identity at any time period in Liang et al.'s CR-IB-PRE scheme.*

Proof. A revoked user Alice with identity id can decrypt any ciphertext regarding her identity at any time period as follows.

– *Re-encryption key forgery attack:* Suppose Alice was not revoked at the time period T_i, but she is revoked at the current time period $T_{i'}$. We denote Alice's decryption key at the time period T_i by

$$dk_{\mathsf{id}|i} = (dk_{\mathsf{id}|i,1}, dk_{\mathsf{id}|i,2}, dk_{\mathsf{id}|i,3}) = (g_2^{\alpha} F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1} F_{\mathrm{BB}}(T_i)^{\hat{r}_2}, g^{\hat{r}_1}, g^{\hat{r}_2})$$

Assume that there is an original ciphertext $C = (C_0, C_1, C_2, C_3)$, which is encrypted under (id, T_i). Alice chooses ϵ_R at random from the plaintext space, and sends the re-encryption key $rk_{\mathsf{id}|i \to i'}$ from T_i to $T_{i'}$ to the CSP, where

$$rk_{\mathsf{id}|i \to i'} = (rk_1, rk_2, rk_3),\ rk_1 = dk_{\mathsf{id}|i,1} = g_2^{\alpha} F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1} F_{\mathrm{BB}}(T_i)^{\hat{r}_2},$$
$$rk_2 = dk_{\mathsf{id}|i,2} = g^{\hat{r}_1},\ rk_3 = \mathbf{IBEnc}(\mathsf{id}, T_{i'}, \epsilon_R)$$

Upon receiving the re-encryption key from Alice, the CSP re-encrypts the original ciphertext C and obtains $C' = (C_0, C_1, C_4, rk_3)$, where

$$C_4 = \frac{\hat{e}(C_1, rk_1)}{\hat{e}(C_2, rk_2)} = \frac{\hat{e}(g^t, g_2^\alpha F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1} F_{\mathrm{BB}}(T_i)^{\hat{r}_2})}{\hat{e}(F_{\mathrm{Wat}}(\mathsf{id})^t, g^{\hat{r}_1})} = \hat{e}(g_1, g_2)^t \cdot \hat{e}(g^t, F_{\mathrm{BB}}(T_i)^{\hat{r}_2})$$

Note that Alice knows $C_3 = F_{\mathrm{BB}}(T_i)^t$ and $dk_{\mathsf{id}|i,3} = g^{\hat{r}_2}$. Thus, Alice can recover m by computing

$$\frac{C_4}{\hat{e}(C_3, dk_{\mathsf{id}|i,3})} = \frac{\hat{e}(g_1, g_2)^t \cdot \hat{e}(g^t, F_{\mathrm{BB}}(T_i)^{\hat{r}_2})}{\hat{e}(F_{\mathrm{BB}}(T_i)^t, g^{\hat{r}_2})} = \hat{e}(g_1, g_2)^t,\ m = \frac{C_0}{\hat{e}(g_1, g_2)^t}.$$

The re-encryption key forgery attack holds because the CSP cannot verify re-encryption key submitted by the user. A legal re-encryption key generated by Alice can be described as $rk_1 = g_2^\alpha F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1+\rho} F_{\mathrm{BB}}(T_{i'})^{\mathrm{TCR}_1(\epsilon)}$, $rk_2 = g^{\hat{r}_1+\rho}$, $rk_3 = \mathbf{IBEnc}(\mathsf{id}, T_{i'}, \epsilon)$, while a malicious re-encryption key generated by Alice can be described as $rk_1 = g_2^\alpha F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}_1} F_{\mathrm{BB}}(T_i)^{\hat{r}_2}$, $sk_2 = g^{\hat{r}_1}$, $rk_3 = \mathbf{IBEnc}(\mathsf{id}, T_{i'}, \epsilon_R)$ where ϵ_R is independent of ϵ. The CSP can not distinguish $\mathbf{IBEnc}(\mathsf{id}, T_{i'}, \epsilon_R)$ from $\mathbf{IBEnc}(\mathsf{id}, T_{i'}, \epsilon)$ because the underlying SE-RIBE scheme [10] is proved to be IND-CPA secure in the standard model.

