How to Construct Space Efficient Revocable IBE from Non-monotonic ABE

Huang Lin, Zhenfu Cao, Muxin Zhou, Haojin Zhu

Abstract. Since there always exists some users whose private keys are stolen or expired in practice, it is important for identity based encryption (IBE) system to provide a solution for revocation. The current most efficient revocable IBE system has a private key of size $O(\log n)$ and update information of size $O(r\log(\frac{n}{r}))$ where r is the number of revoked users. We describe a new revocable IBE systems where the private key only contains two group elements and the update information size is $O(r)$. To our best knowledge, the proposed constructions serve as the most efficient revocable IBE constructions in terms of space cost. Besides, this construction also provides a generic methodology to transform a non-monotonic attribute based encryption into a revocable IBE scheme. This paper also demonstrates how the proposed method can be employed to present an efficient revocable hierarchical IBE scheme.

Keywords. Revocable IBE, Non-monotonic, Attribute based encryption

1 Introduction

Revocation serves as a vital problem either in the setting of traditional public key encryption or identity based encryption (IBE) [13]. Since there are always some user whose private keys are either stolen or expired, the system has to provide some approaches to revoke these useless keys.

In traditional public key encryption system, there exists a public key infrastructure (PKI) who is responsible to publish some information to verify the mapping between user's identities and its public keys. The encryptor has to look up the public keys and the corresponding certificates of the receivers before the encryption is done. Therefore, the PKI might publish some information to inform the revocation list to the encryptor, which is how the revocation problem is usually solved in the traditional system[1, 7, 8]. However, this does not serve as a practical way in the identity based encryption setting. The primitive idea of IBE was proposed by Shamir [13] to eliminate the need for the verification of the mapping between the identity and the certificate. The encryptor only needs the public parameter and the identity of the receiver to complete encryption steps under IBE system and the whole encryption is done without any involvement of the private key generator (PKG). Therefore, this system does not provide any channel for the PKG to deliver the revocation list to the encryptor. When Boneh and Franklin proposed their first practical IBE solution, they also tried to provide a solution for this revocation problem. Their basic idea is to renew the private key of each user in the system periodically, e.g. each week, and the senders encrypt under the receiver's identities along with the current time period. However, it's easy to observe that this does not serve as an efficient solution since the workload of the PKG would be linearly dependent on the total number of the users in the system. The workload of the PKG would be unbearable when the user number n increases. After that, several revocable IBE constructions based on a trusted mediator [6, 5] were proposed. However, these revocable IBE systems are not very practical since the mediator has to involve in the decryption steps.

Recently, Boldyreva, Goyal, Kumar [3] proposed a revocable IBE system (BGK system) with improved efficiency. Compared with the original revocation system by Boneh and Franklin with $O(n-r)$ key update complexity, Boldyreva et al.'s key update information is of size $O(r\log(\frac{n}{r}))$ while the size of private key is $O(\log n)$.

The basic idea of their construction is to treat identity and time period as two independent attributes in fuzzy identity based encryption [12]. The message is encrypted under these two attributes. In order to successfully open the message, the receiver should not only have the correct identity, but also the right time period. The published key update information aims to bind the unrevoked identities with the current time period. In other words, only those unrevoked users can use these information for the current time period to update their private key. In order to reduce the workload of PKG, they adopt binary tree structure as their underlying tool. Therefore, although the key update information size is saved to $O(r\log(\frac{n}{r}))$, the private key size has to be $O(\log n)$.

Another related work is proposed by Libert and Vergnaud [9]. They provided a solution with a similar performance to that of Boldyreva et al.'s construction [3] while their construction is proven under an adaptive model.

1.1 Our Contribution

The above introduction shows that the update information size of the two most efficient revocable IBE constructions is $O(r\log(\frac{n}{r}))$, and the private key size of both constructions is $O(\log n)$. An important feature shared by these two systems is that the user has to take the key update information ku_t for time period t as input in order to generate the decryption key for the t-th time period. In other words, if a user ω at the t'-th time period wants to decrypt a ciphertext generated at a past time period, say the t''-th ($t'' < t'$) time period, the update information for the t''-th time period should be accessible to the user. Besides, there is no way to guarantee that each user of the system will update his(her) decryption key whenever the update information is published especially if the update information is published frequently. By implication, the system has to store the update information for all the time periods in the system lifetime and keep them publicly accessible. Although the space cost of the current revocable IBE schemes is low, the space cost might not be desirable when the update information or the decryption key pile up as time goes by. Hence, to further reduce the space cost of the revocable IBE scheme remains an important issue, which is one of the major motivation of this paper.

In this paper, we propose a new revocable IBE scheme from non-monotonic ABE. Our revocable construction has constant size private key. The update information size of the construction is $O(r)$ where r is the number of revoked users. The security of the construction is reduced to Decisional BDH assumption under selective-revocable-ID model as in [3]. To the best of our knowledge, the proposed construction is the most efficient revocable IBE scheme in terms of space cost. However, the decryption of our scheme is dominated by $O(r)$ group operations. The proposed constructions are especially suitable for the application scenarios with small revocation number, i.e., $r \ll n$. It fits into the applications where the space cost rather than the decryption efficiency is a major concern. The private storage requirement for each user is significantly reduced since each user only needs to hold two group elements in our proposed constructions. The storage requirement or transmission requirement is also reduced for the encryptor since the logarithmic factor is removed from the size of published information. Especially, they might fit into the same application scenario such as sensor networks mentioned in recently proposed broadcast encryption scheme [11] since our efficiency parameters are really close to that of [11]. Besides, our constructions are also fit into the systems where hybrid encryption or key encapsulation mechanism can be adopted since in this case the decryption efficiency is determined by the symmetric decryption algorithm after the message encryption key is generated from the decryption algorithm of revocable IBE scheme and therefore will not be a problem. Indeed the best revocation system is obtained by combining the BGK system with our system, i.e., using our system when r is really small and using BGK system when r grows to a certain level.

Compared with the existing efficient revocable IBE constructions [3, 9] using binary tree structure and fuzzy identity based encryption as their basic tool, this paper constructs a revocable IBE scheme from attribute based encryption without any involvement of binary tree structure. This is considered as one of our major novelties. Our construction provides a generic methodology to construct a revocable IBE scheme from non-monotonic attribute based encryption scheme. The proposed methodology applies to the existing two non-monotonic attribute based encryption schemes [10, 11]. We illustrate this method by applying our method to the non-monotonic attribute based encryption proposed by Ostrovsky [10]. Besides, we also show how our proposed methodology can be used to construct an efficient revocable hierarchical identity based encryption scheme.

1.2 Organization

The whole paper could be divided into the following sections: At Section 2, we'll first introduce the basic complexity assumption and the definition and security model for revocable IBE is also presented. In Section 3, the revocable IBE scheme is given while the corresponding generic methodology is introduced as the primitive idea of this construction. In Section 4, we show how to construct an efficient revocable hierarchical IBE scheme from the proposed method. At last, several conclusions are drawn.

2 Preliminaries

2.1 Decisional BDH assumption

The bilinear maps is crucial to our construction, some basic facts related to bilinear maps are introduced here.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_1$ and $\hat{e}$ be a bilinear map, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. The bilinear map $\hat{e}$ has the following properties:

1. Bilinearity: for all $u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
2. Non degeneracy: $\hat{e}(g, g) \neq 1$.

$\mathbb{G}_1$ is a bilinear group if the group operation in $\mathbb{G}_1$ and the bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ are both efficiently computable. Note that the map $\hat{e}$ is symmetric since $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab} = \hat{e}(g^b, g^a)$.

