Collusion-Resistant Broadcast Encryption with Tight Reductions

Linfeng Zhou*

Abstract

We present a *tightly secure* broadcast encryption scheme of composite-order bilinear groups in the selective security model. Our construction and proof rely on the recently novel technique from Chen (eprint 2016/891). The proof of our construction will lead to only $O(\log n)$ or $O(\log \lambda)$ security loss, rather than $O(n)$ security loss as the construction given by Wee (TCC-A 16) and many other previous constructions.

1 Introduction

The notion of broadcast encryption was first introduced by Fiat and Noar[FN94]. A broadcast encryption (BE) system consists of n users, and it allows the broadcaster sending encrypted contents to a set of qualified users $S \subseteq \{1, \cdots, n\}$ dynamically such that the set of users in S can decrypt the broadcast with their own secret key, but users outside of the set S cannot decrypt the broadcast even if they all collude and pool their secret keys.

There are many ways to define the security of a broadcast encryption scheme. The most desirable broadcast encryption scheme need to satisfy the notion called *adaptive security*, in which the adversary can adaptively corrupt users, learning their secret keys, and then he chooses an arbitrary uncorrupted "challenge" set S^* of users that he wants to attack in order to learn some information about the plaintext broadcast. The adversary may continue corrupting users so long as he does not corrupt any of the intended recipients of the broadcast. We also consider a weaker security notion called *selective security*, where the adversary may still choose the challenge set S^* arbitrarily, but must choose it before corrupting any users and before even seeing the public parameters of the scheme.

However, in this work we mainly focus on the issue of security reduction and security loss in the broadcast encryption system under static assumptions of bilinear groups. Consider a broadcast encryption scheme with a security reduction showing that attacking the scheme in time T with success probability ϵ implies breaking some computationally hard problem in time roughly T with success probability ϵ/L. We call L as the security loss and a tight reduction is one where L is only related to the security parameter λ and more optimal, L is a constant. Namely we are interested in constructions based on static assumptions like the decisional subgroup assumption that do not rely on random oracles.

Tight reductions are not just theoretical issues for broadcast encryption, rather than are of importance in practice. If the security loss L increases, we must in turn increase the size of the underlying groups in order to compensate for it. This will significantly effect the running time and the performance of the broadcast encryption system.

*Contact: daniel.linfeng.zhou@gmail.com

2 Our Contribution

In this paper, we present the first *tightly secure* broadcast encryption scheme, to the best of our knowledge, that achieves constant-size ciphertext overhead, constant-size private keys and linear-size public parameters under static assumptions in the selective security model. In our scheme the security loss is only $O(\log n)$ or $O(\log \lambda)$ where $n = \mathsf{poly}(\lambda)$ is the number of users in the broadcast encryption system and λ is the security parameter.

Intuition. Our work is inspired by the techniques used by Chen [Che16] in building tightly secure IBE and the techniques used by Wee [Wee16] in building the broadcast encryption under static assumptions with same parameter sizes as the work of Boneh, Gentry and Waters (BGW) [BGW05]. Our result can be regarded as a combination of these two works.

Before describing how to construct our tightly secure broadcast encryption scheme. We first briefly review Wee's broadcast encryption scheme and Chen's techniques in building tightly secure IBE.

Wee's Broadcast Encryption. Given the composite-order bilinear group $\mathbb{G} = (N = p_1p_2p_3, G, G_T, e)$. Wee constructed the broadcast encryption in a similar approach as BGW while embedding the dual system methodology[Wat09] and Déjà Q framework [CM14]. From a high level, the parameters in Wee's broadcast encryption can be described as follows

$$\begin{aligned}
\mathsf{mpk} &: (g_1, \{g_j\}_{j\in[n]}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n}, e(g_1, u_{n+1}), \mathsf{H}) \\
\mathsf{sk}_i &: u^{n-i+1}R_{3.i} \quad i = 1, \cdots, n \\
\mathsf{ct} &: \left(g_1^s, g_1^{(\beta+\sum_{k\in S}\alpha^k)s}\right) \\
\mathsf{key} &: \mathsf{H}(e(g_1, u^{n+1})^s)
\end{aligned}$$

where $(g_1, u) \leftarrow G_{p_1}$, $\alpha, \beta, s \leftarrow \mathbb{Z}_N$, $R_{4.i} \leftarrow G_{p_4}$ and S is the challenge set. Here we consider G_{p_1} as normal space, G_{p_2} as semi-functional space and G_{p_3} is the space to be used to randomize secret keys. Here we call G_{p_3} as the randomness space.

To establish security, Wee introduced the semi-functional space G_{p_2} to the $2n$ terms $u^{\alpha}, u^{\alpha^2}, \cdots, u^{\alpha^{2n}}$ and use the entropy derived from $u^{\alpha^{n+1}}$ to hide the message m through a function $F_{2n}(k) = \sum_{j=1}^{2n} r_j\alpha_j^k$ where the input k is in the interval $[1, 2n]$ and $r_1, \cdots, r_{2n}, \alpha_1, \cdots, \alpha_{2n} \leftarrow \mathbb{Z}_N$. Finally, we could replace the function $F_{2n}(\cdot)$ with a truly random function $\mathsf{RF}(\cdot)$. In order to avoid leaking $\alpha \mod p_2$ in the ciphertext in order to carry out the transformation to the secret keys. We need to eliminate all occurrences of α in the polynomial $\beta + \sum_{k\in S}\alpha^k$ which shows up in the ciphertext. Due to this restriction, we need to generate β by first randomly selecting another value $\widetilde{\beta}$ such that $\widetilde{\beta} = \beta + \sum_{k\in S}\alpha^k$ and then rewrite the ciphertext, symmetric key, and secret keys as

$$\mathsf{ct} = \left(g_1^s, g_1^{\widetilde{\beta}s}\right) \tag{1}$$

$$\mathsf{key} = \mathsf{H}(e(g_1, u_{n+1})^s) \tag{2}$$

$$\mathsf{sk}_i = u^{\alpha^{n-i+1}\widetilde{\beta} - \sum_{k\in S}\alpha^{n+1+k-i}}R_{3.i} \tag{3}$$

Thus the monomials in α only show up on the same side of the pairing in both the ciphertext and the secret keys in the exponents of u.

