# Collusion-Resistant Broadcast Encryption with Tight Reductions and Beyond

**Abstract**

The issue of tight security in identity-based encryption scheme (IBE) has been widely investigated. Recently, a tightly secure IBE scheme of bilinear groups in the weak multi-challenge setting has been achieved by Chen (eprint 2016/891), and their scheme even achieves constant public parameters and is adaptively secure. However, we note that the issue of tight security in broadcast encryption scheme (BE) of bilinear groups has received less attention so far. Actually current broadcast encryption systems of bilinear groups are either not tightly secure or based on non-static assumptions. In this work we mainly focus on the issue of tight security in the standard broadcast encryption scheme.

We present the first *tightly secure* broadcast encryption scheme (BE) from the static assumptions (i.e., decisional subgroup assumptions) in the selective security model. Our construction and proof rely on the recently novel techniques from Chen (eprint 2016/891). The proof of our construction will lead to only $O(\log n)$ or $O(\log \lambda)$ security loss, where $n$ is the number of users in the system and $\lambda$ is the security parameter, rather than $O(n)$ security loss as the construction given by Wee (TCC-A 16) and many other previous constructions.

Following this result, we present the first *tightly secure* non-zero inner product encryption scheme (NIPE) from decisional subgroup assumptions in the selective security model. This NIPE scheme has the same parameter sizes as our BE scheme and there is only $O(\log n)$ or $O(\log \lambda)$ security loss as well, where $n$ is the dimension of inner product space and $\lambda$ is the security parameter. This result improves the most optimal NIPE scheme so far from static assumptions recently proposed by Chen et al.(SCN 16), which also suffers $O(n)$ security loss during the reduction.

Finally, we further present the first functional commitment scheme (FC) for linear functions with *tight reductions*, also from decisional subgroup assumptions. Recently Libert et al.(ICALP 16) introduced the notion of functional commitment scheme for linear functions. However, their scheme also suffers $O(n)$ security loss during the reduction. In contrast, there is only $O(\log n)$ or $O(\log \lambda)$ security loss in our FC scheme, which significantly improves the result of Libert et al.

## 1 Introduction

The notion of broadcast encryption was first introduced by Fiat and Naor[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 $\epsilon$ implies breaking some computationally hard problem in time roughly $T$ with success probability $\epsilon/L$. We call $L$ as the security loss and a tight reduction is one where $L$ is only related to the security parameter $\lambda$ 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.

In this paper, our main goal is to design *tightly secure* broadcast encryption schemes from well-studied hardness assumptions, which simultaneously provides optimal parameters (e.g., constant-size ciphertext overhead, and constant-size private keys). Furthermore, we will design *tightly secure* non-zero inner product encryption scheme and *tightly secure* functional commitment scheme for linear functions based on techniques used in our design of broadcast encryption scheme.

## 2 Our Contributions

In this paper, we made the following contributions.

1. 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 $\lambda$ is the security parameter. This result 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]. This result can be regarded as a combination of these two works.

2. In addition, inspired by the observation that our BE construction is essentially obtained by tweaking the BGW scheme and Wee's broadcast encryption, we apply Chen's techniques [Che16] to the NIPE scheme proposed by Chen et al.[CLR16] and the functional commitment scheme proposed by Libert et al.[LRY16]. Then we obtain the following two results.

    - We present the first *tightly secure* non-zero inner product encryption scheme (NIPE) from decisional subgroup assumptions in the selective security model. This NIPE scheme has the same parameter sizes as our BE scheme and there is only $O(\log n)$ or $O(\log \lambda)$ security loss as well, where $n$ is the dimension of inner product space and $\lambda$ is the security parameter. In contrast, the NIPE scheme proposed by Chen et al. suffers $O(n)$ security loss during the reduction.
    - We further present the first functional commitment scheme (FC) for linear functions with *tight reductions* from decisional subgroup assumptions. This scheme has commitments and openings of constant size and is perfectly hiding. Furthermore, it suffers only $O(\log n)$ or $O(\log \lambda)$ security loss rather than $O(n)$ security loss in the work of Libert et al., where $n$ is the dimension of the domain of linear functions and $\lambda$ is the security parameter. [1]

[1]We remark that this functional commitment also implies many other kinds of commitment schemes and cryptographic accumulators for large universes. We refer the reader to the work of Libert et al.[LRY16]

## 3 Outline of Our Constructions and Proofs

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

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

where $(g, u) \leftarrow G_{p_1}$, $\alpha, \beta, s \leftarrow \mathbb{Z}_N$, $R_{4.i} \leftarrow G_{p_4}$, $S$ is the challenge set and $\mathsf{H}$ is a pairwise-independent hash function. 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 $\mathsf{RF}(\cdot)$. 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 $\alpha$ in the polynomial $\beta + \sum_{k\in S}\alpha^k$ which shows up in the ciphertext. Due to this restriction, we need to generate $\beta$ 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