The security proof of the proposed scheme relies on DBDH assumption, the definition of DBDH assumption is shown as follow:

Definition 1. *Decisional Bilinear Diffie-Hellman (DBDH) Assumption . Let* z_1, z_2, z_3, $z \leftarrow \mathbb{Z}_p$ *be chosen randomly,* $\mathbb{G}_1$ *be a group of generator* g*. The Decisional BDH assumption is that no probabilistic polynomial time algorithm* $\mathcal{B}$ *can distinguish the tuple* $(g_1 = g^{z_1}, g_2 = g^{z_2}, g_3 = g^{z_3}, \hat{e}(g, g)^{z_1 z_2 z_3})$ *from the tuple* $(g_1 = g^{z_1}, g_2 = g^{z_2}, g_3 = g^{z_3}, \hat{e}(g, g)^{z})$ *with more than a negligible advantage. The advantage of* $\mathcal{A}$ *is*

$$|Pr[\mathcal{A}(g_1, g_2, g_3, \hat{e}(g, g)^{z_1 z_2 z_3}) = 1] - Pr[\mathcal{A}(g_1, g_2, g_3, \hat{e}(g, g)^{z})] = 1|$$

where the probability is taken over the random choice of the generator g*, the random choice of* a, b, c, z *in* $\mathbb{Z}_p$*, and the random bits consumed by* $\mathcal{B}$*.*

2.2 Definition and Security Model

Definition The definition of revocable IBE ($\mathcal{RIBE}$) is shown as follow: A revocable IBE scheme is defined by seven algorithms which have their associated message space **M**, identity space **I** and time space **T**. Key authority maintains a revocation identity list rl recording all the revoked identity set R and state st. In the following, an algorithm is called stateful only if it updates rl or st. The life time of the system is divided into several periods during which update information is published.

The stateful **setup** algorithm $\mathcal{S}$ (run by key authority) takes input security parameter 1^{κ} and number of users n, and outputs public parameters PK, master key MK, revocation list rl (initially empty) and state st.
The stateful **private key generation** algorithm $\mathcal{SK}$ (run by key authority) takes input public parameters PK, master key MK, identity $\omega \in \mathbf{I}$ and state st, and outputs private key SK_{ω} and an updated state st.
The **key update** generation algorithm $\mathcal{KU}$ (run by key authority) takes input public parameters PK, master key MK, key update time $t \in \mathbf{T}$, revocation list rl and state st, and outputs key update KU_t.
The **encryption** algorithm $\mathcal{E}$ (run by sender) takes input public parameters PK, identity $\omega \in \mathbf{I}$, encryption time $t \in \mathbf{T}$ and message $m \in \mathbf{M}$, and outputs ciphertext c.
The deterministic **decryption** algorithm $\mathcal{D}$ (run by receiver) takes input private key SK_{ω}, key update information KU_t and ciphertext c, and outputs a message $m \in \mathbf{M}$ or, a special symbol $\perp$ indicating that the ciphertext is invalid.
The stateful **revocation** algorithm $\mathcal{R}$ (run by key authority) takes input identity to be revoked $\omega \in \mathbf{I}$, revocation time $t \in \mathbf{T}$, revocation list rl and state st, and outputs an updated revocation list rl.

The consistency condition requires that for all $\kappa \in \mathbb{N}$ and polynomials (in κ) n, all PK and MK output by setup algorithm $\mathcal{S}$, all $m \in \mathbf{M}$, $\omega \in \mathbf{I}$, $t \in \mathbf{T}$ and all possible valid states st and revocation lists1 rl, if identity ω was not revoked before or, at time t then the following experiment returns 1 with probability 1: $(SK_{\omega}, st) \overset{r}{\leftarrow} \mathcal{SK}(PK, MK, \omega, st)$; $KU_t \overset{r}{\leftarrow} \mathcal{KU}(PK, MK, t, rl, st)$, $c \overset{r}{\leftarrow} \mathcal{E}(PK, \omega, t, m)$; If $\mathcal{D}(SK_{\omega}, KU_t, c) = m$, then return 1, else return 0.

Security model In the following game, we define the selective-revocable-ID security for revocable IBE schemes. The security model imitates the definition of the selective-ID security for traditional IBE scheme while taking into account possible revocation. In the beginning of the game, the adversary declares the challenge time and identity. The adversary is able to revoke users of its choices (including the challenge identity) at any period of time and see all the key update information. The adversary is also allowed to see the private key of users including the challenge identity but when it was revoked prior or at the challenge time.

In the following we only restrict in the security against the chosen plaintext attack and the definition for chosen-ciphertext attack can be found in [3], which is neglected in this paper.

The adversary first outputs the challenge identity and time, and also some information *state* it wants to preserve. Later it's given access to three oracles that correspond to the algorithms of the scheme. The oracles share state. We define them as follow:

- The *private key generation* oracle $\mathcal{SK}(\cdot)$ takes input identity ω and runs $\mathcal{SK}(pk, mk, \omega, st)$ to return private key sk_ω.
- The *key update* oracle $\mathcal{KU}(\cdot)$ takes input time t and runs $\mathcal{KU}(pk, mk, t, rl, st)$ to return key update ku_t.
- The *revocation* oracle $\mathcal{R}(\cdot, \cdot)$ takes input identity ω and time t and runs $\mathcal{R}(\omega, t, rl, st)$ to update rl.

The following two restrictions about the above oracles must hold: first, $\mathcal{KU}(\cdot)$ and $\mathcal{R}(\cdot, \cdot)$ can be queried on time which is greater than or equal to the time of all previous queries i.e. the adversary is allowed to query only in non-decreasing order of time. Also, the oracle $\mathcal{R}(\cdot, \cdot)$ can't be queried on time t if $\mathcal{KU}(\cdot)$ was queried on t. Second, if $\mathcal{SK}(\cdot)$ was queried on identity ω^* then $\mathcal{R}(\cdot, \cdot)$ must be queried on ω^*, t for any $t \leq t^*$.

For adversary $\mathcal{A}$ and number of users n define the following experiments:

- **Init** The adversary declares the challenge identity ω^* and time t^*, that he wishes to be challenged upon.
- **Setup** The challenger runs the $Setup$ algorithm and gives the public parameters to the adversary.
- **Phase 1** The adversary is allowed issue the aforementioned three oracles *private key generation* oracle, *key update* oracle, and *revocation* oracle.
- **Challenge** The adversary submits two equal length message $m_0, m_1 \in \mathcal{M}$. The challenger flips a coin $b \in \{0, 1\}$, and runs $\mathcal{E}(pk, \omega^*, t^*, m_b)$ to generate the challenge ciphertext c^* and pass it to the adversary.
- **Phase 2** Phase 1 is repeated
- **Guess** The adversary outputs a guess b' of b.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $Pr[b' = b] - \frac{1}{2}$.

Definition 2. *The revocable IBE scheme is said to be sRID-CPA secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

3 $\mathcal{RIBE}$ from Non-monotonic ABE

3.1 A Generic Transformation from Non-monotonic ABE to Revocable IBE

The BGK system can realize logarithmic efficiency since the private key assignment is based on a binary tree. However, the binary tree manipulation also results in a private key of size $O(\log n)$ and a $O(\log n/r)$ factor in the key update information, and this seems to be inevitable under the current binary-tree-based framework [9, 3]. In order to realize a revocable IBE scheme with constant size private key and $O(r)$ size key update information, we have to emancipate the construction from the restriction of binary tree. Our basic thought is to implement a revocable IBE scheme from a non-monotonic ABE scheme. We note that although the NOT gate of non-monotonic ABE seems naturally gives capability to exclude some users, there is no straightforward to realize this thought.

The underlying idea of BGK construction is each user fetches its private key for its identity ω at first, and then the periodically published update information will represent which user identities are revoked at the t-th time period. In other words, the update information can be considered as representing a predicate $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}}$ if each of the identity set $\{\omega^{(i)}\}_{i=1}^{r}$ is revoked at time period t. Here, $\overline{\omega^{(i)}}$ denotes the negation of $\omega^{(i)}$.

For any user ω, it is possible to control the user's decryption ability at time period t if the system can enforce him(her) to use a private key corresponding to a predicate $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}} \bigwedge \omega$ to decrypt a ciphertext. We can encrypt a message under an attribute set $\{\omega, t\}$. If a user ω is revoked at time t, then there must exist an $\omega^{(j)} \in \{\omega^{(i)}\}_{i=1}^{r}$ such that $\omega = \omega^{(j)}$. In this case, this specific user will be forced to use a private key for a predicate $t \bigwedge_{i=1}^{j-1} \overline{\omega^{(i)}} \bigwedge \overline{\omega} \bigwedge_{i=j+1}^{r} \overline{\omega^{(i)}} \bigwedge \omega$ to decrypt the ciphertext. A key policy non-monotonic attribute based encryption(ABE) scheme (see [10] for the concrete definition of key policy non-monotonic ABE) can be used to make sure that the decryption fails because the attribute set $\{\omega, t\}$ does not satisfy the predicate $t \bigwedge_{j=1}^{i-1} \overline{\omega^{(j)}} \bigwedge \overline{\omega} \bigwedge_{j=i+1}^{r} \overline{\omega^{(j)}} \bigwedge \omega$. The decryption is also guaranteed to be successful if the attribute set $\{\omega, t\}$ satisfies the respective predicate $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}} \bigwedge \omega$ when ω is unrevoked at time t and hence is not in the revoked set $\{\omega^{(i)}\}_{i=1}^{r}$.