– *Collusion attack:* Suppose Alice is a revoked user and Bob is a non-revoked user at the current time period T_i. The PKG generates and broadcasts the update token $\tau_i = (E_{\tau_i}^{(1)}, E_{\tau_i}^{(2)})$ corresponding to a set of identities $\mathbf{id}$ and time period T_i. Bob can perform the **DeKeyGen** algorithm and obtain $(\tau_{i,1}, \tau_{i,2})$ corresponding to T_i. If Bob colludes with Alice, he sends $(\tau_{i,1}, \tau_{i,2})$ to Alice. Then Alice performs the **DeKeyGen** algorithm as Bob does. Finally, Alice obtains her valid decryption key in the time period T_i. Thus, Alice can decrypt any ciphertext regarding her identity at any time period by using her valid decryption key. The collusion attack holds because the update token $\tau_i = (E_{\tau_i}^{(1)}, E_{\tau_i}^{(2)})$ corresponding to a set of identities $\mathbf{id}$ and time period T_i consists of two independent components, where $E_{\tau_i}^{(1)}$ only depends on $\mathbf{id}$ and $E_{\tau_i}^{(2)}$ only depends on T_i.

This ends the proof.

4 Syntax and Security Definition for CR-IB-PRE Scheme

Let **ID**, **T**, **M** and **C** be identity space, time space, plaintext space and ciphtertext space, respectively. A CR-IB-PRE scheme Π can be defined by the following eight polynomial-time algorithms:

– **Setup:** The probabilistic setup algorithm is run by the PKG. It inputs a security parameter κ and a maximal number of users N. It outputs the public system parameters mpk, the master key msk, an empty revocation list **RL** and initial state **ST**.

- **IBKeyGen:** The probabilistic identity-based private key generation algorithm is run by the PKG. It inputs the public parameters mpk, the master key msk, an identity $\mathsf{id} \in \mathbf{ID}$. It outputs the corresponding identity-based initial private key sk_{id} and an update state $\mathbf{ST}$.
- **TokenUp:** The probabilistic token update algorithm is run by the PKG. It inputs the public parameters mpk, the master key msk, the key update time period $T_i \in \mathbf{T}$, the current revocation list $\mathbf{RL}$ and state $\mathbf{ST}$. It outputs the key update token τ_i corresponding to the key update time period T_i.
- **DeKeyGen:** The probabilistic decryption key generation algorithm is run by a user. It inputs the public parameters mpk, the user's initial private key sk_{id}, and the key update token τ_i. It outputs decryption key $dk_{\mathsf{id}|i}$ for the user with identity id under time period T_i.
- **IBEnc:** The probabilistic identity-based encryption algorithm is run by a sender. It inputs the public parameters mpk, the receiver's identity $\mathsf{id} \in \mathbf{ID}$, the time period $T_i \in \mathbf{T}$ and a message $m \in \mathbf{M}$. It outputs an original ciphertext $C_{\mathsf{id}|i}$ under (id, T_i) which can be further re-encrypted.
- **ReEnc:** The probabilistic re-encryption algorithm is run by the CSP. It inputs the public parameters mpk, the receiver's identity $\mathsf{id} \in \mathbf{ID}$, an original ciphertext $C_{\mathsf{id}|i} \in \mathbf{C}$ or a re-encrypted ciphertext $C_{\mathsf{id}|i\to k} \in \mathbf{C}$ that is re-encrypted from the original ciphertext $C_{\mathsf{id}|i}$, and a time period T_j. It outputs a re-encrypted ciphertext $C_{\mathsf{id}|i\to j}$.
- **IBDec:** The deterministic identity-based decryption algorithm is run by a receiver. It inputs the public parameters mpk, an original ciphertext $C_{\mathsf{id}|i}$ or a re-encrypted ciphertext $C_{\mathsf{id}|i\to j}$, the receiver's decryption key $dk_{\mathsf{id}|i}$ for time period T_i or the receiver's decryption keys for time period T_i and T_j, i.e., $dk_{\mathsf{id}|i}$ and $dk_{\mathsf{id}|j}$. It outputs the message m if decryption keys are valid. Otherwise, it outputs a reject symbol $\perp$.
- **Revoke:** The deterministic revocation algorithm is run by the PKG. It inputs the public parameters mpk, a set id of identity to be revoked, the revocation time period T, the current revocation lists $\mathbf{RL}$ and state $\mathbf{ST}$. It outputs the updated revocation lists $\mathbf{RL}'$.