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

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

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

Thus the monomials in $\alpha$ 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 $\alpha$ 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 $\beta$, 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 $F_{2^i}(i \le \lceil \log q \rceil)$ and then inject them back into $F_{2^{i+1}}$. 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 and this is the key step leading to achieve tight reductions. To temporarily store the entropy extracted from each $F_{2^i}$, Chen 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), (2) and (3), 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:

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

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

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

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 $F_{2^i} (i \leq \lceil \log n + 1 \rceil)$ itself and then inject them back into $F_{2^{i+1}}$. Therefore, we only need $\lceil \log n + 1 \rceil$ 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 $\alpha$ 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 change the distribution of the set of parameters $\{u_k\}_{k\in[2n]}$ with the help of semi-functional space $G_{p_2}$ and the shadow semi-functional space $G_{p_3}$. 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 $\widehat{F_{2^i}}'(k) = \sum_{j=1}^{2^i} r_j\alpha_j^k + \hat{r}_j\hat{\alpha}_j^k$ is another new function required in the reduction and $r_1, \cdots, r_{2^i}, \alpha_1, \cdots, \alpha_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_2}$.

**Our NIPE Scheme and FC Scheme**. The construction of our NIPE scheme and our FC scheme are essentially the same as the NIPE scheme proposed by Chen et al. [CLR16] and the FC scheme proposed by Libert et al.[LRY16], except that our schemes are working on the bilinear group $G$ of order $N = p_1p_2p_3p_4$ instead of $N = p_1p_2p_3$. More specifically, the public parameters, master secret key, secret key, ciphertext and symmetric key in our NIPE scheme are in the form as follows:

$$\begin{aligned}
\mathsf{mpk} &: (g, g^{\beta}, \{g_j\}_{j\in[n]}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n}, \mathsf{H}) \\
\mathsf{msk} &: (u, \alpha, \beta, R_4) \\
\mathsf{sk}_{\vec{y}} &: \left(\prod_{i=1}^{n} u^{\alpha^i y_i}\right)^{\beta} \cdot R_4' \\
\mathsf{ct} &: \left(g^s, g^{(\beta+\sum_{i=1}^{n} x_i\alpha^i)s}\right) \\
\mathsf{key} &: \mathsf{H}(e(g, u^{n+1})^s)
\end{aligned}$$

where $(g, u, \alpha, \beta, s) \xleftarrow{\$} G_{p_1}^2 \times \mathbb{Z}_N^3$, $R_4, R_4' \xleftarrow{\$} G_{p_4}$, $\vec{x} = (x_1, \cdots, x_n), \vec{y} = (y_1, \cdots, y_n) \in \mathbb{Z}_N^n$ and $\mathsf{H}$ is a pairwise-independent hash function.

Furthermore, in our FC scheme, the commitment key, trapdoor key, commitment and the witness for the linear function are in the form as follows

$$\begin{aligned}
\mathsf{mpk} &: (g, \{g_j\}_{j\in[n]}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n}, R_4) \\
\mathsf{tk} &: u_{n+1} \\
\mathsf{C} &: g^{\beta+\sum_{i=1}^{n} m_j\alpha_j} \\
\mathsf{W}_y &= \prod_{i=1}^{n} \mathsf{W}_i^{x_i}, \qquad \mathsf{W}_i = u_{n-i+1}^{\beta} \cdot \prod_{j=1, j\neq i}^{n} u_{n+j-i+1}^{m_j}, \qquad \forall i \in [n]
\end{aligned}$$

where $(g, u, \alpha, \beta) \xleftarrow{\$} G_{p_1}^2 \times \mathbb{Z}_N^2$, $R_4 \xleftarrow{\$} G_{p_4}$, $\vec{m} = (m_1, \cdots, m_n)$ is the message vector, and $y$ is the result of linear function evaluating over the message vector $\vec{m}$.

To establish tight reductions of our NIFE scheme and FC scheme, we essentially follow the same s-trategy as the proof of our broadcast encryption scheme. Namely, we introduce the three kinds of function families $\{F_{2^i}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$, $\{\widehat{F_{2^i}}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ and $\{\widehat{F_{2^i}}'(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ defined as above to modify the distribution of $\{u_k\}_{k\in[2n]}$ with the help of the semi-functional space $G_{p_2}$ and the shadow semi-functional space $G_{p_3}$. We refer the reader to our security proofs in appendix A and appendix B for more details.

## 4 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.

Following the framework used in Wee's broadcast encryption scheme, Libert et al. [LRY16] applied the same framework to construct functional commitment schemes for linear functions and Chen et al. [CLR16] also utilized this framework to construct NIPE schemes with short ciphertexts and short private keys. However, since they both applied Wee's framework, their scheme both suffer a $O(n)$ security loss as well, where $n$ is the dimension of inner product space and the dimension of the domain of linear functions, and hence they are not tight.

## 5 Paper Organization

In section 6, we introduce preliminaries to be used in this work. In section 7, we give the description of our BE scheme and its security analysis. We give the description of our NIPE scheme and our FC scheme and their security analysis respectively in section 8 and section 9. Since the security analysis of our NIPE scheme and FC scheme lies in a similar strategy as our BE scheme, we leave their formal proofs in appendix A and appendix B respectively.