The tricky problem left is how we could force a user ω to use a private key corresponding to such a predicate, i.e., $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}} \bigwedge \omega$ in the decryption steps. In our construction, we divide the system master key $MK = \alpha$ into two parts $MK_1 = \lambda$ and $MK_2 = \alpha - \lambda$, i.e., $MK = MK_1 + MK_2$. MK_1 is used to distribute private key for user identity ω. MK_2 is used to generate update information, i.e., a private key corresponding to a predicate $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}}$. A message will be encrypted under a public key corresponding to the system master key MK, and hence the decryptor has to use both private key and key update information in order to successfully open a message. We note that collusion attack is impossible since all the unrevoked users hold information on the same partial master key MK_1 in their own private keys. Therefore, intuitively, nothing can be done by them to help each other obtain any useful information on the system master key α.

We note that our proposed transformation method from non-monotonic ABE is almost fully generic except the fact that we require the two sub master keys for private key distribution system and key update system can be combined as one system master key. Both of the two recently proposed non-monotonic ABE schemes [10, 14] can be used as our underlying tool to present a secure revocable IBE scheme. We use OSW non-monotonic ABE [10] scheme as an example to show how our transformation method works.

3.2 Revocable IBE from OSW Non-monotonic ABE

Let $\mathbb{G}_1$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_1$. In addition, let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ denote the bilinear map. A security parameter κ will determine the size of the groups. We also define the Lagrange coefficient $\Delta_{i,J}$ for $i \in \mathbb{Z}_p$ and a set J of elements in $\mathbb{Z}_p$: $\Delta_{i,J}(x) = \prod_{j \in J, j \neq i} \frac{x-j}{i-j}$.

Setup $\mathcal{S}(1^\kappa, n)$ Choose α, β uniformly at random from $\mathbb{Z}_p^*$. Set $g_1 = g^\alpha$ and $g_2 = g^\beta$. Choose two polynomials $h(x)$ and $q(x)$ of degree two at random subject to the constraint that $q(0) = \beta$.

The public parameters PK contains $(g, g_1, g_2 = g^{q(0)}, g^{q(1)}, g^{q(2)}; g^{h(0)}, g^{h(1)}, g^{h(2)})$. The master key is $MK = \alpha$. The public parameters define two publicly computable functions $T, V : \mathbb{Z}_p \rightarrow \mathbb{G}$. The function $T(x) = g_2^{x^2} \cdot g^{h(x)}$ and $V(x) = g^{q(x)}$. Besides, choose λ uniformly at random from $\mathbb{Z}_p$ and set $MK_1 = \lambda$ and $MK_2 = \alpha - \lambda$.

Private Key Generation $\mathcal{SK}(MK_1,\ \omega,\ PK)$ Choose ρ_ω uniformly at random from $\mathbb{Z}_p$. The algorithm outputs a private key for identity ω as

$$SK_\omega = \left(D_\omega^{(1)}, D_\omega^{(2)}\right) = \left(g_2^{MK_1} \cdot T(\omega)^{\rho_\omega}, g^{\rho_\omega}\right) = \left(g_2^{\lambda} \cdot T(\omega)^{\rho_\omega}, g^{\rho_\omega}\right)$$

Key Update Generation $\mathcal{KU}(MK_2,\ PK,\ t,\ rl,\ st)$ This algorithm outputs a private key corresponding to the access structure $t \bigwedge_{i=1}^{r} \overline{\omega^{(i)}}$. For each revoked user $\omega^{(i)}, i \in [1, r]$, select two random values $\lambda_i, \rho_i \in (\mathbb{Z}_p)^2$, publish the respective update information as

$$D_i = (D_i^{(3)}, D_i^{(4)}, D_i^{(5)}) = (g_2^{\lambda_i+\rho_i}, V(\omega^{(i)})^{\rho_i}, g^{\rho_i})$$

For the update time t, select $\rho_t \overset{r}{\leftarrow} \mathbb{Z}_p$, publish the update information as

$$D_t = (D_t^{(1)}, D_t^{(2)}) = (g_2^{MK_2-\sum_{i=1}^r \lambda_i} \cdot T(t)^{\rho_t}, g^{\rho_t}) = (g_2^{\alpha-\lambda-\sum_{i=1}^r \lambda_i} \cdot T(t)^{\rho_t}, g^{\rho_t})$$

Return the update information as $KU_t = \left(\{(\omega^{(i)}, D_i)\}_{i=1}^r, D_t\right)$.

Encryption $\mathcal{E}(PK, \omega, t, M)$ Choose a random s from $\mathbb{Z}_p$. Generate the ciphertext as $C = \left(E^{(1)} = M\hat{e}(g_1,\ g_2)^s, E^{(2)} = g^s, E_\omega^{(3)} = T(\omega)^s, E_t^{(3)} = T(t)^s, E_\omega^{(4)} = V(\omega)^s,\right.$ $\left.E_t^{(4)} = V(t)^s\right)$

Decryption $\mathcal{D}(SK_\omega, KU_t, PK, C)$ If ω is not revoked at time t, then we have $\omega \notin \{\omega^{(i)}\}_{i=1}^r$. Using $D_\omega^{(1)}$ and $D_\omega^{(2)}$ from the secrete key SK_ω of ω, compute the partial decryption as

$$Z_\omega = \hat{e}(D_\omega^{(1)},\ E^{(2)})/\hat{e}(D_\omega^{(2)},\ E_\omega^{(3)}) = \hat{e}(g_2^\lambda \cdot T(\omega)^{\rho_\omega},\ g^s)/\hat{e}(g^{\rho_\omega},\ T(\omega)^s) = \hat{e}(g_2,\ g)^{s\lambda}$$

Using $D_t^{(1)}$ and $D_t^{(2)}$ from the update information D_t corresponding to update time t, compute the partial decryption for update time t as

$$Z_t = \frac{\hat{e}(D_t^{(1)},\ E^{(2)})}{\hat{e}(D_t^{(2)},\ E_t^{(3)})} = \frac{\hat{e}(g_2^{\alpha-\lambda-\sum_{i=1}^r \lambda_i} \cdot T(t)^{\rho_t},\ g^s)}{\hat{e}(g^{\rho_t},\ T(t)^s)} = \hat{e}(g_2,\ g)^{s(\alpha-\lambda-\sum_{i=1}^r \lambda_i)}$$

For each revoked user $\omega^{(i)}, i \in [1, r]$, compute Lagrangian coefficients $\{\sigma_x\}$ $_{x\in\{\omega,t,\omega^{(i)}\}}$ such that $\sum_{x\in\{\omega,t,\omega^{(i)}\}} \sigma_x q(x) = q(0) = \beta$. Using the update information $D_i^{(3)}$, $D_i^{(4)}$, $D_i^{(5)}$, compute the corresponding partial decryption as

$$Z_i = \frac{\hat{e}(D_i^{(3)},\ E^{(2)})}{\hat{e}\left(D_i^{(5)},\ \prod_{x\in\{\omega,t\}}\left(E_x^{(4)}\right)^{\sigma_x}\right)\cdot\hat{e}\left(D_i^{(4)},\ E^{(2)}\right)^{\sigma_{\omega^{(i)}}}} = \frac{\hat{e}(g_2^{\lambda_i+\rho_i},\ g^s)}{\hat{e}(g^{\rho_i},\ V(\omega)^{s\sigma_\omega} V(t)^{s\sigma_t})\cdot\hat{e}(V(\omega^{(i)})^{\rho_i},\ g^s)^{\sigma_{\omega^{(i)}}}}$$
$$= \frac{\hat{e}(g_2^{\lambda_i+\rho_i},\ g^s)}{\hat{e}(g^{\rho_i},\ g^{s[\sigma_\omega q(\omega)+\sigma_t q(t)]})\cdot\hat{e}(g^{\rho_i \sigma_{\omega^{(i)}} q(\omega^{(i)})},\ g^s)} = \frac{\hat{e}(g_2^{\lambda_i+\rho_i},\ g^s)}{\hat{e}(g^{\rho_i},\ g^{sq(0)})} = \hat{e}(g_2^{\lambda_i},\ g^s)$$

Finally, compute

$$\hat{e}(g_2,\ g)^{s\lambda} \cdot \hat{e}(g_2,\ g)^{s(\alpha-\lambda-\sum_{i=1}^r \lambda_i)} \cdot \prod_{i=1}^r \hat{e}(g_2,\ g)^{s\lambda_i} = \hat{e}(g_2,\ g)^{s\alpha} = \hat{e}(g_2,\ g_1)^s$$

and $\frac{E^{(1)}}{\hat{e}(g_2,\ g_1)^s} = M$.