We define indistinguishability against adaptive chosen identity and plaintext attack (IND-ID-CPA) experiment for CR-IB-PRE scheme as follows.

$\mathrm{Exp}^{\text{IND-ID-CPA}}_{\Pi,\mathcal{A}}(1^\kappa, N)$.

$(mpk, msk, \mathbf{RL}, \mathbf{ST}) \leftarrow \mathbf{Setup}(1^\kappa, N)$.

$(m_0, m_1, \mathsf{id}^*, \boldsymbol{T}^*, \mathbf{ST}) \leftarrow \mathcal{A}^{\mathcal{O}}(\text{Find}, mpk)$ such that $|m_0| = |m_1|$,

where $\boldsymbol{T}^*$ is a time period vector of (T_{i^*}) or $(T_{i^*}, T_{j^*})$with $T_{j^*} > T_{i^*}$.

$b \overset{\$}{\leftarrow} \{0,1\}$.

If $\boldsymbol{T}^* = (T_{i^*})$, then $C^* \leftarrow \mathbf{IBEnc}(mpk, \mathsf{id}^*, T_{i^*}, m_b)$;

If $\boldsymbol{T}^* = (T_{i^*}, T_{j^*})$, then $C^* \leftarrow \mathbf{ReEnc}(mpk, \mathsf{id}^*, T_{j^*},$

$\mathbf{IBEnc}(mpk, \mathsf{id}^*, T_{i^*}, m_b))$,

$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\text{Guess}, C^*, \mathbf{ST})$.

Return 1 if $b' = b$ and 0 otherwise.

In the above experiment, $\mathcal{O}$ is a set of oracles defined as follows.

- *IBKeyGen Oracle:* For $\mathsf{id} \in \mathbf{I}$, it returns sk_{id} and update state $\mathbf{ST}$ by running $\mathbf{IBKeyGen}(mpk, msk, \mathsf{id}, \mathbf{ST}) \rightarrow (sk_{\mathsf{id}}, \mathbf{ST})$.
- *TokenUp Oracle:* For $T_i \in \mathbf{T}$, it returns update token τ_i by running $\mathbf{TokenUp}(mpk, msk, T_i, \mathbf{RL}, \mathbf{ST}) \rightarrow \tau_i$.
- *DKeyGen Oracle:* For $\mathsf{id} \in \mathbf{I}$ and $T_i \in \mathbf{T}$, it returns decryption key $dk_{\mathsf{id}|i}$ under (id, T_i) by running $\mathbf{DeKeyGen}(mpk, sk_{\mathsf{id}}, \tau_i, \mathbf{ST}) \rightarrow dk_{\mathsf{id}|i}$.
- *ReEnc Oracle:* For an original ciphertext $C_{\mathsf{id}|i} \in \mathbf{C}$, $\mathsf{id} \in \mathbf{I}$ and $T_j \in \mathbf{T}$ with $T_j > T_i$, it returns a re-encrypted ciphertext $C_{\mathsf{id}|i\rightarrow j}$ of $C_{\mathsf{id}|i}$ by running $\mathbf{ReEnc}(mpk, \mathsf{id}, T_j, C_{\mathsf{id}|i})$. For a re-encrypted ciphertext $C_{\mathsf{id}|i\rightarrow k} \in \mathbf{C}$, $\mathsf{id} \in \mathbf{I}$ and $T_j \in \mathbf{T}$ with $T_j > T_k > T_i$, it returns a re-encrypted ciphertext $C_{\mathsf{id}|i\rightarrow j}$ of $C_{\mathsf{id}|i\rightarrow k}$ by running $\mathbf{ReEnc}(mpk, \mathsf{id}, T_j, C_{\mathsf{id}|i\rightarrow k})$.
- *Revoke Oracle:* For $\mathsf{id} \in \mathbf{I}$ and $T_i \in \mathbf{T}$, it returns an updated revocation list $\mathbf{RL}'$ by running $\mathbf{Revoke}(mpk, \mathsf{id}, T_i, \mathbf{RL}, \mathbf{ST}) \rightarrow \mathbf{RL}'$.