## 6 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 $1^\lambda$ as the security parameter in unary throughout the context. We use $[n]$ to denote the set $\{1, \cdots, n\}$.

### 6.1 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 $\lambda$ 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 $\lambda$.

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

  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 $\lambda$.

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

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 $\lambda$.

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

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 $\lambda$.

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

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

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

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

**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 $\lambda$.

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

where the probability is defined over random coins used by $\mathsf{Setup}$ and $\mathsf{Enc}$, and the adversary $\mathcal{A}$ as well as the random bit $b$.

## 6.3 Non-Zero Inner Product Encryption (NIPE)

We give the definition of non-zero inner product encryption scheme in this section. For convenience we give the definition in the key encapsulation setting.

Let $\mathcal{V}$ denote an inner product space of dimension $n$. A non-zero inner product encryption (NIPE) scheme for inner products over the space $\mathcal{V}$ is defined by four probabilistic algorithms ($\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec}$).

- $\mathsf{Setup}(1^\lambda, 1^n)$: takes as input a security parameter $\lambda$ in unary and the dimension of $\mathcal{V}$ in unary. It outputs the public parameters $\mathsf{mpk}$ and the master secret key $\mathsf{msk}$.
- $\mathsf{KeyGen}(\mathsf{msk}, \vec{y})$: takes as input the master secret key $\mathsf{msk}$ and a vector $\vec{y}$ in the space $\mathcal{V}$. It outputs a secret key $\mathsf{sk}_{\vec{y}}$ for the vector $\vec{y}$.
- $\mathsf{Enc}(\mathsf{mpk}, \vec{x})$: takes as input an attribute vector $\vec{x} \in \mathcal{V}$ and the public parameters $\mathsf{mpk}$. It outputs a ciphertext $\mathsf{ct}$ and a symmetric key $\mathsf{key} \in \{0,1\}^\lambda$. The symmetric key $\mathsf{key}$ is used to encrypt any message through any symmetric key encryption scheme.
- $\mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}, \mathsf{sk}_{\vec{y}})$: If $\langle \vec{x}, \vec{y} \rangle \neq 0$, this algorithm returns the symmetric key $\mathsf{key}$. Otherwise it outputs $\perp$.

**Correctness**. A NIPE scheme is correct if for all vectors $\vec{x}, \vec{y} \in \mathcal{V}$ satisfying $\langle \vec{x}, \vec{y} \rangle \neq 0$, any key pair $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $\mathsf{sk}_{\vec{y}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \vec{y})$ and any $(\mathsf{ct}, \mathsf{key}) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \vec{x})$, we have $\Pr[\mathsf{key} = \mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}, \mathsf{sk}_{\vec{y}})] = 1$.

**Selective Security of NIPE**. A non-zero inner product encryption scheme is *selectively secure* if for all PPT adversaries $\mathcal{A}$, the following advantage function is a negligible function in the security parameter $\lambda$.

$$\mathbf{Adv}_{\mathcal{A}}^{\mathsf{S-NIPE}}(\lambda) := \Pr\left[ b = b' : \begin{array}{l} \vec{x}^* \leftarrow \mathcal{A}(1^\lambda, 1^n); \\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda); \\ b' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\cdot), \mathsf{Enc}(\mathsf{mpk}, b, \cdot)}(1^\lambda, 1^n, \mathsf{mpk}) \end{array} \right] - \frac{1}{2}$$

where the oracles $\mathsf{KeyGen}(\cdot)$ and $\mathsf{Enc}(\mathsf{mpk}, b, \cdot)$ are defined as follows.

1. $\mathsf{KeyGen}(\cdot)$: On input a vector $\vec{y} \in \mathcal{V}$, the oracle returns $\mathsf{KeyGen}(\mathsf{msk}, \vec{y})$.
2. $\mathsf{Enc}(\mathsf{mpk}, b, \cdot)$: On input the committed vector $\vec{x}^*$, it computes $(\mathsf{ct}^*, \mathsf{key}_0^*) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \vec{x}^*)$ and then samples a symmetric key $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ uniformly at random. It returns $(\mathsf{ct}^*, \mathsf{key}_b^*)$. If the input vector is not the committed vector, it aborts and output $\perp$.

the probability of the security game is defined over random coins used by $\mathsf{Setup}$ and oracles $\mathsf{KeyGen}$ and $\mathsf{Enc}$, and the adversary $\mathcal{A}$ as well as the random bit $b$.

## 6.4 Functional Commitments for Linear Functions

**Definition 6.1** (Functional Commitments)**.** A functional commitments scheme ($\mathsf{FC}$) for the linear function $(\mathcal{D}, n, \langle\cdot,\cdot\rangle)$, where $\mathcal{D}$ is the domain for the linear functions $\langle\cdot,\cdot\rangle : \mathcal{D}^n \times \mathcal{D}^n \to \mathcal{D}$ defined by $\langle\vec{x},\vec{m}\rangle = \sum_{i=1}^{n} x_i m_i$ for $\vec{x},\vec{m} \in \mathcal{D}^n$ with $\vec{x} = (x_1,\cdots,x_n), \vec{m} = (m_1,\cdots,m_n)$, is a tuple of four polynomial time algorithms $(\mathsf{Setup},\mathsf{Commit},\mathsf{Open},\mathsf{Vrf})$.