Remark. We remark that even though this step allows us to eliminate the occurrences of α in the ciphertext, it also sacrifies the possibility to achieve adaptive security or semi-static security. Because the challenger must know the challenge set S in order to generate β, such that $\widetilde{\beta} = \beta + \sum_{k\in S}\alpha^k$. Therefore, the adversary must commit to a challenge set at the very beginning of the selective security game. Thus, to solve this problem we must change the structure of the scheme to avoid this restriction while preserving existing properties. We leave it as an open problem.

Chen's technique. In Chen's tightly secure IBE, they noted that only one unity of entropy can be injected into the semi-functional space each time, since only one unit of random source can be extracted from the normal space. To achieve tight security, Chen tries to inject more entropy each time such that the reduction could reach the function $F_q(\cdot)$ from $F_1(\cdot)$ as quick as possible, where q is total number of key queries and challenge queries. Chen's idea is to extract entropy from $F_i(i \leq q)$ and then inject them back into F_i. If so, each F_i has i units of entropy and the structure of F_i allows to reach F_{2^i} directly. This is achieved through a iterated approach. Roughly speaking, for each iteration i, we first extract entropy $\widehat{F_{2^i}}$ from each F_{2^i} and then inject the entropy $\widehat{F_{2^i}}$ back into $F_{2^{i+1}}$ in the next iteration. This significantly accelerates the construction of F_q because we only need $\lceil \log q \rceil$ steps. To temporarily store the entropy extracted from each F_i, they introduced another subgroup, which can be viewed as *shadow semi-functional space*, into the reduction , and then we can flip all stored entropy in the shadow semi-functional space back to the old one. This leads to that the reduction should be working on a bilinear group of order $N = p_1p_2p_3p_4$ instead of order $N = p_1p_2p_3$. Therefore, in the group G, the subgroup G_{p_1} works as the normal space, the subgroup G_{p_2} works as the semi-functional space, the subgroup G_{p_3} works as the shadow semi-functional space, and the subgroup G_{p_4} works as the randomness space.

Our Approach. Following the construction of Wee's broadcast encryption and Chen's techniques used in tight reductions, we construct our tightly secure broadcast encryption scheme. We describe our method in a nutshell as follows.

Assume that we have rewritten the ciphertext overhead and the symmetric key as equation (1) and (2), except that we use the subgroup G_{p_4} to randomize secret keys instead of using the subgroup G_{p_3}. More specifically, the public parameters, secret keys, ciphertext overhead and the symmetric key are in the form as follows:

$$\begin{aligned}
\mathsf{mpk} &: (g_1, \{g_j\}_{j\in[n]}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n}, e(g_1, u_{n+1}), \mathsf{H}) \\
\mathsf{sk}_i &= u^{\alpha^{n-i+1}\widetilde{\beta} - \sum_{k\in S}\alpha^{n+1+k-i}} R_{4.i} \\
\mathsf{ct} &= \left(g_1^s, g_1^{\widetilde{\beta}s}\right) \\
\mathsf{key} &: \mathsf{H}(e(g_1, u^{n+1})^s)
\end{aligned}$$

Let us see how to apply Chen's technique into Wee's broadcast encryption scheme. Firstly we still need the function $F_{2n}(k) = \sum_{j=1}^{2n} r_j\alpha_j^k \in \mathbb{Z}_{p_2}$, where $r_1, \cdots, r_{2n}, \alpha_1, \cdots, \alpha_{2n} \xleftarrow{\$} \mathbb{Z}_{p_2}$. Recall that this function has been applied in Wee's broadcast encryption, and Wee constructed $F_{2n}(\cdot)$ from the entropy in the normal space following the roadmap