The above oracles can be queried by $\mathcal{A}$ with the following restrictions:

- $\mathcal{A}$ is only allowed to query *TokenUp Oracle* and *Revoke Oracle* in non-decreasing order of time.
- $\mathcal{A}$ is not allowed to query *Revoke Oracle* on time T_i if *TokenUp Oracle* was queried on T_i.
- $\mathcal{A}$ is not allowed to query *DeKeyGen Oracle* on time T_i before *TokenUp Oracle* was queried on T_i.
- For $\mathcal{A}$'s queries corresponding to vector of challenge time period $\boldsymbol{T}^* = (T_{i^*})$ or $\boldsymbol{T}^* = (T_{i^*}, T_{j^*})$, $\mathbf{DeKeyGen}(\mathsf{id}^*, T_{i^*})$ cannot be queried; If $\mathbf{IBKeyGen}(\mathsf{id}^*)$ was queried, then $\mathbf{Revoke}(\mathsf{id}^*, T_i)$ must be queried for $T_i \leq T_{i^*}$.

A CR-IB-PRE scheme is said to be IND-ID-CPA if for any PPT adversary $\mathcal{A}$, the following advantage is negligible in the security parameter κ.

$$\mathrm{Adv}_{\Pi,\mathcal{A}}^{\text{IND-ID-CPA}}(1^\kappa, N) = \left| \Pr[\mathrm{Exp}_{\Pi,\mathcal{A}}^{\text{IND-ID-CPA}}(1^\kappa, N) = 1] - \frac{1}{2} \right|.$$

5 Our Improved CR-IB-PRE Scheme

Our improved CR-IB-PRE scheme is described as follows.

- $\mathbf{Setup}(1^\kappa, N)$: The PKG generates $(q, \mathbf{G}, \mathbf{G}_T, \hat{e}, g)$, chooses $\alpha \xleftarrow{\$} \mathbf{Z}_q^*$ and $g_2, u_0, u_1, \cdots, u_n, v_0, v \xleftarrow{\$} \mathbf{G}$, sets $g_1 = g^\alpha$, $\mathbf{RL} = \emptyset$ and $\mathbf{ST} = \mathbb{T}$, where $\mathbb{T}$ is a binary tree. Finally, the PKG publishes $mpk = \{g, g_1, g_2, u_0, u_1, \cdots, u_n, v_0, v\}$, while keeps $msk = \{g_2^\alpha\}$ secret.
- $\mathbf{IBKeyGen}(mpk, msk, \mathsf{id}, \mathbf{ST})$: The PKG chooses an unassigned leaf node η from $\mathbb{T}$, stores id in the node η. For each node $\theta \in \mathrm{Path}(\eta)$, the PKG performs as follows.
  1. Recall g_θ if it is defined. Otherwise, $g_\theta \xleftarrow{\$} \mathbf{G}$ and store $(g_\theta, \tilde{g}_\theta = g_2/g_\theta)$ in node θ.