In the above construction, the private key only contains two group elements while the update information contains $3r + 2$ group elements. This justifies our efficiency claims in the introduction.

The security of this construction can be stated as the following theorem. The proof of this theorem could be found in Appendix A.

Theorem 1. *If an adversary can break the proposed scheme in the sRID model, then a simulator can be constructed to play the Decisional BDH game with a non-negligible advantage.*

4 Extension: Revocable Hierarchical IBE from Non-monotonic ABE

To construct a revocable hierarchical IBE scheme has been considered as an interesting open problem in [9]. In the following section, we'll show how our proposed transformation method can be used to construct an efficient revocable hierarchical IBE (HIBE) scheme. The underlying HIBE is due to Boneh and Boyen [4]. Similar to [4], we assume the identity ω of depth ℓ are vectors of elements in $\mathbb{Z}_p^\ell$, and also write $\omega = (\omega_1, \cdots, \omega_\ell) \in \mathbb{Z}_p^\ell$. The j-th component corresponds to the identity at level j. The j-bit prefix of ω is denoted by $\omega|_j$=$(\omega_1, \cdots, \omega_j)$. The basic idea of revocable HIBE is that if an identity ω is revoked, then all its subordinate identities should be revoked automatically. In other words, if $\omega = (\omega_1, \cdots, \omega_\ell)$ is revoked, then all the identities including a prefix of ω will be revoked immediately. Instead of explicitly associate all the revoked identities of the hierarchy to the leaf nodes of a binary tree in [2], our scheme only needs to publish update information related to the the top-level identity of these revoked users. Therefore, our construction achieves efficient revocation without blowing up the number of total users n in the system and also minimizes the revocation number r to some extent.

4.1 Concrete Construction

Let $\mathbb{G}_1$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_1$. In addition, let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ denote the bilinear map. A security parameter κ will determine the size of the groups. We also define the Lagrange coefficient $\Delta_{i,J}$ for $i \in \mathbb{Z}_p$ and a set J of elements in $\mathbb{Z}_p$: $\Delta_{i,J}(x) = \prod_{j \in J, j \neq i} \frac{x-j}{i-j}$.

Setup $\mathcal{S}(1^\kappa, n)$ Choose α, β uniformly at random from $\mathbb{Z}_p^*$. Set $g_1 = g^\alpha$ and $g_2 = g^\beta$. Choose a polynomial $q(x)$ of degree ℓ at random subject to the constraint that $q(0) = \beta$. Also choose $h_0, h_1, \cdots, h_\ell$ randomly from $\mathbb{Z}_p$.

The public parameters PK contains (g, g_1, $g_2 = g^{q(0)}$, $g^{q(1)}$, $g^{q(2)}$, $\cdots$, $g^{q(\ell)}$; $h_0, h_1, \cdots, h_\ell$). The master key is $MK = \alpha$. The public parameters define the following publicly computable functions $F_j, V : \mathbb{Z}_p \to \mathbb{G}$ for $j \in [0, \ell]$ as $F_j(x) = g_1^x h_j$ and $V(x) = g^{q(x)}$. We also define a collision resistant hash $H : (\mathbb{Z}_p)^\ell \to \mathbb{Z}_p$. Besides, choose λ uniformly at random from $\mathbb{Z}_p$ and sets $MK_1 = \lambda$ and $MK_2 = \alpha - \lambda$.

Private Key Generation $\mathcal{SK}(MK_1,\ \omega,\ PK)$ Choose $(\rho_1, \cdots, \rho_j)$ uniformly at random from $\mathbb{Z}_p^j$. For an identity $\omega = (\omega_1, \cdots, \omega_j) \in \mathbb{Z}_p^j$ of depth $j \le \ell$, the algorithm outputs a private key for identity ω as $SK_\omega{=}\left(D_\omega^{(0)}, D_\omega^{(1)}, \cdots, D_\omega^{(j)}\right) = \left(g_2^{MK_1} \cdot \prod_{k=1}^{j} F_k(\omega_k)^{\rho_k},\ g^{\rho_1}, \cdots, g^{\rho_j}\right){=}\left(g_2^{\lambda} \cdot \prod_{k=1}^{j} F_k(\omega_k)^{\rho_k},\ g^{\rho_1}, \cdots, g^{\rho_j}\right)$. Note that the private key for ω can be generated just given a private key for $\omega|_{j-1} = \{\omega_1, \cdots, \omega_{j-1}\} \in \mathbb{Z}_p^{j-1}$. Let $SK_{\omega|_{j-1}}{=}\left(D^{(0)}_{\omega|_{j-1}}, D^{(1)}_{\omega|_{j-1}}, \cdots, D^{(j-1)}_{\omega|_{j-1}}\right)$. To generate SK_ω pick random $(\rho'_1, \cdots, \rho'_j) \in \mathbb{Z}_p^j$ uniformly at random from $\mathbb{Z}_p$ and output $SK_\omega{=}$ $\left(D^{(0)}_{\omega|_{j-1}} \cdot \prod_{k=1}^{j} F_k(\omega_k)^{\rho'_k},\ D^{(1)}_{\omega|_{j-1}} \cdot g^{\rho'_1}, \cdots,\ D^{(j-1)}_{\omega|_{j-1}} \cdot g^{\rho'_{j-1}}, g^{\rho'_j}\right)$.

Key Update Generation $\mathcal{KU}(MK_2,\ PK,\ t,\ rl,\ st)$ For the revoked identities $\{\omega^{(i)}\}_{i=1}^r \in rl$, output a private key corresponding to an access structure $t \bigwedge_{i=1}^r \overline{\omega^{(i)}}$ where $\{\omega^{(i)}\}_{i=1}^r$. [1]

For each revoked user $\omega^{(i)}, i \in [1, r]$, select two random values $\lambda_i, \rho_i \in \mathbb{Z}_p^2$, and publish the update information as $D_i{=}(D_i^{(3)}, D_i^{(4)}, D_i^{(5)}){=}(g_2^{\lambda_i+\rho_i},\ V(\omega^{(i)})^{\rho_i},\ g^{\rho_i})$ for $\omega^{(i)}, i \in [1, r]$. Note that we need to map $\omega^{(i)}$ to $\mathbb{Z}_p$ using hash function H before the above information can be computed. For the update time t, select $\rho_t \overset{r}{\leftarrow} \mathbb{Z}_p$, publish the update information as $D_t{=}\left(D_t^{(1)}, D_t^{(2)}\right){=}\left(g_2^{MK_2-\sum_{i=1}^r \lambda_i} \cdot F_0(t)^{\rho_t}, g^{\rho_t}\right)$ $=(g_2^{\alpha-\lambda-\sum_{i=1}^r \lambda_i} \cdot F_0(t)^{\rho_t}, g^{\rho_t})$. Return the update information as $KU_t = \left(\{(\omega^{(i)}, D_i)\}_{i=1}^r,\right.$ $\left.D_t\right)$.

Encryption $\mathcal{E}(PK, \omega, t, M)$ Choose a random s from $\mathbb{Z}_p$. Generate the ciphertext as $C{=}\left(E^{(-1)}, E^{(0)}, E_\omega^{(1)}, E_\omega^{(2)}, \cdots, E_\omega^{(j)}, E_t, \overline{E}_\omega^{(1)}, \overline{E}_\omega^{(2)}, \cdots, \overline{E}_\omega^{(j)}, \overline{E}_{dum}^{(j+1)}, \cdots, \overline{E}_{dum}^{(\ell-1)}, \overline{E}_t\right)$ $=\left(M\hat{e}(g_1,\ g_2)^s, g^s, F_1(\omega_1)^s, F_2(\omega_2)^s, \cdots, F_j(\omega_j)^s, F_0(t)^s, V(\omega|_1)^s, V(\omega|_2)^s, \cdots,\right.$ $\left.V(\omega|_j)^s, V(\varpi_{j+1})^s, \cdots, V(\varpi_{\ell-1})^s, V(t)^s\right)$. $\{\varpi_{j+1}, \cdots, \varpi_{\ell-1}\}$ are dummy identities.