$$F_1 \rightarrow F_2 \rightarrow F_2 \rightarrow \cdots \rightarrow F_{2n}$$

We note that there is also only one unity of entropy that can be injected into the semi-functional space each time. Therefore, in order to achieve tight reduction, we try to inject more entropy each time following Chen's technique: extract entropy from $F_i(i \leq 2n)$ itself and then inject them back into F_i. Therefore, we only need $\lceil \log 2n \rceil = \lceil \log n + 1 \rceil$ steps to construct $F_{2n}(\cdot)$. To introduce the shadow semi-functional space, we need another function $\widehat{F_{2^i}}(k) = \sum_{j=1}^{2^i} \hat{r}_j\hat{\alpha}_j^k \in \mathbb{Z}_{p_3}$, where $\hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_3}$. We could first extract one unity entropy from u and α in the normal space and puts them into the semi-functional space. It is used to define the function $F_{2^0}(\cdot)$. Then we can execute the setups described above with the help of shadow semi-functional space. More specifically, we first introduce the semi-functional space to transform each $u_k = u^{\alpha^k}R_{4.k}$ into the form of $u^{\alpha^k}g_2^{F_{2^i}(k)}R_{4.k}$ and then introduce the shadow semi-functional space via transforming each $u^{\alpha^k}g_2^{F_{2^i}(k)}R_{4.k}$ into the form $u^{\alpha^k}g_2^{F_{2^i}(k)}g_3^{\widehat{F_{2^i}}(k)}R_{4.k}$. To flip all stored entropy back to the old one, we finally transform $u_k = u^{\alpha^k}g_2^{F_{2^i}(k)}g_3^{\widehat{F_{2^i}}(k)}R_{4.k}$ into the form of $u_k = u^{\alpha^k}g_2^{\widehat{F_{2^i}}'(k)}R_{4.k}$, where $\widehat{F_{2^i}}'(k) = \sum_{j=1}^{2^i} r_j\alpha_j^k + \hat{r}_j\hat{\alpha}_j^k$.

Open Problem. Our construction achieves tight security but it is only achieved in the selective security model. It would be interesting to strengthen it to semi-static or adaptive security. Furthermore, how to shrink the size of the public parameters, which is linear in the number of users, is also an interesting problem.

3 Related Work

Tight security in identity-based encryption system[CW13, GDCC16, Che16] and in digital signature schemes[Hof16a, Hof16b, BL16] has been widely investigated. However, we noted that tight security in broadcast encryption system has received less attention so far. Actually current broadcast encryption systems from bilinear groups are either not tightly secure or based on non-static assumptions.

Boneh, Gentry, and Waters [BGW05] give the first broadcast encryption scheme from bilinear maps under the selective security model (i.e., their construction does not capture the power of fully collusion resistant under adaptive attacks). This scheme has constant-size ciphertexts and constant-size secret keys (in terms of the number of users n), and a public key that is linear in n. However, their construction is based on q-type assumption, even though their reduction is tight.

Some other schemes with constant-size ciphertexts based on bilinear maps[DPP07, GW09] have been proven adaptively secure and/or are identity based, with the public parameters in all of these schemes is at least linear in the maximum number of recipients. However, their constructions are also based on q-type assumptions even though their schemes achieve tight security reductions. Recently Wee proposed a new broadcast encryption scheme [Wee16] under static assumptions (i.e., subgroup assumptions) using techniques derived from Déjà Q framework [CM14]. Furthermore, Wee's scheme has the same parameter sizes as BGW scheme. Nevertheless, we note that the reduction of Wee's scheme is not tight. More specifically, their construction suffers a $O(n)$ security loss during the reduction, where n is the number of users in the broadcast encryption system.

4 Preliminaries

We denote by $s \xleftarrow{\$} S$ the fact that s is picked uniformly at random from a finite set S. By PPT, we denote a probabilistic polynomial-time algorithm. We use 1^λ as the security parameter in unary throughout the context. We use $[n]$ to denote the set $\{1, \cdots, n\}$.

4.1 Composite-Order Bilinear Groups and Cryptographic Assumptions

We instantiate our system in composite-order bilinear groups. A generator $\mathcal{G}$ takes as input a security parameter λ and outputs a description $\mathbb{G} := (N, G, G_T, e)$, where N is product of distinct primes of $\Theta(\lambda)$ bits, G and G_T are cyclic groups of order N, and $e : G \times G \to G_T$ is a non-degenerate bilinear map. We require that the group operations in G and G_T as well the bilinear map e are computable in deterministic polynomial time. We consider bilinear groups whose orders N are products of four distinct primes p_1, p_2, p_3, p_4 such that $N = p_1p_2p_3p_4$. We can write $G = G_{p_1}G_{p_2}G_{p_3}G_{p_4}$ or $G = G_{p_1p_2p_3p_4}$ where G_{p_1}, G_{p_2}, G_{p_3} and G_{p_4} are subgroups of G of order p_1, p_2, p_3 and p_4 respectively. In addition, we use $G^*_{p_i}$ to denote $G_{p_i} \backslash \{1\}$. We will often write g_1, g_2, g_3, g_4 to denote random generators for the subgroups G_{p_1}, G_{p_2}, G_{p_3}, G_{p_4} respectively.

Cryptographic Assumptions. Our construction relies on the following four decisional subgroup assumptions. We define the following four advantage functions.

- **Assumption 1 (DS1)** For any PPT adversary $\mathcal{A}$ the following advantage is negligible in the security parameter λ.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DS1}}(\lambda) = |\Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, T_0) = 1] - \Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, T_1)]| \qquad (p_1 \to p_1p_2p_3)$$

where $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$, $g_1 \leftarrow G_{p_1}$, $g_4 \leftarrow G_{p_4}^*$,

$$T_0 \leftarrow G_{p_1}, T_1 \leftarrow G_{p_1p_2p_3}$$

- **Assumption 2 (DS2)** For any PPT adversary $\mathcal{A}$ the following advantage is negligible in the security parameter λ.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DS2}}(\lambda) = |\Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, T_0) = 1] - \Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, T_1)]| \qquad (p_1 \to p_1p_2)$$

where $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$, $g_1 \leftarrow G_{p_1}$, $g_4 \leftarrow G_{p_4}^*$, $X_1X_2X_3 \leftarrow G_{p_1p_2p_3}$,