- $\mathsf{Setup}(1^\lambda, 1^n) \to (\mathsf{ck},\mathsf{tk})$: takes as input a security parameter $\lambda \in \mathbb{N}$ in unary, a desired message length $n = \mathsf{poly}(\lambda)$. It outputs a commitment key $\mathsf{ck}$ and a trapdoor key $\mathsf{tk}$.
- $\mathsf{Commit}(\mathsf{ct},\vec{m}) \to (\mathsf{C},\mathsf{aux})$: takes as input the commitment key $\mathsf{ck}$, a message vector $\vec{m} \in \mathcal{D}^n$ and outputs a commitment $\mathsf{C}$ for the message $\vec{m}$ and $\mathsf{aux}$, which is the auxiliary information.
- $\mathsf{Open}(\mathsf{ck},\mathsf{C},\mathsf{aux},\vec{m}) \to \mathsf{W}_y$: takes as input the commitment key $\mathsf{ck}$, a commitment $\mathsf{C}$ corresponding to the message $\vec{m}$, auxiliary information $\mathsf{aux}$ and a vector $\vec{x} \in \mathcal{D}^n$. It outputs a witness $\mathsf{W}_y$ for $y = \langle\vec{x},\vec{m}\rangle$. Namely, $\mathsf{W}_y$ is a witness for the fact that the linear function defined by $\vec{x}$ when evaluated on $\vec{m}$ gives $y$.
- $\mathsf{Vrf}(\mathsf{ck},\mathsf{C},\mathsf{W}_y,\vec{x},y) \to 0/1$: takes as input the commitment key $\mathsf{ck}$, a commitment $\mathsf{C}$, a witness $\mathsf{W}_y$, a vector $\vec{x} \in \mathcal{D}^n$ and $y \in \mathcal{D}$. It outputs 1 if $\mathsf{W}_y$ is a witness for $\mathsf{C}$ being a commitment for some $\vec{m} \in \mathcal{D}^n$ such that $\langle\vec{x},\vec{m}\rangle = y$ and outputs 0 otherwise.

**Correctness**. We require that for every $(\mathsf{ck},\mathsf{tk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, for all $\vec{m},\vec{x} \in \mathcal{D}^n$, if $(\mathsf{C},\mathsf{aux}) \leftarrow \mathsf{Commit}(\mathsf{ck},\vec{m})$ and $\mathsf{W}_y \leftarrow \mathsf{Open}(\mathsf{ck},\mathsf{C},\mathsf{aux},\vec{x})$, then $\mathsf{Vrf}(\mathsf{ck},\mathsf{C},\mathsf{W}_y,\vec{x},y) = 1$ with probability 1.

**Security**. The security requirements of functional commitments are formalized as follows.

**Definition 6.2** (Perfectly Hiding)**.** A commitment scheme is perfectly hiding if for a commitment key $\mathsf{ck}$ generated by an honest setup, for all $\vec{m}_0,\vec{m}_1 \in \mathcal{D}^n$ with $\vec{m}_0 \neq \vec{m}_1$, the two distributions $\{\mathsf{ck},\mathsf{Commit}(\mathsf{ck},\vec{m}_0)\}$ and $\{\mathsf{ck},\mathsf{Commit}(\mathsf{ck},\vec{m}_1)\}$ are identical given that the random coins of $\mathsf{Commit}$ are chosen according to the uniform distribution from the respective domain.

The binding property captures the requirement that the adversary, without the knowledge of the trapdoor key $\mathsf{tk}$, cannot generate a commitment $\mathsf{C}$ and accept the witnesses for two distinct values $y, y'$.

**Definition 6.3** (Function Binding)**.** A functional commitment scheme $\mathsf{FC} = (\mathsf{Setup},\mathsf{Commit},\mathsf{Open},\mathsf{Vrf})$ for $(\mathcal{D}, n, \langle\cdot,\cdot\rangle)$ is *computationally binding* if for any PPT adversary $\mathcal{A}$ winning the following experiment with negligible advantage.

1. The challenger generates $(\mathsf{ck},\mathsf{tk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and gives $\mathsf{ck}$ to the adversary $\mathcal{A}$.
2. The adversary $\mathcal{A}$ outputs a commitment $\mathsf{C}$, a vector $\vec{x} \in \mathcal{D}^n$, two values $y, y' \in \mathcal{D}$ and two witnesses $\mathsf{W}_y, \mathsf{W}_{y'}$. We say that the adversary $\mathcal{A}$ wins the game if the following conditions hold
   - $y \neq y'$.
   - $\mathsf{Vrf}(\mathsf{ck},\mathsf{C},\mathsf{W}_y,\vec{x},y) = \mathsf{Vrf}(\mathsf{ck},\mathsf{C},\mathsf{W}_{y'},\vec{x},y') = 1$.