Decryption $\mathcal{D}(SK_\omega, KU_t, PK, C)$ If any prefix of $\omega{=}(\omega_1, \cdots, \omega_j)$ is not revoked at time t, then we have $\omega|_k = (\omega_1, \cdots, \omega_k) \notin \{\omega^{(i)}\}_{i=1}^r$ for each $k \in [1, j]$. Compute the partial decryption as $Z_\omega{=}\frac{\prod_{k=1}^{j} \hat{e}(D_\omega^{(k)},\ E_\omega^{(k)})}{\hat{e}(D_\omega^{(0)},\ E^{(0)})}{=}\frac{\prod_{k=1}^{j} \hat{e}(g^{\rho_k},\ F_k(\omega_k)^s)}{\hat{e}(g_2^{\lambda} \cdot \prod_{k=1}^{j} F_k(\omega_k)^{\rho_k},\ g^s)}{=}\frac{1}{\hat{e}(g_2^{\lambda},\ g^s)}$

Compute the partial decryption for update time t as:
$Z_t{=}\frac{\hat{e}(D_t^{(2)},\ E_t)}{\hat{e}(D_t^{(1)},\ E^{(0)})}{=}\frac{\hat{e}(g^{\rho_t},\ F_0(t)^s)}{\hat{e}(g_2^{\alpha-\lambda-\sum_{i=1}^r \lambda_i} \cdot F_0(t)^{\rho_t},\ g^s)}{=}\frac{1}{\hat{e}(g_2,\ g)^{s(\alpha-\lambda-\sum_{i=1}^r \lambda_i)}}.$

For each revoked user $\omega^{(i)}, i \in [1, r]$, let $\gamma{=}\{\omega|_1,\ \cdots,\ \omega|_j,\ \varpi_{j+1},\ \cdots,\ \varpi_{\ell-1},$ $t,\ \omega^{(i)}\}$. The following partial decryption uses the partial ciphertext $(\overline{E}_\omega^{(1)},\ \overline{E}_\omega^{(2)},$

[1] We note that all the subordinate identities of an identity are revoked if this particular identity is revoked as stated in the above, and hence $\{\omega_i\}_{i=1}^r$ do not necessarily include all the revoked identities in *rl*.

$\cdots, \overline{E}_{\omega}^{(j)}, \overline{E}_{dum}^{(j+1)}, \cdots, \overline{E}_{dum}^{(\ell-1)}, \overline{E}_t) = (V(\omega|_1)^s, V(\omega|_2)^s, \cdots, V(\omega|_j)^s, V(\varpi_{j+1})^s, \cdots,$ $V(\varpi_{\ell-1})^s, V(t)^s)$ and the update information $(D_i^{(3)}, D_i^{(4)}, D_i^{(5)}) = (g_2^{\lambda_i+\rho_i}, V(\omega^{(i)})^{\rho_i},$ $g^{\rho_i})$. Compute the Langrangian coefficients $\{\sigma_x\}_{x\in\gamma}$ such that $\sum_{x\in\gamma}\sigma_x q(x) = q(0) = \beta$.

For each revoked user $\omega^{(i)}$, compute

$$Z_i = \frac{\hat{e}(g^{\rho_i}, V(\omega|_1)^{s\sigma_{\omega|_1}} \cdots V(\omega|_j)^{s\sigma_{\omega|_j}} V(\varpi_{j+1})^{s\sigma_{\varpi_{j+1}}} \cdots V(\varpi_{\ell-1})^{s\sigma_{\varpi_{\ell-1}}} V(t)^{s\sigma_t})\hat{e}(V(\omega^{(i)})^{\sigma_{\omega^{(i)}}\rho_i}, g^s)}{\hat{e}(g_2^{\lambda_i+\rho_i}, g^s)}$$

$$= \frac{\hat{e}(g^{\rho_i}, g^{sq(0)})}{\hat{e}(g_2^{\lambda_i+\rho_i}, g^s)} = \frac{1}{\hat{e}(g_2^{\lambda_i}, g^s)}$$

Finally, compute $\frac{1}{\hat{e}(g_2, g)^{s\lambda}} \cdot \frac{1}{\hat{e}(g_2, g)^{s(\alpha-\lambda-\sum_{i=1}^{r}\lambda_i)}} \cdot \frac{1}{\prod_{i=1}^{r}\hat{e}(g_2, g)^{s\lambda_i}} = \frac{1}{\hat{e}(g_2, g)^{s\alpha}}$ $= \frac{1}{\hat{e}(g_2, g_1)^s}$ and $\frac{E^{(-1)}}{\hat{e}(g_2, g_1)^s} = M$.

The security of this construction can be stated as the following theorem, the proof of which could be found in Appendix B.

Theorem 2. *If an adversary can break the proposed scheme in the sRID model, then a simulator can be constructed to play the Decisional BDH game with a non-negligible advantage.*

5 Conclusions

This presents a revocable IBE construction where the private key only contains two group elements. The update information size depends on the revocation number r. The proposed construction is the most space efficient revocable IBE constructions to our best knowledge. This paper provides a generic way to transform a non-monotonic attribute based encryption scheme into a revocable IBE scheme. We also show how this method can be applied to construct a space efficient revocable hierarchical IBE scheme.

References

1. W. Aiello, S. Lodha, and R. Ostrovsky. Fast digital identity revocation (extended abstract). In CRYPTO, pages 137–152, 1998.
2. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation.
3. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In ACM Conference on Computer and Communications Security, pages 417–426, 2008.
4. D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In EUROCRYPT, pages 223–238, 2004.
5. D. Boneh, X. Ding, G. Tsudik, and M. Wong. A method for fast revocation of public key certificates and security capabilities. In 10th USENIX Security Symposium, pages 297–308, 2001.
6. X. Ding and G. Tsudik. Simple identity-based cryptography with mediated rsa. In CT-RSA, pages 193–210, 2003.

7. C. Gentry. Certificate-based encryption and the certificate revocation problem. In EUROCRYPT, pages 272–293, 2003.
8. V. Goyal. Certificate revocation using fine grained certificate space partitioning. In Financial Cryptography, pages 247–259, 2007.
9. B. Libert and D. Vergnaud. Adaptive-id secure revocable identity-based encryption. In CT-RSA, pages 1–15, 2009.
10. R. Ostrovsky, A. Sahai, and B. Waters. Attribute-based encryption with non-monotonic access structures. In ACM Conference on Computer and Communications Security, pages 195–203, 2007.
11. A. Sahai and B. Waters. Revocation systems with very small private keys. http://eprint.iacr.org/2008/309.
12. A. Sahai and B. Waters. Fuzzy identity-based encryption. In EUROCRYPT, pages 457–473, 2005.
13. A. Shamir. Identity-based cryptosystems and signature schemes. In CRYPTO, pages 47–53, 1984.
14. B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. http://eprint.iacr.org/2008/290.

6 Appendix

A. Proof of Theorem A

Proof. The simulator is provided a random tuple (X, Y, Z, W) which is either $g_1, g_2, g_3, \hat{e}(g, g)^{z_1 z_2 z_3}$ or a random tuple $g_1, g_2, g_3, \hat{e}(g, g)^{z}$. The simulator is given the input tuple $g_1, g_2, g_3, \hat{e}(g, g)^{z_1 z_2 z_3}$ when $\mu = 1$ and $\mu = 0$ otherwise.
Init The adversary declares the challenge identity ω^*, time t^*. The simulator sets $\gamma^* = \{\omega^*, t^*\}$

Setup The simulator assigns the public parameters $g_1 = X$ and $g_2 = Y$ which implicitly sets $\alpha = z_1$ and $\beta = z_2$. It then chooses a random two-degree polynomial $f(x)$ and fixes a two-degree polynomial $u(x)$ as follows: set $u(x) = -x^2$ for all $x \in \gamma^*$ and $u(x) \neq -x^2$ for $x \notin \gamma^*$. Because $-x^2$ and $u(x)$ are two degree-2 polynomials, they will have at most two points in common or they are the same polynomial. This construction ensures that $\forall x, u(x) = -x^2$ iff $x \in \gamma^*$.