$$T_0 \leftarrow G_{p_1}, T_1 \leftarrow G_{p_1p_2}$$

- **Assumption 3 (DS3)** For any PPT adversary $\mathcal{A}$ the following advantage is negligible in the security parameter λ.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DS3}}(\lambda) = |\Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, T_0) = 1] - \Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, T_1)]| \qquad (p_2 \to p_2p_3)$$

where $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$, $g_1 \leftarrow G_{p_1}$, $g_4 \leftarrow G_{p_4}^*$, $X_1X_2X_3 \leftarrow G_{p_1p_2p_3}$,

$$T_0 \leftarrow G_{p_2}, T_1 \leftarrow G_{p_2p_3}$$

- **Assumption 4 (DS4)** For any PPT adversary $\mathcal{A}$ the following advantage is negligible in the security parameter λ.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{DS4}}(\lambda) = |\Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, Y_2Y_4, T_0) = 1] - \Pr[\mathcal{A}(\mathbb{G}, g_1, g_4, X_1X_2X_3, Y_2Y_4, T_1)]| \qquad (p_2p_4 \to p_3p_4)$$

where $\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$, $g_1 \leftarrow G_{p_1}$, $g_4 \leftarrow G_{p_4}^*$, $X_1X_2X_3 \leftarrow G_{p_1p_2p_3}$, $Y_2Y_4 \leftarrow G_{p_2p_4}$,

$$T_0 \leftarrow G_{p_2p_4}, T_1 \leftarrow G_{p_3p_4}$$

4.2 Broadcast Encryption

A broadcast encryption scheme consists of three algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{Setup}(1^\lambda, 1^n) \to (\mathsf{mpk}, (\mathsf{sk}_1, \cdots, \mathsf{sk}_n))$. The setup algorithm gets as input the security parameter λ in unary and n, which is a polynomial in λ, in unary specifying the number of users and outputs the public parameter mpk, and secret keys $\mathsf{sk}_1, \cdots, \mathsf{sk}_n$.
- $\mathsf{Enc}(\mathsf{mpk}, S) \to (\mathsf{ct}, \mathsf{key})$. The encryption algorithm takes as input mpk and a subset $S \subseteq [n]$. It outputs a ciphertext ct (sometimes we call it ciphertext header) and a symmetric key $\mathsf{key} \in \{0,1\}^\lambda$. Note that given ct the set S is public.
- $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_i, \mathsf{ct}) \to \mathsf{key}$. The decryption algorithm gets as input mpk, a decryption key sk_i corresponding to identity $i \in [n]$ and ct. It outputs a symmetric key key if the identity i is in the set S.

Correctness. We require that for all $S \subseteq [n]$ and all $i \in [n]$ for which $i \in S$,

$$\Pr[(\mathsf{ct}, \mathsf{key}) \leftarrow \mathsf{Enc}(\mathsf{mpk}, S); \mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_i, \mathsf{ct}) = \mathsf{key}] = 1$$

where the probability is taken over $(\mathsf{mpk}, (\mathsf{sk}_1, \cdots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and the random coins used in Enc.

Selective Security Definition. A broadcast encryption scheme is *selectively secure* if for all PPT adversaries $\mathcal{A}$, the following advantage function is a negligible function in the security parameter λ.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{S-BE}}(\lambda) := \Pr\left[b = b' : \begin{array}{l} S^* \leftarrow \mathcal{A}(1^\lambda); \\ (\mathsf{mpk}, (\mathsf{sk}_1, \cdots, \mathsf{sk}_n)) \leftarrow \mathsf{Setup}(1^\lambda); \\ b \xleftarrow{\$} \{0,1\}; \mathsf{key}_1 \xleftarrow{\$} \{0,1\}^\lambda; \\ (\mathsf{ct}^*, \mathsf{key}_0) \leftarrow \mathsf{Enc}(\mathsf{mpk}, S^*); \\ b' \leftarrow \mathcal{A}(\mathsf{ct}^*, \mathsf{key}_b, \{\mathsf{sk}_i : i \notin S^*\}) \end{array} \right] - \frac{1}{2}$$

4.3 Core Lemma

We review the core lemma used by Chen and Wee as follows in a slightly different form in order to adapt to our construction of broadcast encryption.

Lemma 4.1 ([CM14, Wee16]). *Fix a prime p. For any (possibly unbounded) adversary $\mathcal{A}$ making at most q queries, we have*

$$\left| \Pr[\mathcal{A}^{\mathsf{F}_q(\cdot)}(1^q) = 1] - \Pr[\mathcal{A}^{\mathsf{RF}(\cdot)}(1^q) = 1] \right| \leq \frac{q^2}{p}$$

where we define oracles as follows

- F_q*: The oracle behaves as a function mapping from $\mathbb{Z}_p$ to $\mathbb{Z}_p$. It is initialized by picking the parameters $r_1, \cdots, r_q, \alpha_1, \cdots, \alpha_q \leftarrow \mathbb{Z}_p$. On input $x \in \mathbb{Z}_p$, it outputs*

  $$\sum_{i=1}^{q} r_i \alpha_i^x \in \mathbb{Z}_p$$

  Every query is answered using the same parameters $r_1, \cdots, r_q, \alpha_1, \cdots, \alpha_q$ we picked at the very beginning.

- $\mathsf{RF}(\cdot)$*: This oracle behaves as a truly random function $\mathsf{RF} : \mathbb{Z}_p \to \mathbb{Z}_p$. On input $x \in \mathbb{Z}_p$, it returns $\mathsf{RF}(x)$ if it has been defined, otherwise it returns $y \leftarrow \mathbb{Z}_p$ and defines $\mathsf{RF}(x) = y$.*

5 Our Broadcast Encryption Scheme

In this section we give our construction of tightly secure broadcast encryption and its security analysis.