  2. Choose $r_\theta \xleftarrow{\$} \mathbf{Z}_q^*$.
  3. Compute $(D_{\theta,0}, D_{\theta,1}) = (g_\theta^\alpha F_{\mathrm{Wat}}(\mathsf{id})^{r_\theta}, g^{r_\theta})$.

  Finally, the PKG sends $sk_{\mathsf{id}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathrm{Path}(\eta)}$ back to the user.
- **TokenUp**$(mpk, msk, T_i, \mathbf{RL}, \mathbf{ST})$ The PKG parses $\mathbf{ST} = \mathbb{T}$. For each node $\theta \in \mathbf{KUNode}(\mathbb{T}, \mathbf{RL}, T_i)$, the PKG performs as follows.
  1. Retrieve $\tilde{g}_\theta$. Note that $\tilde{g}_\theta$ is always pre-defined in the **IBKeyGen** algorithm.
  2. Choose $s_\theta \xleftarrow{\$} \mathbf{Z}_q^*$.
  3. Compute $(\tilde{D}_{\theta,0}, \tilde{D}_{\theta,1}) = (\tilde{g}_\theta^\alpha F_{\mathrm{BB}}(T_i)^{s_\theta}, g^{s_\theta})$.

  Finally, the PKG returns $\tau_i = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathbf{KUNode}(\mathbb{BT}, \mathbf{RL}, T_i)}$.
- **DeKeyGen**$(mpk, sk_{\mathsf{id}}, \tau_i)$: The user first parses $sk_{\mathsf{id}} = \{(\theta, D_{\theta,0}, D_{\theta,1})\}_{\theta \in \mathbf{I}}$ and $\tau_i = \{(\theta, \tilde{D}_{\theta,0}, \tilde{D}_{\theta,1})\}_{\theta \in \mathbf{J}}$. If $\mathbf{I} \cap \mathbf{J} = \emptyset$, then returns $\perp$. Otherwise, the user chooses $\theta \in \mathbf{I} \cap \mathbf{J}$, $r, s \xleftarrow{\$} \mathbf{Z}_q$ and computes $dk_{\mathsf{id}|i,1} = D_{\theta,0} \cdot \tilde{D}_{\theta,0} \cdot F_{\mathrm{Wat}}(\mathsf{id})^r \cdot F_{\mathrm{BB}}(T_i)^s = g_2^\alpha \cdot F_{\mathrm{Wat}}(\mathsf{id})^{\hat{r}} \cdot F_{\mathrm{BB}}(T_i)^{\hat{s}}$, $dk_{\mathsf{id}|i,2} = D_{\theta,1} \cdot g^r = g^{\hat{r}}$ and $dk_{\mathsf{id}|i,3} = \tilde{D}_{\theta,1} \cdot g^s = g^{\hat{s}}$, where $\hat{r} = r_\theta + r, \hat{s} = s_\theta + s$. Finally, the user sets $dk_{\mathsf{id}|i} = (dk_{\mathsf{id}|i,1}, dk_{\mathsf{id}|i,2}, dk_{\mathsf{id}|i,3})$.
- **IBEnc**$(mpk, \mathsf{id}, T_i, m)$: The sender chooses $t \xleftarrow{\$} \mathbf{Z}_q^*$, computes

$$C_0 = m \cdot \hat{e}(g_1, g_2)^t, \quad C_1 = g^t, \quad C_2 = F_{\mathrm{Wat}}(\mathsf{id})^t, \quad C_3 = F_{\mathrm{BB}}(T_i)^t.$$