### 6.5 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 6.1** ([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*

- $\mathsf{F}_q$: *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 $\mathsf{RF}(x) = y$.*

## 7 Our Broadcast Encryption Scheme

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

### 7.1 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

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

and then compute

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

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

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

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

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

and

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

where $\mathsf{ct}_1$ is computed as $\left(g^{\beta} \cdot \prod_{k\in S} g_k\right)^s$ and the symmetric key $\mathsf{key}$ is computed as $\mathsf{H}(e(g_1, u_n)^s)$. The algorithm finally outputs $(\mathsf{ct}, \mathsf{key})$.

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

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

## 7.2 Security Analysis

**Theorem 7.1.** *For any PPT adversary $\mathcal{A}$, there exists an algorithm $\mathcal{B} = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$ whose running times are essentially the same as that of $\mathcal{A}$, 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.

$\underline{\mathsf{Hyb}_0}$: is the real experiment as defined in the selective security model.

$\underline{\mathsf{Hyb}_1}$: This is the same as $\mathsf{Hyb}_0$, 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 $\mathsf{mpk}$ are computed as in the honest setup, but the challenge ciphertext $\mathsf{ct}^*$ is computed as $(g^s, (g^s)^{\tilde{\beta}})$ and the symmetric key $\mathsf{key}_0$ is computed as $\mathsf{H}(e(g^s, u_{n+1}))$. 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 $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$ 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 $\mathsf{Hyb}_1$, 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 $G_{p_1p_2p_3}$.

**Lemma 7.1.** *Under the decisional subgroup assumption* 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 $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, T)$ where either $T \leftarrow G_{p_1}$ or $T \leftarrow G_{p_1p_2p_3}$, 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 $\mathsf{H}$, 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
$$\mathsf{mpk} = ((g, g^{\beta}, e(g, u_{n+1}), \mathsf{H}), g_1^{\alpha}, \cdots, g_1^{\alpha^n}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n})$$
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 $\mathsf{sk}_i$ 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 $\mathsf{sk}_i$ 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 $\mathsf{Hyb}_1$; when $T \leftarrow G_{p_1p_2p_3}$, the simulation is identical to $\mathsf{Hyb}_2$. 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 7.2.** *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 $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, T)$, where either $T = u \leftarrow G_{p_1}$ or $T = ug_2^r \leftarrow G_{p_1p_2}$. 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 $\mathsf{H}$ 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 $\mathsf{Hyb}_2$ to compute the public parameters $\mathsf{mpk}$ 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 $\mathsf{Hyb}_2$; 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 7.3.** *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 $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, T)$ 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 $\mathsf{H}$ 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 $u^{\alpha^k} \cdot T^{\sum_{j=1}^{2^i} r'_j \alpha'^k_j} R_{4.k}$. Finally it proceeds as $\mathsf{Hyb}_2$ using $\alpha, u_1, \cdots, u_{2n}$ as computed above to compute the public parameters $\mathsf{mpk}$ 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 7.4.** *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 the instance $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, Y_2Y_4 \in G_{p_2p_4}, T)$ 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 $\mathsf{H}$ 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 $\mathsf{mpk}$ and the set of secret keys $\{\mathsf{sk}_i : i \notin S^*\}$ as in $\mathsf{Hyb}_2$.

- **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 $\mathsf{RF}$ 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 6.1, 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. □

## 8 Our Tightly Secure Non-Zero Inner Product Encryption Scheme

In this section we give the description of our tightly secure NIPE scheme $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$. Recall that our NIPE construction is in the key encapsulation setting as we defined.

### 8.1 Construction

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

- $\mathsf{Setup}(1^\lambda, 1^n)$: takes as input the security parameter $\lambda$ and $n$ as the dimension of the inner product space. Compute $\mathcal{G}(1^\lambda)$ to obtain parameters of bilinear groups $(N = p_1p_2p_3p_4, G, G_T, e)$, where $p_i > 2^{\ell(\lambda)}$ for each $i \in \{1,2,3,4\}$ for a suitable polynomial $\ell : \mathbb{N} \to \mathbb{N}$. We consider inner products defined over $\mathbb{Z}_N^n$. Choose $(\alpha, \beta, g, u) \xleftarrow{\$} G_{p_1}^2 \times \mathbb{Z}_N^2$ and $R_4 \xleftarrow{\$} G_{p_4}$. Define $g_j := g^{\alpha^j}$ for each $j \in [n]$ and

$u_k := u^{\alpha^k} R_{4.k}$, where $R_{4.k} \xleftarrow{\$} G_{p_4}$, for each $k \in [2n]\backslash\{n+1\}$. It selects a pairwise-independent hash function $\mathsf{H} : G_T \to \{0,1\}^\lambda$. Finally it outputs the public parameters

$$\mathsf{mpk} := \big((G, G_T, e), g, g^\beta, \{g_j\}_{j=1}^n, \{u_k\}_{k\in[2n]\backslash\{n+1\}}, \mathsf{H}\big)$$

and the master secret key as $\mathsf{msk} = (u, \alpha, \beta, R_4)$.