5.1 Construction

Our broadcast encryption scheme $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ is described as follows.

- $\mathsf{Setup}(1^\lambda, 1^n)$: Compute $\mathbb{G} = (N, G, G_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

  $$(\alpha, \beta, g_1, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}^2 \text{ and } R_{4,k} \xleftarrow{\$} G_{p_4}, k = 1, \cdots, 2n$$

  and then compute

  $$u_k = u^{\alpha^k} R_{4,k}, k = 1, \cdots, 2n$$

Pick a pairwise hash function $\mathsf{H} : \{0,1\}^\lambda \rightarrow G_T$. It outputs the public parameters

$$\mathsf{mpk} = ((g_1, g_1^\beta, e(g_1, u_{n+1}), \mathsf{H}), g_1^\alpha, \cdots, g_1^{\alpha^n}, u_1, u_2, \cdots, u_n, u_{n+2}, \cdots, u_{2n})$$

and secret keys $\mathsf{sk}_i = u^{\alpha^{n-i+1}} R_{4.i}$ for $i \in [n]$, where $R_{4.i} \xleftarrow{\$} G_{p_4}$ for each $i \in [n]$.

- $\mathsf{Enc}(\mathsf{mpk}, S \subseteq [n])$: Sample $s \xleftarrow{\$} \mathbb{Z}_N$. It computes

  $$\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1) = (g_1^s, g_1^{(\beta + \sum_{k\in S} \alpha^k)s})$$

  and

  $$\mathsf{key} = \mathsf{H}(e(g_1, u^{\alpha^{n+1}})^s)$$

  Finally it outputs the pair $(\mathsf{ct}, \mathsf{key})$.

- $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_i, \mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1))$: It outputs

  $$\mathsf{key} = \frac{e(\mathsf{ct}_1, u_{n-i+1})}{e(\mathsf{ct}_0, \mathsf{sk}_i \prod_{k\in S, k\neq i} u_{n+1+k-i})}$$

  .

Correctness. It is easy to show the correctness of our scheme through the following equations.

$$\begin{aligned}\frac{e(\mathsf{ct}_1, u_{n-i+1})}{e(\mathsf{ct}_0, \mathsf{sk}_i \prod_{k\in S, k\neq i} u_{n+1+k-i})} &= \frac{e(g_1^{(\beta+\sum_{k\in S}\alpha^k)s}, u_{n-i+1})}{e(g_1^s, u^{\alpha^{n-i+1}} \cdot \prod_{k\in S, k\neq i} u_{n+1+k-i})} \\ &= \frac{e(g_1^{(\beta+\sum_{k\in S}\alpha^k)s}, u^{\alpha^{n-i+1}})}{e(g_1^s, u^{\alpha^{n-i+1} + \sum_{k\in S, k\neq i} \alpha^{n+1+k-i}})} \\ &= e(g_1, u^{\alpha^{n+1}})^s \\ &= \mathsf{key}\end{aligned}$$

5.2 Security Analysis

Theorem 5.1. *For any PPT adversary $\mathcal{A}$, there exists an algorithm $\mathcal{B} = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$ such that*

$$\boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{S-BE}}(\lambda) \leq \boldsymbol{Adv}_{\mathcal{B}_1}^{\mathsf{DS1}} + \boldsymbol{Adv}_{\mathcal{B}_2}^{\mathsf{DS2}}(\lambda) + O(\log\lambda) \cdot (\boldsymbol{Adv}_{\mathcal{B}_3}^{\mathsf{DS3}}(\lambda) + \boldsymbol{Adv}_{\mathcal{B}_4}^{\mathsf{DS4}}(\lambda)) + 2^{-\Omega(\lambda)}$$

Proof. Firstly we define the advantage function of any PPT adversary $\mathcal{A}$ in hybrid $\mathsf{Hyb}_{x,y,z}$ as

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{x,y,z}}(\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|$$

We prove the theorem using the following hybrid arguments.

Hyb_0: is the real experiment as defined in the selective security model.

Hyb_1: This is the same as Hyb_0, except that the system generates the random parameter $\tilde{\beta}$ such that $\beta = \tilde{\beta} - \sum_{k\in S^*} \alpha^k$. More concretely, in this hybrid $u_1, \cdots, u_{2n}$ and mpk are computed as in the honest setup, but the challenge ciphertext ct^* is computed as $(g_1^s, (g_1^s)^{\tilde{\beta}})$ and the symmetric key key_0 is computed as $\mathsf{H}(e(g_1^s, u_{n+1}))$. To simulate the secret keys $\{\mathsf{sk}_i : i \notin S^*\}$, we compute each $\mathsf{sk}_i = u_{n-i+1}^{\tilde{\beta}} \cdot (\prod_{k\in S^*, k\neq i} u_{n+1+k-i})^{-1} \cdot R_{4.i}$

using the parameter $\tilde{\beta}$ and $(u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n})$. Since Hyb_0 and Hyb_1 are identically distributed, we have $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_0} = \mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_1}$.

$\underline{\mathsf{Hyb}_2}$: This is the same as Hyb_1, except that we change the distribution of $(\mathsf{ct}^*, \mathsf{key}_0^*)$. Namely

$$(\mathsf{ct}^*, \mathsf{key}_0^*) = \Big((T^s, (T^s)^{\tilde{\beta}}), \mathsf{H}(e(T^s, u_{n+1}))\Big)$$

where T is sampled from $G_{p_1 p_2 p_3}$.