The simulator will now set the polynomials h and q as follows: first, $h(x) = \beta u(x) + f(x)$. Then, the simulator chooses two points θ_{ω^*} and θ_{t^*} uniformly at random from $\mathbb{Z}_p^*$, and implicitly sets $q(x)$ such that $q(0) = \beta$, while $q(\omega^*) = \theta_{\omega^*}$ and $q(t^*) = \theta_{t^*}$. The simulator outputs the following group elements for the public key: for $i = 1, 2$, it sets outputs $V(i) = g^{q(i)}$ by interpolation in the exponent using $\{\theta_{\omega^*}, \theta_{t^*}\}$ and B. For $i = 0, 1, 2$, it sets $g^{h(i)} = g_2^{u(i)} g^{f(i)}$. These values are (jointly) distributed identically to their distribution in the actual scheme. Note that implicitly we have $T(x) = g_2^{x^2 + u(x)} g^{f(x)}$.

The simulator also needs to pick up a random bit $rev \in \{0, 1\}$ and $\lambda \in \mathbb{Z}_p^*$. For $rev = 1$, set $MK_1 = \lambda$ and implicitly set $MK_2 = \alpha - \lambda$. For $rev = 0$, set

$MK_2 = \lambda$ and implicitly set $MK_1 = \alpha - \lambda$. We note that these two kinds of choice are both well formed due to the involvement of λ.

Phase 1 *Private key generation* oracle $\mathcal{SK}(\omega)$:

If *rev* $= 1$, then the private key can be provided similar to the real scenario. Pick up a random $\rho_\omega \in \mathbb{Z}_p^*$, $D_\omega{=}(B^\lambda \cdot T(\omega)^{\rho_\omega} = g_2^\lambda \cdot T(\omega)^{\rho_\omega}, g^{\rho_\omega})$.

If *rev* $= 0$ and $\omega = \omega^*$ then abort and the simulator outputs $\mu' = 1$.

If *rev* $= 0$ and $\omega \neq \omega^*$ then: let $g_4 = A/g^\lambda = g^{\alpha-\lambda}$. Choose $\rho'_\omega \in \mathbb{Z}_p^*$ at random, compute D_ω as: $D_\omega^{(1)}{=}g_4^{\frac{-f(\omega)}{\omega^2+u(\omega)}} \cdot \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\rho'_\omega}$, $D_\omega^{(2)}{=}g_4^{\frac{-1}{\omega^2+u(\omega)}} \cdot g^{\rho'_\omega}$.

Let's verify the above private key is correctly distributed. Let $\rho_\omega = \frac{\lambda-\alpha}{\omega^2+u(\omega)} + \rho'_\omega$, then $D_\omega^{(2)}{=}g_4^{\frac{-1}{\omega^2+u(\omega)}} {\cdot} g^{\rho'_\omega}{=}g^{\rho_\omega}$, $D_\omega^{(1)}{=}g_4^{\frac{-f(\omega)}{\omega^2+u(\omega)}} {\cdot} \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\rho'_\omega} {=} g_4^{\frac{-f(\omega)}{\omega^2+u(\omega)}} \cdot \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\frac{\alpha-\lambda}{\omega^2+u(\omega)}} {\cdot} \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\rho'_\omega} {\cdot} \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\frac{\lambda-\alpha}{\omega^2+u(\omega)}} {=} g_2^{\alpha-\lambda} {\cdot} \left(g_2^{\omega^2+u(\omega)} g^{f(\omega)}\right)^{\rho_\omega} {=} g_2^{MK_1} \cdot T(\omega)^{\rho_\omega}$.

Revocation $\mathcal{R}(\omega, t)$:

For all user ω, add (ω, t) to *rl*.

Key update $\mathcal{KU}(t)$:

If *rev* $= 1$ and we have that $(\omega^*, t^*) \notin rl$, then abort and the simulator outputs $\mu' = 1$.

Else if *rev* $= 1$ and $t = t^*$, choose $\lambda_i \in \mathbb{Z}_p^*$ for each $\{\omega^{(i)}\}_{i=1}^{r-1}$ and $\lambda'_r \in \mathbb{Z}_p^*$, $\rho'_r \in \mathbb{Z}_p^*$. Wlog, we assume that $\omega^{(r)} = \omega^*$. The update information for ω^* can be given as $D_r^{(3)} = g_2^{\rho'_r}$, $D_r^{(4)} = g_1^{-\theta_{\omega^*}} \cdot g^{\theta_{\omega^*} \cdot (\lambda'_r + \rho'_r)} = g^{\theta_{\omega^*} \cdot (-z_1 + \lambda'_r + \rho'_r)}$, $D_r^{(5)} = g_1^{-1} \cdot g^{(\lambda'_r + \rho'_r)} = g^{-z_1 + \lambda'_r + \rho'_r}$. We implicitly set $\lambda_r = z_1 - \lambda'_r$ and $\rho_r = \lambda'_r - z_1 + \rho'_r {=} \rho'_r - \lambda_r$. Therefore, we can verify the above update information is correctly formed: $D_r^{(3)} = g_2^{\rho'_r} {=} g_2^{\lambda_r + \rho_r}$, $D_r^{(4)} = g^{\theta_{\omega^*} \cdot (-z_1 + \lambda'_r + \rho'_r)} {=} V(\omega^*)^{\rho_r}$, $D_r^{(5)} = g^{-z_1 + \lambda'_r + \rho'_r} {=} g^{\rho_r}$.

For the rest update information, the generation process is similar to that of the real scenario: for $i \in [1, r-1]$, $D_i^{(3)} = g_2^{\lambda_i + \rho_i}$, $D_i^{(4)} {=} V(\omega^{(i)})^{\rho_i}$, $D_i^{(5)} {=} g^{\rho_i}$; for update time t^*, $D_{t^*}^{(1)} {=} g_2^{\lambda'_r - \sum_{i=1}^{r-1} \lambda_i - \lambda} {\cdot} T(t^*)^{\rho_{t^*}} {=} g_2^{z_1 - \lambda - (z_1 - \lambda'_r) - \sum_{i=1}^{r-1} \lambda_i} {\cdot} T(t^*)^{\rho_{t^*}} {=} g_2^{MK_2 - \sum_{i=1}^{r} \lambda_i} \cdot T(t^*)^{\rho_{t^*}}$, $D_{t^*}^{(2)} {=} g^{\rho_{t^*}}$. We have ρ_{t^*} randomly selected from $\mathbb{Z}_p^*$.

If *rev* $= 1$ and $t \neq t^*$, the simulator selects $\{\lambda_i, \rho_i\} \overset{r}{\leftarrow} (\mathbb{Z}_p^*)^2$ for $i \in [1, r]$. The simulator also randomly selects ρ'_t and let $g_4 = g_1 \cdot g^{-\lambda - \sum_{i=1}^{r} \lambda_i} = g^{\alpha - \lambda - \sum_{i=1}^{r} \lambda_i}$. The update information for update time t would be publish as $D_t^{(1)} {=} g_4^{\frac{-f(t)}{t^2+u(t)}} \left(g_2^{t^2+u(t)} g^{f(t)}\right)^{\rho'_t}$, $D_t^{(2)} {=} g_4^{\frac{-1}{t^2+u(t)}} g^{\rho'_t}$. Let $\rho_t = \frac{\sum_{i=1}^{r} \lambda_i + \lambda - \alpha}{t^2+u(t)} + \rho'_t$, we can verify that $D_t^{(2)} {=} g^{\rho_t}$, $D_t^{(1)} {=} g_2^{\alpha - \lambda - \sum_{i=1}^{r} \lambda_i} \cdot \left(g_2^{t^2+u(t)} g^{f(t)}\right)^{\rho_t} {=} g_2^{MK_2 - \sum_{i=1}^{r} \lambda_i} \cdot T(t)^{\rho_t}$ by the similar analysis as in the third case of

Private key generation oracle. The update information for the revoked users $\omega^{(i)}, i \in [1, r]$ can be generated as: $D_i^{(3)} = g_2^{\lambda_i+\rho_i}$, $D_i^{(4)}{=}V(\omega^{(i)})^{\rho_i}$, $D_i^{(5)}{=}g^{\rho_i}$.

If $rev = 0$, the update information is given as the real scenario since $MK_2 = \lambda$ is known to the simulator.

Challenge $E^{(1)} = M_b{\cdot}W$, $E^{(2)}{=}Z$, $E^{(3)}{=}g_3^{f(\omega^*)}$, $E^{(4)}{=}g_3^{f(t^*)}$, $E^{(4)}{=}g_3^{q(\omega^*)}$, $E^{(5)}{=}g_3^{q(t^*)}$.

Phase 2 Same to **Phase 1**.