  Finally, the sender sets the original ciphertext $C_{\mathsf{id}|i} = (C_0, C_1, C_2, C_3)$.
- **ReEnc**$(mpk, \mathsf{id}, C, T_j)$: There are two cases according to C. If C is an original ciphertext, i.e., $C = C_{\mathsf{id}|i} = (C_0, C_1, C_2, C_3) = (C_{(0,0)}, C_{(0,1)}, C_{(0,2)}, C_{(0,3)})$, then the CSP chooses $t_1 \xleftarrow{\$} \mathbf{Z}_q^*$, and computes $C_{(1,0)} = C_{(0,0)} \cdot \hat{e}(g_1, g_2)^{t_1}$, $C_{(1,1)} = g^{t_1}$, $C_{(1,2)} = F_{\mathrm{Wat}}(\mathsf{id})^{t_1}$ and $C_{(1,3)} = F_{\mathrm{BB}}(T_j)^{t_1}$. Finally, the CSP sets the one time re-encrypted ciphertext $C_{\mathsf{id}|i \to j}$ associated with time period T_i and T_j as

$$C_{\mathsf{id}|i \to j} = (C_{(1,0)}, C_{(0,1)}, C_{(0,2)}, C_{(0,3)}, C_{(1,1)}, C_{(1,2)}, C_{(1,3)}).$$

  If C is an $\ell - 1$ times re-encrypted ciphertext, i.e.,

$$C = C_{\mathsf{id}|i \to k} = (C_{(\ell-1,0)}, C_{(0,1)}, C_{(0,2)}, C_{(0,3)}, C_{(\ell-1,1)}, C_{(\ell-1,2)}, C_{(\ell-1,3)}),$$

  then the CSP chooses $t_\ell \xleftarrow{\$} \mathbf{Z}_q^*$, and computes $C_{(\ell,0)} = C_{(\ell-1,0)} \cdot \hat{e}(g_1, g_2)^{t_\ell - t_{\ell-1}}$, $C_{(\ell,1)} = g^{t_\ell}$, $C_{(\ell,2)} = F_{\mathrm{Wat}}(\mathsf{id})^{t_\ell}$ and $C_{(\ell,3)} = F_{\mathrm{BB}}(T_j)^{t_\ell}$, where $\ell \geq 2$, $T_i < T_k < T_j$, and $t_{\ell-1}$ was chosen by the CSP at the time period T_k. Finally, the CSP returns the new re-encrypted ciphertext $C_{\mathsf{id}|i \to j}$ associated with time period T_i and T_j, where $= (C_{(\ell,0)}, C_{(0,1)}, C_{(0,2)}, C_{(0,3)}, C_{(\ell,1)}, C_{(\ell,2)}, C_{(\ell,3)})$.
- **IBDec**$(mpk, C, dk_{\mathsf{id}|i}, dk_{\mathsf{id}|j})$: Note that the current time period is T_j where $T_i \leq T_j$. $dk_{\mathsf{id}|i}$ and $dk_{\mathsf{id}|j}$ are the decryption keys of T_i and T_j, respectively. There are two cases according to C.
  1. If $C = C_{\mathsf{id}|i} = (C_0, C_1, C_2, C_3)$ is an original ciphertext, i.e., $T_i = T_j$ and $dk_{\mathsf{id}|i} = dk_{\mathsf{id}|j}$, the decryptor can recover the plaintext m by computing

$$C_0 \cdot \frac{\hat{e}(C_2, dk_{\mathsf{id}|i,2})\hat{e}(C_3, dk_{\mathsf{id}|i,3})}{\hat{e}(C_1, dk_{\mathsf{id}|i,1})} = m \cdot \hat{e}(g_1, g_2)^t \cdot \frac{1}{\hat{e}(g_1, g_2)^t} = m.$$