Lemma 5.1. *Under the decisional subgroup assumption* $\mathsf{DS1}$*, we have*

$$\left|\boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_1}(\lambda) - \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_2}(\lambda)\right| \leq \boldsymbol{Adv}_{\mathcal{B}_1}^{\mathsf{DS1}}(\lambda)$$

Proof. Given $(\mathbb{G}, g_1, g_4, T)$ where either $T \leftarrow G_{p_1}$ or $T \leftarrow G_{p_1 p_2 p_3}$, the algorithm $\mathcal{B}_1$ works as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a challenge set $S^* \subseteq [n]$.
- **Setup Phase**: Pick parameters $(\alpha, \tilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, $\beta = \tilde{\beta} - \sum_{k \in S^*} \alpha^k$. Select the hash function H, all randomness $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k = 1, \cdots, 2n$ and $R_4.i \xleftarrow{\$} G_{p_4}$ for $i = 1, \cdots, n$. It outputs the public parameters
$$\mathsf{mpk} = ((g_1, g_1^{\beta}, e(g_1, u_{n+1}), \mathsf{H}), g_1^{\alpha}, \cdots, g_1^{\alpha^n}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n})$$
and outputs secret keys $(\mathsf{sk}_1, \cdots, \mathsf{sk}_n)$. Note that we compute the set of secret keys $\{\mathsf{sk}_i : i \notin S^*\}$, in which sk_i is computed as $u_{n-i+1}^{\tilde{\beta}} \cdot (\prod_{k \in S^*, k \neq i} u_{n+1+k-i})^{-1} \cdot R_{4.i}$. Otherwise the secret key sk_i is computed normally as $u^{\alpha^{n-i+1}} R_{4.i}$.
- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$, and compute the ciphertext header $\mathsf{ct}^* = (\mathsf{ct}_0^* = T^{s'}, \mathsf{ct}_1^* = (T^{s'})^{\tilde{\beta}})$ and the symmetric key $\mathsf{key}_0^* = \mathsf{H}(e(T^{s'}, u_{n+1}))$. The reduction $\mathcal{B}_1$ picks random $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^{\lambda}$ and return $(\mathsf{ct}^*, \mathsf{key}_b^*)$, where b is a random coin.
- **Guess**: The reduction $\mathcal{B}_1$ returns 1 if $b = b'$. Otherwise it returns 0.

Observe that when $T \leftarrow G_{p_1}$, the simulation is identical to Hyb_1; when $T \leftarrow G_{p_1 p_2 p_3}$, the simulation is identical to Hyb_2. This proves the lemma. □

$\underline{\mathsf{Hyb}_{3.i}}$: for $0 \leq i \leq \lceil \log n + 1 \rceil$, we change the distribution of $u_1, \cdots, u_{2n}$ from $u^{\alpha^k} R_{4.k}$ to $u^{\alpha^k} g_2^{\sum_{j=1}^{2^i} r_j \alpha_j^k} R_{4.k}$, where $r_j, \alpha_j \xleftarrow{\$} \mathbb{Z}_N$, $j \in [2^i]$ and $g_2 \xleftarrow{\$} G_{p_2}$.

Lemma 5.2. *Under the decisional subgroup assumption* $\mathsf{DS2}$*, we have*

$$\left|\boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.0}}(\lambda) - \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_2}(\lambda)\right| \leq \boldsymbol{Adv}_{\mathcal{B}_2}^{\mathsf{DS2}}(\lambda)$$

Proof. Given $(\mathbb{G}, g_1, g_4, X_1 X_2 X_3, T)$, where either $T = u \leftarrow G_{p_1}$ or $T = u g_2^r \leftarrow G_{p_1 p_2}$. The algorithm $\mathcal{B}_2$ works as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a challenge set $S^* \subseteq [n]$.
- **Setup Phase**: Sample parameters $(\alpha, \tilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$ and $\beta = \tilde{\beta} - \sum_{k \in S^*} \alpha^k$. Select a hash function H and sample randomness from the group G_{p_4}. That is $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k = 1, \cdots, 2n$ and $R_{4.i} \xleftarrow{\$} G_{p_4}$ for $i = 1, \cdots, n$. Compute each u_k as $T^{\alpha^k} R_{4.k}$ for each $k \in [2n]$. Proceed as Hyb_2 to compute the public parameters mpk and $\{\mathsf{sk}_i : i \notin S^*\}$.

- **Challenge Phase**: Sample a random value $s' \xleftarrow{\$} \mathbb{Z}_N$, it uses $X_1X_2X_3$ as provided and u_{n+1} as computed above to compute

$$\mathsf{ct}^* = ((X_1X_2X_3)^{s'}, ((X_1X_2X_3)^{s'})^{\tilde{\beta}}), \mathsf{key}_0^* = \mathsf{H}(e((X_1X_2X_3)^{s'}, u_{n+1}))$$

- **Guess**: The reduction $\mathcal{B}_2$ returns 1 if $b = b'$. Otherwise it returns 0.

Observe that if $T = u$, the simulation is identical to Hyb_2; when $T = ug_2^r$, the simulation is identical to $\mathsf{Hyb}_{3.0}$. This proves the lemma. □

In order to prove $\mathsf{Hyb}_{3.i}$ is computationally indistinguishable from $\mathsf{Hyb}_{3.i+1}$ for all $i \in [1, \cdots, \lceil \log n + 1 \rceil]$, we construct the following two sub-hybrid arguments.