Guess Let **nare** and **nara** denote the events that none of the oracles abort when $\mu = 1$ and $\mu = 0$ respectively, namely the simulator is given g_1, g_2, g_3, $\hat{e}(g, g)^{z_1 z_2 z_3}$ or a random tuple $g_1, g_2, g_3, \hat{e}(g, g)^z$ respectively. It is easy to see that Pr[**nare**]=Pr[**nara**] since the probability that the two oracles $\mathcal{SK}(\omega)$ and $\mathcal{KU}(t)$ abort depends on the bit *rev* which is chosen independently from whether $\mu = 1$ or $\mu = 0$. We'll show that Pr[**nare**]$\geq \frac{1}{2}$. In our security model, we require that $\mathcal{SK}(\omega)$ oracle can be queried on ω^* only if $\mathcal{R}(\omega, t)$ oracle was queried on (ω^*, t) at $t = t^*$. Thus, we have $\Pr[\omega = \omega^*] \leq \Pr[(\omega^*, t^*) \in rl] \Rightarrow 1{-}\Pr[\omega = \omega^*] \geq 1{-}\Pr[(\omega^*, t^*) \notin rl]$.

We see that $\mathcal{SK}(\omega)$ oracle aborts if $rev = 0$ and $\omega = \omega^*$ and $\mathcal{KU}(t)$ oracle aborts if $rev = 1$ and $(\omega^*, t^*) \notin rl$. Thus,

$$\begin{aligned}\Pr[\overline{\mathbf{nare}}] &= \Pr[(rev = 0) \bigwedge (\omega = \omega^*)] + \Pr[(rev = 1) \bigwedge (\omega^*, t^*) \notin rl] \\ &= \Pr[rev = 0] \cdot \Pr[\omega = \omega^*] + \Pr[rev = 1] \cdot \Pr[(\omega^*, t^*) \notin rl] \\ &\leq \quad \tfrac{1}{2}\Pr[\omega = \omega^*] \quad + \quad \tfrac{1}{2}(1 - \Pr[\omega = \omega^*]) \\ &\leq \quad \tfrac{1}{2}\end{aligned}$$

Therefore, Pr[**nare**] $\geq \frac{1}{2}$.

Let **succ** $= \epsilon$ denote the probability that the adversary outputs a $b' = b$, and **real** denote the probability that the simulator outputs $\mu' = 1$ when $\mu = 1$ and **rand** denote the probability that the simulator outputs $\mu' = 1$ when $\mu = 0$. It is easy to observe that when none of the oracles abort, then the simulator is playing the real CPA game for the adversary when $\mu = 1$, so

$$\Pr[\mathbf{real}|\mathbf{nare}] \geq \frac{1}{2}\Pr[\mathbf{succ}]$$

When $\mu = 0$ and none of the above oracles abort then b should be information-theoretically hidden from the adversary. So,

$$\Pr[\mathbf{rand}|\mathbf{nara}] = \frac{1}{2}$$

Also, since the simulator outputs $\mu' = 1$ when either of the oracles aborts, so

$$\Pr[\mathbf{real}|\overline{\mathbf{nare}}] = 1$$
$$\Pr[\mathbf{rand}|\overline{\mathbf{nara}}] = 1$$

Let **Adv** denote the probability that the simulator successfully solve the decisional BDH problem. We have

$$\begin{aligned}
\mathbf{Adv} = & \quad \mathbf{Pr[real] - Pr[rand]} \\
= & \quad \mathbf{Pr[nare] \cdot Pr[real|nare]} \\
& \quad + \mathbf{Pr}[\overline{\mathbf{nare}}] \cdot \mathbf{Pr}[\mathbf{real}|\overline{\mathbf{nare}}] \\
& \quad - \mathbf{Pr[nara] \cdot Pr[rand|nara]} \\
& \quad - \mathbf{Pr}[\overline{\mathbf{nara}}] \cdot \mathbf{Pr}[\mathbf{rand}|\overline{\mathbf{nara}}] \\
\geq & \quad \frac{1}{2} \cdot (\Pr[\mathbf{real}|\mathbf{nare}] \\
& \quad - \Pr[\mathbf{rand}|\mathbf{nara}]) \\
\geq & \quad \frac{1}{2}(\Pr(succ) - \frac{1}{2}) \\
\geq & \quad \frac{1}{4}\epsilon
\end{aligned}$$

B. Proof of Theorem B

Proof. The simulator is provided a random tuple (X, Y, Z, W) which is either $g_1, g_2, g_3, \hat{e}(g, g)^{z_1 z_2 z_3}$ or a random tuple $g_1, g_2, g_3, \hat{e}(g, g)^z$. The simulator is given the input tuple $g_1, g_2, g_3, \hat{e}(g, g)^{z_1 z_2 z_3}$ when $\mu = 1$ and $\mu = 0$ otherwise.

Init The adversary declares the challenge identity $\omega^* = (\omega_1^*, \cdots, \omega_j^*) \in \mathbb{Z}_p^j$ of depth $j \leq \ell$, time t^*. The simulator sets $\gamma^* = \{\omega^*|_1, \cdots, \omega^*|_j, \varpi_{j+1}^*, \cdots, \varpi_{\ell-1}^*, t^*\}$.

Setup The simulator assigns the public parameters $g_1 = X$ and $g_2 = Y$ which implicitly sets $\alpha = z_1$ and $\beta = z_2$. It then picks $\alpha_0, \cdots, \alpha_\ell \in \mathbb{Z}_p^{\ell+1}$ at random and defines $h_j = g_1^{-\omega_j^*} g^{\alpha_j} \in \mathbb{G}_1$ for $j = 1, \cdots, \ell$ and $h_0 = g_1^{-t^*} g^{\alpha_0} \in \mathbb{G}_1$. Then the function $F_j : \mathbb{Z}_p \rightarrow \mathbb{G}_1$ is defined as: $F_j(x) = g_1^x h_j = g_1^{x-\omega_j^*} g^{\alpha_j}$ for $j \in [1, \ell]$, and $F_0(x) = g_1^{x-t^*} g^{\alpha_0}$.

The simulator will now set the polynomials q as follows: the simulator chooses ℓ points $\theta_{\omega^*|_1}, \cdots, \theta_{\omega^*|_j}, \theta_{\varpi_{j+1}^*}, \cdots, \theta_{\varpi_{\ell-1}^*}$, and θ_{t^*} uniformly at random from $\mathbb{Z}_p^*$, and implicitly sets $q(x)$ such that $q(0) = \beta$, while $q(\omega^*|_k) = \theta_{\omega^*|_k}$ for $k \in [1, j]$, $q(\varpi_k^*) = \theta_{\varpi_k^*}$ for $k \in [j+1, \ell-1]$ and $q(t^*) = \theta_{t^*}$. The simulator outputs the following group elements for the public key: for $i = 0, \cdots, \ell$, it sets outputs $V(i)=g^{q(i)}$ by interpolation in the exponent using $\theta_{\omega^*|_1}, \cdots, \theta_{\omega^*|_j}, \theta_{\varpi_{j+1}^*}, \cdots, \theta_{\varpi_{\ell-1}^*}, \theta_{t^*}$ and B. These values are (jointly) distributed identically to their distribution in the actual scheme.

The simulator also needs to pick up a random bit $rev \in \{0, 1\}$ and $\lambda \in \mathbb{Z}_p^*$. For $rev = 1$, set $MK_1 = \lambda$ and implicitly set $MK_2 = \alpha - \lambda$. For $rev = 0$, set

$MK_2 = \lambda$ and implicitly set $MK_1 = \alpha - \lambda$. We note that these two kinds of choice are both well formed since λ are uniformly selected at random.

Phase 1 *Private key generation* oracle $\mathcal{SK}(\omega)$:

If $rev = 1$, then the private key can be provided exactly as the real scenario since MK_1 is known to the simulator.

If $rev = 0 \bigwedge(\omega = \omega^* \bigvee_{k=1}^{j} \omega = \omega^*|_k)$ then abort.