- $\mathsf{Enc}(\mathsf{mpk}, \vec{x})$: To generate an encryption under the vector $\vec{x} \in \mathbb{Z}_N^n$. It samples $s \xleftarrow{\$} \mathbb{Z}_N$ and define the ciphertext $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1)$ and a symmetric key $\mathsf{key}$ as follows.

$$\mathsf{ct}_0 = g^s, \qquad \mathsf{ct}_1 = g^{s\cdot(\beta + \sum_{i=1}^n x_i \alpha^i)}, \qquad \mathsf{key} = \mathsf{H}(e(g, u_{n+1})^s)$$

  where $\mathsf{ct}_1$ is computed as $\big(g^\beta \cdot \prod_{i=1}^n g_i^{x_i}\big)^s$ and the symmetric key $\mathsf{key}$ is computed as $\mathsf{H}(e(g_1, u_n)^s)$. The algorithm finally outputs $(\mathsf{ct}, \mathsf{key})$.

- $\mathsf{KeyGen}(\mathsf{msk}, \vec{y})$: The secret key for the vector $\vec{y} = (y_1, \cdots, y_n) \in \mathbb{Z}_N^n$ is given by

$$\mathsf{sk}_{\vec{y}} = \left(\prod_{i=1}^n u^{\alpha^i \cdot y_i}\right)^\beta \cdot R_4'$$

  where $R_4' \xleftarrow{\$} G_{p_4}$ is sampled using $R_4$.

- $\mathsf{Dec}(\mathsf{ct}, \vec{x}, \vec{y}, \mathsf{sk}_{\vec{y}})$: Let $\omega = \langle \vec{x}, \vec{y}\rangle \mod N$. If $\omega \neq 0$ the algorithm computes

$$A_i = \prod_{j=1, j\neq i}^n u_{n+1+j-i}^{x_j}, \qquad \forall i \in [n]$$

  and recover the symmetric key $\mathsf{key}$ as

$$\mathsf{H}\left(\left(\frac{e(\mathsf{ct}_1, \prod_{i=1}^n u_{n-i+1}^{y_i})}{e(\mathsf{ct}_0, \mathsf{sk}_{\vec{y}} \cdot \prod_{i=1}^n A_i^{y_i})}\right)^{1/\omega}\right)$$

**Correctness**. The correctness of our construction can be verified as follows

$$\begin{aligned}
e(\mathsf{ct}_1, \prod_{i=1}^{n} u_{n-i+1}^{y_i}) &= \prod_{i=1}^{n} e(g^{s(\beta+\sum_{i=1}^{n}\alpha^i x_i)}, u^{\alpha^{n-i+1}\cdot y_i}) \\
&= \prod_{i=1}^{n} e(g^{\beta} \cdot \prod_{i=1}^{n} g_i^{x_i}, u^{\alpha^{n-i+1} y_i})^s \\
&= \prod_{i=1}^{n} e(g^{\beta}, \prod_{i=1}^{n} g^{\alpha^i x_i}, u^{\alpha^{n-i+1} y_i})^s \\
&= \prod_{i=1}^{n} e(g,u)^{\alpha^{n+1}\cdot s\cdot x_i \cdot y_i} \cdot \prod_{i=1}^{n} e\left(g, u_{n-i+1}^{\beta} \cdot \prod_{j=1, j\neq i}^{n} u^{\alpha^{n+1+j-i} x_j}\right)^{s\cdot y_i} \\
&= \prod_{i=1}^{n} e(g,u)^{\alpha^{n+1}\cdot s\cdot x_i \cdot y_i} \cdot \prod_{i=1}^{n} e(g, u^{\alpha^{n-i+1}\beta} \cdot A_i)^{s \cdot y_i} \\
&= e(g,u)^{\alpha^{n+1}\cdot s \cdot \langle \vec{x}, \vec{y} \rangle} \cdot e\left(g^s, \prod_{i=1}^{n} u^{\alpha^{n-i+1}\cdot\beta\cdot y_i}, A_i^{y_i}\right) \\
&= e(g,u)^{\alpha^{n+1}\cdot s\cdot\omega} \cdot e\left(\mathsf{ct}_0, \mathsf{sk}_{\vec{y}} \cdot \prod_{i=1}^{n} A_i^{y_i}\right)
\end{aligned}$$

From the sequence of equations we have that

$$\left(\frac{e(\mathsf{ct}_1, \prod_{i=1}^{n} u_{n-i+1}^{y_i})}{e(\mathsf{ct}_0, \mathsf{sk}_{\vec{y}} \cdot \prod_{i=1}^{n} A_i^{y_i})}\right)^{1/\omega} = e(g, u_{n+1})^s$$

## 8.2 Security Analysis

**Theorem 8.1.** *For any PPT adversary $\mathcal{A}$, there exists an algorithm $\mathcal{B} = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$ whose running times are essentially the same as that of $\mathcal{A}$, such that*

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

The security analysis of our NIPE scheme is essentially similar to the security analysis of our broadcast encryption scheme. Thus we leave its security analysis in the appendix A.