- $\underline{\mathsf{Hyb}_{3.i.1}}$: is equivalent to $\mathsf{Hyb}_{3.i}$, except that we change the distribution of $u_1, \cdots, u_{2n}$ as

$$u_k = u^{\alpha^k} g_2^{\sum_{j=1}^{2^i} r_j \alpha_j^k} \cdot g_3^{\sum_{j=1}^{2^i} \hat{r}_j \hat{\alpha}_j^k} R_{4.k}, \quad k = 1, \cdots, 2n$$

- $\underline{\mathsf{Hyb}_{3.i.2}}$: is equivalent to $\mathsf{Hyb}_{3.i.1}$, except that we change the distribution of $u_1, \cdots, u_{2n}$ as

$$u_k = u^{\alpha^k} g_2^{\sum_{j=1}^{2^i} r_j \alpha_j^k + \sum_{j=1}^{2^i} \hat{r}_j \hat{\alpha}_j^k} R_{4.k}, \quad k = 1, \cdots, 2n$$

Lemma 5.3. *Under the decisional subgroup assumption* DS3*, for each* $i \in [1, \cdots, \lceil \log n + 1 \rceil]$*, we have*

$$\left| \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i}}(\lambda) - \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.1}}(\lambda) \right| \leq \boldsymbol{Adv}_{\mathcal{B}_3}^{\mathsf{DS3}}(\lambda)$$

Proof. Given parameters $(\mathbb{G}, g_1, g_4, X_1X_2X_3, T)$ where either $T = g_2 \leftarrow G_{p_2}$ or $T = g_2g_3 \leftarrow G_{p_2p_3}$. The algorithm $\mathcal{B}_3$ works as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a challenge set $S^* \subseteq [n]$.
- **Setup Phase**: Pick parameters $(\alpha, \tilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_2}$, $\beta = \tilde{\beta} - \sum_{k \in S^*} \alpha^k$. Select the hash function H and all randomness to be used $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k = 1, \cdots, 2n$ and $R_{4.i} \xleftarrow{\$} G_{p_4}$ for $i = 1, \cdots, n$. Sample $\alpha'_1, \cdots, \alpha'_{2^i}, r'_1, \cdots, r'_{2^i} \xleftarrow{\$} \mathbb{Z}_N$. Compute each u_k as $u^{\alpha^k} \cdot T^{\sum_{j=1}^{2^i} r'_j \alpha_j'^k} R_{4.k}$. Finally it proceeds as Hyb_2 using $\alpha, u_1, \cdots, u_{2n}$ as computed above to compute the public parameters mpk and $\{\mathsf{sk}_i : i \notin S^*\}$.
- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$ and compute $\mathsf{ct}_0^* = (X_1X_2X_3)^{s'}$, $\mathsf{ct}_1^* = (X_1X_2X_3)^{s' \cdot \tilde{\beta}}$ and the symmetric key $\mathsf{key}_0^* = \mathsf{H}(e((X_1X_2X_3)^{s'}, u_{n+1}))$. The reduction $\mathcal{B}_3$ then picks $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ and returns $(\mathsf{ct}^* = (\mathsf{ct}_0^*, \mathsf{ct}_1^*), \mathsf{key}_b^*)$ where b is the random coin flipped by the challenger.
- **Guess**: The reduction $\mathcal{B}_3$ outputs 1 if $b = b'$. Otherwise it returns 0.

When $T = g_2$, the simulation is identical to $\mathsf{Hyb}_{3.i}$; when $T = g_2g_3$, the simulation is identical to $\mathsf{Hyb}_{3.i.1}$, where for all $j \in [2^i]$ we have $\alpha_j = \alpha'_j \mod p_2$, $r_j = r'_j \mod p_2$, $\hat{\alpha}_j = \alpha'_j \mod p_3$ and $\hat{r}_j = r'_j \mod p_3$. This proves the lemma. □

Lemma 5.4. *Under the decisional subgroup assumption* DS4*, for each* $i \in [1, \cdots, \lceil \log n + 1 \rceil]$*, we have*

$$\left| \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.1}}(\lambda) - \boldsymbol{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.2}}(\lambda) \right| \leq \boldsymbol{Adv}_{\mathcal{B}_4}^{\mathsf{DS4}}(\lambda)$$

Proof. Given $(\mathbb{G}, g_1, g_4, X_1X_2X_3, Y_2Y_4, T)$ where either $T = g_2R_4 \xleftarrow{\$} G_{p_2p_4}$ or $T = g_3R_4 \xleftarrow{\$} G_{p_3p_4}$, the algorithm $\mathcal{B}_4$ works as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to the challenge set $S^* \subseteq [n]$.
- **Setup Phase**: Sample parameters $(\alpha, \tilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, $\beta = \tilde{\beta} - \sum_{k \in S^*} \alpha^k$. Select the hash function H and sample all randomness to be used, $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k = 1, \cdots, 2n$ and $R_{4.i} \xleftarrow{\$} G_{p_4}$ for $i = 1, \cdots, n$. Sample $\alpha'_1, \cdots, \alpha'_{2^i}, r'_1, \cdots, r'_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i} \xleftarrow{\$} \mathbb{Z}_N$. Compute each u_k as

  $$u^{\alpha^k} \cdot (Y_2Y_4)^{\sum_{j=1}^{2^i} r'_j \alpha'^k_j} \cdot T^{\sum_{j=1}^{2^i} \hat{r}'_j \hat{\alpha}'^k_j} \cdot R_{4.k}$$