If $rev = 0 \bigwedge(\omega \neq \omega^* \bigwedge_{k=1}^{j} \omega \neq \omega^*|_k)$ then: since $\omega = (\omega_1, \cdots, \omega_u) \in \mathbb{Z}_p^u$ is not a prefix of ω^*. Let j be the smallest index such that $\omega_j \neq \omega_j^*$, and hence $1 \leq j \leq u$. To respond to this query, the simulator first derives a private key for the identity $(\omega_1, \cdots, \omega_j)$ from which it can construct a private key for the requested identity $\omega = (\omega_1, \cdots, \omega_j, \cdots, \omega_u)$. The simulator picks up random elements $\rho_1, \cdots, \rho_j \in \mathbb{Z}_p^j$ and sets $D_\omega^{(0)} = g_2^{\frac{-\alpha_j}{\omega_j - \omega_j^*}} g_2^{-\lambda} \prod_{v=1}^{j} F_v(\omega_v)^{\rho_v}$, $D_\omega^{(1)} = g^{\rho_1}, \cdots, D_\omega^{(j-1)} = g^{\rho_{j-1}}$, $D_\omega^{(j)} = g_2^{\frac{-1}{\omega_j - \omega_j^*}} g^{\rho_j}$. We claim that $(D_\omega^{(0)}, \cdots, D_\omega^{(j)})$ is a valid private key for $(\omega_1, \cdots, \omega_j)$. To see this, let $\tilde{\rho}_j = \rho_j - z_2/(\omega_j - \omega_j^*)$. Then we have that $g_2^{\frac{-\alpha_j}{\omega_j - \omega_j^*}} F_j(\omega_j)^{\rho_j} = g_2^{\frac{-\alpha_j}{\omega_j - \omega_j^*}} (g_1^{\omega_j - \omega_j^*} g^{\alpha_j})^{\rho_j} = g_2^{z_1} (g_1^{\omega_j - \omega_j^*} g^{\alpha_j})^{\rho_j - \frac{z_2}{\omega_j - \omega_j^*}} = g_2^{z_1} F_j(\omega_j)^{\tilde{\rho}_j}$, hence $D_\omega^{(0)} = g_2^{z_1 - \lambda} \cdot \prod_{k=1}^{j-1} F_k(\omega_k)^{\rho_k} F_j(\omega_j)^{\tilde{\rho}_j} = g_2^{MK_1} \cdot \prod_{k=1}^{j-1} F_k(\omega_k)^{\rho_k} F_j(\omega_j)^{\tilde{\rho}_j}$, $D_\omega^{(1)} = g^{\rho_1}$, $\cdots$, $D_\omega^{(j-1)} = g^{\rho_{j-1}}$, $D_\omega^{(j)} = g^{\tilde{\rho}_j}$. The provided private key is correctly distributed since $\rho_1, \cdots, \rho_{j-1}, \tilde{\rho}_j$ are random.

Revocation $\mathcal{R}(\omega, t)$:

For all user ω, add (ω, t) to *rl*.

Key update $\mathcal{KU}(t)$:

If $rev = 1$ and $t = t^*$ and $\forall t \leq t^*$ we have that $(\omega^*, t) \notin rl \bigwedge_{k=1}^{j} (\omega^*|_k, t) \notin rl$, then abort.

Else if $rev = 1$ and $t = t^*$, choose $\lambda_i \in \mathbb{Z}_p^*$ for each revoked identity $\omega^{(i)}, i = 1, \cdots, r-1$ and $\lambda'_r \in \mathbb{Z}_p^*$, $\rho'_r \in \mathbb{Z}_p^*$. Wlog, we assume that $\omega^{(r)} = \omega^*$ or $\omega^{(r)}$ is any prefix of ω^*, and for simplicity of discussion and wlog we assume ω^* is the revoked identity. The update information for ω^* can be given as $D_r^{(3)} = g_2^{\rho'_r}$, $D_r^{(4)} = g_1^{-\theta_{\omega^*}} \cdot g^{\theta_{\omega^*} \cdot (\lambda'_r + \rho'_r)} = g^{\theta_{\omega^*} \cdot (-z_1 + \lambda'_r + \rho'_r)}$, $D_r^{(5)} = g_1^{-1} \cdot g^{(\lambda'_r + \rho'_r)} = g^{-z_1 + \lambda'_r + \rho'_r}$. We implicitly set $\lambda_r = z_1 - \lambda'_r$ and $\rho_r = \lambda'_r - z_1 + \rho'_r = \rho'_r - \lambda_r$. Therefore, we can verify the above update information is correctly formed: $D_r^{(3)} = g_2^{\rho'_r} = g_2^{\lambda_r + \rho_r}$, $D_r^{(4)} = g^{\theta_{\omega^*} \cdot (-z_1 + \lambda'_r + \rho'_r)} = V(\omega^*)^{\rho_r}$, $D_r^{(5)} = g^{-z_1 + \lambda'_r + \rho'_r} = g^{\rho_r}$.

For the rest update information of the other revoked users $i \in [1, r-1]$, the update information is generated exactly as the real: $D_i^{(3)} = g_2^{\lambda_i + \rho_i}$, $D_i^{(4)} = V(\omega_i)^{\rho_i}$, $D_i^{(5)} = g^{\rho_i}$; for update time t^*, $D_{t^*}^{(1)} = g_2^{\lambda'_r - \sum_{i=1}^{r-1} \lambda_i - \lambda} \cdot F_0(t^*)^{\rho_{t^*}} = g_2^{z_1 - \lambda - (z_1 - \lambda'_r) - \sum_{i=1}^{r-1} \lambda_i}$.

$F_0(t^*)^{\rho_{t^*}} = g_2^{MK_2-\sum_{i=1}^{r}\lambda_i} \cdot F_0(t^*)^{\rho_{t^*}}$, $D_{t^*}^{(2)} = g^{\rho_{t^*}}$. We have ρ_{t^*} randomly selected from $\mathbb{Z}_p^*$.

If $rev = 1$ and $t \neq t^*$, the simulator selects $\{\lambda_i, \rho_i\} \overset{r}{\leftarrow} (\mathbb{Z}_p^*)^2$ for $i \in [1, r]$. The simulator also randomly selects ρ_t' and let $g_4 = g_1 \cdot g^{-\lambda-\sum_{i=1}^{r}\lambda_i} = g^{\alpha-\lambda-\sum_{i=1}^{r}\lambda_i}$. The update information for update time t would be publish as $D_t^{(1)} = g_2^{\frac{-\alpha_0}{t-t^*}} g_2^{-\lambda-\sum_{i=1}^{r}\lambda_i} F_0(t)^{\rho_t}$, $D_t^{(2)} = g_2^{\frac{-1}{t-t^*}} g^{\rho_t}$. Let $\tilde{\rho}_t = \rho_t - z_2/(t-t^*)$, we can verify that $D_t^{(2)} = g^{\tilde{\rho}_t}$, $D_t^{(1)} = g_2^{\frac{-\alpha_0}{t-t^*}} g_2^{-\lambda-\sum_{i=1}^{r}\lambda_i} g_1^{(t-t^*)\rho_t} g^{\alpha_0\rho_t} = g_2^{\alpha-\lambda-\sum_{i=1}^{r}\lambda_i} \cdot \left(g_1^{t-t^*} \cdot g^{\alpha_0}\right)^{\rho_t - z_2/(t-t^*)} = g_2^{\alpha-\lambda-\sum_{i=1}^{r}\lambda_i} \cdot F_0(t)^{\tilde{\rho}_t}$. The update information for the revoked users $\omega_i, i \in [1, r]$ can be generated as the real scenario since λ_i is known to the simulator.

If $rev = 0$, the update information is given as the real scenario since $MK_2 = \lambda$ is known to the simulator.

Challenge $\left(E^{(-1)}, E^{(0)}, E_\omega^{(1)}, E_\omega^{(2)}, \cdots, E_\omega^{(j)}, E_t, \overline{E}_\omega^{(1)}, \overline{E}_\omega^{(2)}, \cdots, \overline{E}_\omega^{(j)}, \overline{E}_{dum}^{(j+1)}, \cdots, \overline{E}_{dum}^{(\ell-1)}, \overline{E}_t\right)$
$=\left(M_b W, Z, Z^{\alpha_1}, Z^{\alpha_2}, \cdots, Z^{\alpha_j}, Z^{\alpha_0}, Z^{\theta_{\omega^*|_1}}, Z^{\theta_{\omega^*|_2}}, \cdots, Z^{\theta_{\omega^*|_j}}, Z^{\theta_{\varpi^*|_{j+1}}}, \cdots, Z^{\theta_{\varpi^*|_{\ell-1}}}, Z^{\theta_{t^*}}\right)$

Phase 2 Same to **Phase 1**.

Guess Since the condition that abortion happens in the above simulation is exactly the same to that of [2], therefore a similar analysis to **Claim 4.4** can be applied to the above simulation. Hence, the overall advantage of the simulator in the Decisional BDH game is larger than $\frac{1}{4\ell}\epsilon$, where ϵ denotes the probability that the adversary successfully attacks the proposed construction.