# 9 Our Functional Commitment Scheme for Linear Functions

In this section we give the description of our tightly secure functional commitment scheme for linear functions.

## 9.1 Construction

Our functional commitment scheme for linear functions $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Vrf})$ is described as follows.

- $\mathsf{Setup}(1^\lambda, 1^n)$: It generates the parameters of bilinear group $\mathbb{G} = (N, G, G_T, e)$ by executing the algorithm $\mathcal{G}(1^\lambda)$. The algorithm then samples $(g, u, \alpha) \xleftarrow{\$} G_{p_1}^2 \times \mathbb{Z}_N$ and the randomness $R_4 \xleftarrow{\$} \mathbb{G}_{p_4}$. The it computes $u_k = u^{\alpha^k} R_{4.k}$ for all $k \in [2n]$, where $R_{4.k} \xleftarrow{\$} G_{p_4}$. It outputs the commitment key as

$$\mathsf{ck} = (g, g^\alpha, \cdots, g^{\alpha^n}, u_1, \cdots, u_n, u_{n+2}, \cdots, u_{2n}, R_4)$$

and the trapdoor key as $\mathsf{tk} = u_{n+1}$.

- $\mathsf{Commit}(\mathsf{ck}, \vec{m})$: On input a message vector $\vec{m} = (m_1, \cdots, m_n) \in \mathbb{Z}_N^n$, it chooses a random value $\beta \xleftarrow{\$} \mathbb{Z}_N$ and computes the commitment $\mathsf{C} = g^{\beta + \sum_{j=1}^n m_j \alpha_j}$. The commitment $\mathsf{C}$ is computed as $g^\beta \cdot \prod_{j=1}^n g_j^{m_j}$. Finally it outputs the commitment $\mathsf{C}$ and the auxiliary information $\mathsf{aux} = (m_1, \cdots, m_n, \beta)$.

- $\mathsf{Open}(\mathsf{ck}, \mathsf{C}, \mathsf{W}_y, \vec{x}, y)$: Given $\vec{x} = (x_1, \cdots, x_n) \in \mathbb{Z}^n$, the auxiliary information $\mathsf{aux} = (m_1, \cdots, m_n, \beta)$ allows generating a witness for the function $\langle \vec{m} \cdot \vec{x} \rangle = \sum_{i=1}^n m_i \cdot x_i$ by computing

$$\mathsf{W}_i = u_{n-i+1}^\beta \cdot \prod_{j=1, j\neq i}^n u_{n+1+j-i}^{m_j}, \quad \forall i \in \{1, \cdots, n\}$$

  and outputting $\mathsf{W}_y = \prod_{i=1}^n \mathsf{W}_i^{x_i}$.

- $\mathsf{Vrf}(\mathsf{ck}, \mathsf{C}, \mathsf{W}_y, \vec{x}, y)$: Given the commitment $\mathsf{C} \in G$ and $\vec{x} = (x_1, \cdots, x_n) \in \mathbb{Z}_N^n$. It accepts the witness $\mathsf{W}_y \in G$ as evidence that $\mathsf{C}$ is a commitment to the message $\vec{m} \in \mathbb{Z}_N^n$ such that $y = \langle \vec{m}, \vec{x} \rangle$ if and only if it holds that

$$e(\mathsf{C}, \prod_{i=1}^n u_{n-i+1}^{x_i}) = e(g, u_{n+1})^y \cdot e(g, \mathsf{W}_y)$$

  If so, it outputs 1. Otherwise output 0.

**Correctness**. The correctness of our functional commitment scheme can be verified as follows

$$\begin{aligned} e(\mathsf{C}, \prod_{i=1}^n u_{n-i+1}^{x_i}) &= \prod_{i=1}^n e\left(g, u\right)^{\alpha^{n+1} m_i x_i} \cdot \prod_{i=1}^n e\left(g, u_{n-i+1}^\beta \cdot \prod_{j=1, j\neq i}^n u_{n+j-i+1}^{m_j}\right)^{x_i} \\ &= \prod_{i=1}^n e\left(g, u_{n+1}\right)^{m_i \cdot x_i} \cdot \prod_{i=1}^n e\left(g, \mathsf{W}_i\right)^{x_i} \\ &= e\left(g, u_{n+1}\right)^{\langle \vec{m}, \vec{x} \rangle} \cdot e\left(g, \mathsf{W}_y\right) \end{aligned}$$

## 9.2 Security Analysis

It is clear that the commitment is perfectly hiding. Since the commitment $\mathsf{C}$ is in the cyclic subgroup $G_{p_1}$, and for all message vector $\vec{m} = (m_1, \cdots, m_n) \in \mathbb{Z}_N^n$ has a corresponding opening $\beta \in \mathbb{Z}_N$. This even holds for the openings for the subgroup $G_{p_2 p_3 p_4}$ since only the randomness $\beta$ and $p_1$ are fixed by $\vec{m}$. Here we prove the binding property of our functional commitments.

**Theorem 9.1.** *The functional commitment scheme described in section 9.1 is function binding under the decisional subgroup assumptions* $\mathsf{DS1}, \mathsf{DS2}, \mathsf{DS3}, \mathsf{DS4}$.