  It proceeds to compute the public parameters mpk and the set of secret keys $\{\mathsf{sk}_i : i \notin S^*\}$ as in Hyb_2.
- **Challenge Phase**: Sample a random value $s' \xleftarrow{\$} \mathbb{Z}_N$ and compute $\mathsf{ct}_0^* = (X_1X_2X_3)^{s'}$, $\mathsf{ct}_1^* = (X_1X_2X_3)^{s' \cdot \tilde{\beta}}$ and $\mathsf{key}_0^* = \mathsf{H}(e(X_1X_2X_3)^{s'}, u_{n+1})$, where u_{n+1} is computed as above. The reduction $\mathcal{B}_4$ picks $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ and returns $(\mathsf{ct}^* = (\mathsf{ct}_0^*, \mathsf{ct}_1^*), \mathsf{key}_b^*)$, where b is a random bit flipped by the challenger.
- **Guess**: The reduction $\mathcal{B}_4$ returns 1 if $b = b'$ and returns 0 in the other case.

Observe that if $T = g_3R_4$, the simulation is identical to $\mathsf{Hyb}_{3.i.1}$ and if $T = g_2R_4$, then the simulation is identical to $\mathsf{Hyb}_{3.i.2}$. □

Moreover, it is easy to notice that $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.2}}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.2}}(\lambda)$ since all r_j and all r'_j are i.i.d variables in $\mathsf{Hyb}_{3.i.2}$, by setting $\alpha_{2^i+j} = \hat{\alpha}_j$, $r_{2^i+j} = \hat{r}_j$, for all $j \in [2^i]$.

$\underline{\mathsf{Hyb}_4}$: is equivalent to $\mathsf{Hyb}_{3.\lceil \log n+1 \rceil}$, except that for each $k \in [2n]$, each u_k is in the form

$$u^{\alpha^k} g_2^{\mathsf{RF}(k)} R_{4.k}$$

where $g_2 \leftarrow G_{p_2}$ and RF is a truly random function. In this hybrid argument, the challenger computes the challenge ciphertext $\mathsf{ct}^* = \Big((X_1X_2X_3)^s, (X_1X_2X_3)^{s\tilde{\beta}}\Big)$ and the symmetric key

$$\begin{aligned}\mathsf{key}_0^* &= e((X_1X_2X_3)^s, u_{n+1}) \\ &= e((X_1X_2X_3)^s, u^{\alpha^{n+1}} g_2^{\mathsf{RF}(n+1)} R_{4.n+1}) \\ &= e((X_1X_2X_3)^s, u^{\alpha^{n+1}}) \cdot e((X_1X_2X_3)^s, g_2^{\mathsf{RF}(n+1)})\end{aligned}$$

has $\log p_2 = \Theta(\lambda)$ bits of min-entropy coming from $\mathsf{RF}(n+1)$; this holds as long as the G_{p_2}-component of $(X_1X_2X_3)^s$ is not 1, which happens with probability $1 - 1/p_2$. Therefore, by the core lemma described as lemma 4.1, we have

$$\left| \mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.\lceil \log n+1 \rceil}}(\lambda) - \mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_4}(\lambda) \right| \leq 2^{-\Omega(\lambda)}$$

This proves the main theorem. □

Acknowledgement. The author wants to thank Jie Chen for insightful discussions.

References

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology–CRYPTO 2005*, pages 258–275. Springer, 2005.

[BL16] Xavier Boyen and Qinyi Li. Towards tightly secure short signature and ibe. Cryptology ePrint Archive, Report 2016/498, 2016. http://eprint.iacr.org/2016/498.

[Che16] Jie Chen. Tightly secure ibe under constant-size master public key. Cryptology ePrint Archive, Report 2016/891, 2016. http://eprint.iacr.org/2016/891.

[CM14] Melissa Chase and Sarah Meiklejohn. Déja q: Using dual systems to revisit q-type assumptions. In *Advances in Cryptology–EUROCRYPT 2014*, pages 622–639. Springer, 2014.

[CW13] Jie Chen and Hoeteck Wee. Fully,(almost) tightly secure ibe and dual system groups. In *Advances in Cryptology–CRYPTO 2013*, pages 435–460. Springer, 2013.

[DPP07] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *Pairing-Based Cryptography–Pairing 2007*, pages 39–59. Springer, 2007.

[FN94] Amos Fiat and Moni Naor. Broadcast encryption. In *Advances in Cryptology—CRYPTO'93*, pages 480–491. Springer, 1994.

[GDCC16] Junqing Gong, Xiaolei Dong, Jie Chen, and Zhenfu Cao. Efficient ibe with tight reduction to standard assumption in the multi-challenge setting. Cryptology ePrint Archive, Report 2016/860, 2016. http://eprint.iacr.org/2016/860.

[GW09] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In *Advances in Cryptology-EUROCRYPT 2009*, pages 171–188. Springer, 2009.

[Hof16a] Dennis Hofheinz. Adaptive partitioning. 2016.

[Hof16b] Dennis Hofheinz. Algebraic partitioning: Fully compact and (almost) tightly secure cryptography. In *Theory of Cryptography Conference*, pages 251–281. Springer, 2016.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In *Advances in Cryptology-CRYPTO 2009*, pages 619–636. Springer, 2009.

[Wee16] Hoeteck Wee. Déjà q: Encore! un petit ibe. In *Theory of Cryptography*, pages 237–258. Springer, 2016.