2. If $C = C_{\mathsf{id}|i \to j} = (C_{(\ell,0)}, C_{(0,1)}, C_{(0,2)}, C_{(0,3)}, C_{(\ell,1)}, C_{(\ell,2)}, C_{(\ell,3)})$ is re-encrypted ℓ times, i.e., $T_i < T_j$ and $\ell \geq 1$, the decryptor can recover the plaintext m by computing

$$C_{(\ell,0)} \cdot \frac{\hat{e}(C_{(\ell,2)}, dk_{\mathsf{id}|j,2})\hat{e}(C_{(\ell,3)}, dk_{\mathsf{id}|j,3})}{\hat{e}(C_{(\ell,1)}, dk_{\mathsf{id}|j,1})} \cdot \frac{\hat{e}(C_{(0,2)}, dk_{\mathsf{id}|i,2})\hat{e}(C_{(0,3)}, dk_{\mathsf{id}|i,3})}{\hat{e}(C_{(0,1)}, dk_{\mathsf{id}|i,1})}$$
$$= m \cdot \hat{e}(g_1, g_2)^{t+t_\ell} \cdot \frac{1}{\hat{e}(g_1, g_2)^{t_\ell} \cdot \hat{e}(g_1, g_2)^{t}} = m.$$

– **Revoke**$(mpk, \mathsf{id}, T_i, \mathbf{RL}, \mathbf{ST})$: The PKG updates the revocation list by $\mathbf{RL} \leftarrow \mathbf{RL} \cup \{(\mathsf{id}, T_i)\}$ and return the updated revocation list.

Table 1. Computation cost comparison

Schemes	Decryption cost	Storage cost	Update cost	RE-CT size
[17]	$O(\ell)t_{\hat{e}} + O(1)t_{\mathbf{G}_T}$	$O(p(\lvert\mathbf{Z}_q^*\rvert))$	$O(1)t_{\mathbf{G}} + O(1)t_{\mathbf{G}_T} + O(1)t_{\hat{e}}$	$(2\ell+1)\lvert\mathbf{G}_T\rvert + (\ell+3)\lvert\mathbf{G}\rvert$
Ours	$O(1)t_{\hat{e}} + O(1)t_{\mathbf{G}_T}$	$O(p(\lvert\mathbf{T}\rvert))$	$O(1)t_{\mathbf{G}} + O(1)t_{\mathbf{G}_T}$	$\lvert\mathbf{G}_T\rvert + 6\lvert\mathbf{G}\rvert$

6 Security and Efficiency Analysis

Theorem 2. *If there exists an adversary $\mathcal{A}$ attacking IND-ID-CPA security of the improved CR-IB-PRE scheme, then there exists another adversary $\mathcal{B}$ breaking IND-RID-CPA security of the SE-RIBE scheme.*

Proof. We will give the security proof in the extended version.

We compare our improved CR-IB-PRE scheme with Liang et al.'s CR-IB-PRE scheme in terms of the computation cost of re-encryption algorithm and decryption algorithm, and the size of re-encrypted ciphertext. The results are illustrated in Table 1, where ciphertext is re-encrypted ℓ times. We use big O notation and denote by $p(\cdot)$ some polynomial, $|\mathbf{T}|$ the bit-length of an element in time space $\mathbf{T}$, $|\mathbf{G}|$ the bit-length of an element in group $\mathbf{G}$, $|\mathbf{Z}_q^*|$ the bit-length of an element in $\mathbf{Z}_q^*$, and $|\mathbf{G}_T|$ the bit-length of an element in group $\mathbf{G}_T$, respectively. We denote by $t_{\hat{e}}$, $t_{\mathbf{G}}$ and $t_{\mathbf{G}_T}$ the computation cost of a bilinear pairing $\hat{e}(\mathbf{G}, \mathbf{G}) \rightarrow \mathbf{G}_T$, an exponentiation in group $\mathbf{G}$ and in group $\mathbf{G}_T$, respectively.

7 Conclusion

In this paper, we first showed that there are several security pitfalls in Liang et al.'s cloud-based revocable identity-based proxy re-encryption scheme. Then, we refined the syntax and security model for cloud-based revocable identity-based proxy re-encryption scheme. Finally, we proposed an improved cloud-based revocable identity-based proxy re-encryption scheme, which not only achieves collusion resistance, but also achieves constant size re-encrypted ciphtertext. It is interesting to construct a cloud-based revocable hierarchical identity-based proxy re-encryption scheme.

References

1. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
2. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. Baek, J., Newmarch, J., Safavi-naini, R., et al.: A survey of identity-based cryptography. In: Proceedings of Australian Unix Users Group Annual Conference, pp. 95–102 (2004)
4. Boneh, D., Canetti, R., Halevi, S., et al.: Chosen ciphertext security from identity-based encryption. SIAM J. Comput. **36**, 915–942 (2006)
5. Kang, L., Tang, X.H., Liu, J.F.: Tight chosen ciphertext attack (CCA)-secure hybrid encryption scheme with full public verifiability. Sci. China Inf. Sci. **57**, 112112(14) (2014)
6. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
7. Green, M., Ateniese, G.: Identity-based proxy re-encryption. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007)
8. Boldyreva, A., Goyal, V., Kumar, V.: Identity-based encryption with efficient revocation. In: Proceedings of the 15th ACM Conference on Computer and Communications Security, pp. 417–426. ACM, New York (2008)
9. Libert, B., Vergnaud, D.: Adaptive-ID secure revocable identity-based encryption. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 1–15. Springer, Heidelberg (2009)
10. Seo, J.H., Emura, K.: Revocable identity-based encryption revisited: security model and construction. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 216–234. Springer, Heidelberg (2013)
11. Wang, C.J., Li, Y., Xia, X.N., et al.: An efficient and provable secure revocable identity-based encryption scheme. PLoS ONE **9**(9), e106925 (2014). doi:10.1371/journal.pone.0106925
12. Chu, C.-K., Tzeng, W.-G.: Identity-based proxy re-encryption without random oracles. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 189–202. Springer, Heidelberg (2007)
13. Wang, L., Wang, L., Mambo, M., Okamoto, E.: New identity-based proxy re-encryption schemes to prevent collusion attacks. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 327–346. Springer, Heidelberg (2010)
14. Shao, J., Cao, Z.F.: Multi-use unidirectional identity-based proxy re-encryption from hierarchical identity-based encryption. Inf. Sci. **206**, 83–95 (2012)
15. Tang, Q., Hartel, P., Jonker, W.: Inter-domain identity-based proxy re-encryption. In: Yung, M., Liu, P., Lin, D. (eds.) Inscrypt 2008. LNCS, vol. 5487, pp. 332–347. Springer, Heidelberg (2009)
16. Luo, S., Shen, Q., Chen, Z.: Fully secure unidirectional identity-based proxy re-encryption. In: Kim, H. (ed.) ICISC 2011. LNCS, vol. 7259, pp. 109–126. Springer, Heidelberg (2012)
17. Liang, K., Liu, J.K., Wong, D.S., Susilo, W.: An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing. In: Kutyłowski, M., Vaidya, J. (eds.) ICAIS 2014, Part I. LNCS, vol. 8712, pp. 257–272. Springer, Heidelberg (2014)

18. Lee, K., Choi, S.G., Lee, D.H., Park, J.H., Yung, M.: Self-updatable encryption: time constrained access control with hidden attributes and better efficiency. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 235–254. Springer, Heidelberg (2013)
19. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)

Springer

http://www.springer.com/978-3-662-48682-5

Applications and Techniques in Information Security
6th International Conference, ATIS 2015, Beijing, China, November 4-6, 2015, Proceedings
Niu, W.; Li, G.; Liu, J.; Tan, J.; Guo, L.; Han, Z.; Batten, L. (Eds.)
2015, XVII, 398 p. 147 illus. in color., Softcover
ISBN: 978-3-662-48682-5