The security analysis of our functional commitment scheme is essentially similar to the security analysis of our broadcast encryption scheme and our NIPE scheme. Thus we leave its security analysis in the appendix B.

**Open Problem**. Our constructions (both BE scheme and NIPE scheme) 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 and how to find a prime-order BE or NIPE with tight reduction and same parameter sizes are also interesting problems.

**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.

[CLR16] Jie Chen, Benoît Libert, and Somindu Ramanna. Non-zero inner product encryption with short ciphertexts and private keys. 2016.

[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.

[LRY16] Benoît Libert, Somindu Ramanna, and Moti Yung. Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In *43rd International Colloquium on Automata, Languages and Programming (ICALP 2016)*, 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.

# Appendices

## Appendix A Proof of Our NIPE Scheme

Here we prove the selective security with tight reductions of our NIPE scheme. First we recall the theorem 8.1. We want to prove that for any PPT adversary $\mathcal{A}$, there exists an algorithm $\mathcal{B} = \{\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4\}$ whose running times are essentially the same as that of $\mathcal{A}$, such that

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

We prove it formally as follows.

*Proof.* To prove the security of our NIPE, we need to define the following three kinds of family of functions to modify the distribution of public parameters.

1. The family of functions $\{F_{2^i} : [2n] \to \mathbb{Z}_{p_2}\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$,

$$F_{2^i}(k) = \sum_{j=1}^{2^i} r_j \cdot \alpha_j^k \mod p_2 \text{ for all } i \in [1, \cdots, \lceil \log n + 1 \rceil], \qquad F_0(k) = 0$$

   where $r_1, \cdots, r_{2^i}, \alpha_1, \cdots, \alpha_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_2}$.

2. The family of functions $\{\widehat{F_{2^i}} : [2n] \to \mathbb{Z}_{p_3}\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$,

$$\widehat{F_{2^i}}(k) = \sum_{j=1}^{2^i} \hat{r}_j \cdot \hat{\alpha}_j^k \mod p_3 \text{ for all } i \in [1, \cdots, \lceil \log n + 1 \rceil], \qquad F_0(k) = 0$$

   where $\hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_3}$.

3. The family of functions $\{\widehat{F_{2^i}}' : [2n] \to \mathbb{Z}_{p_2}\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$,

$$\widehat{F_{2^i}}'(k) = \sum_{j=1}^{2^i} r_j \cdot \alpha_j^k + \hat{r}_j \cdot \hat{\alpha}_j^k \mod p_2 \text{ for all } i \in [1, \cdots, \lceil \log n + 1 \rceil], \qquad F_0(k) = 0$$

   where $r_1, \cdots, r_{2^i}, \alpha_1, \cdots, \alpha_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_2}$.

Respectively we need the three kinds of parameters used in the hybrid arguments in order to modify the distribution of $\{u_k\}_{k\in[2n]}$

1. **Type $i$ parameters** ($0 \leq i \leq \lceil \log n + 1 \rceil$): are parameters where $\{u_k\}_{k\in[2n]}$ have a $G_{p_2}$ component determined by the function $F_{2^i}(\cdot)$, such that

$$u_k = u^{\alpha^k} \cdot g_2^{F_{2^i}(k)} \cdot R_{4.k}, \qquad \forall k \in [2n]$$

2. **Type $i.1$ parameters** ($0 \leq i \leq \lceil \log n+1 \rceil$): are parameters where $\{u_k\}_{k\in[2n]}$ have both $G_{p_2}$ component and $G_{p_3}$ component, where $G_{p_2}$ component is still determined by the function $F_{2^i}(\cdot)$ as above, and the

$G_{p_3}$ component is determined by another function $\widehat{F_{2^i}}(\cdot)$ such that

$$u_k = u^{\alpha^k} \cdot g_2^{F_{2^i}(k)} \cdot g_3^{\widehat{F_{2^i}}(k)} \cdot R_{4.k}, \qquad \forall k \in [2n]$$

3. **Type** $i.2$ **parameters** $(0 \le i \le \lceil \log n + 1 \rceil)$: are parameters where $\{u_k\}_{k \in [2n]}$ have a $G_{p_2}$ component determined by the function $\widehat{F_{2^i}}'(\cdot)$, such that

$$u_k = u^{\alpha^k} \cdot g_2^{\widehat{F_{2^i}}'(k)} \cdot R_{4.k}, \qquad \forall k \in [2n]$$

Now we are ready to define hybrid arguments. The proof proceeds through a sequence of hybrids: $\mathsf{Hyb}_0$, $\mathsf{Hyb}_1$, $\mathsf{Hyb}_2$, $\mathsf{Hyb}_{3.i}$ for all $i \in [1, \cdots, \lceil \log n + 1 \rceil]$, and $\mathsf{Hyb}_4$. We define $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{x.y.z}}(\lambda)$ as the advantage function in hybrid $\mathsf{Hyb}_{x.y.z}$.

$\underline{\mathsf{Hyb}_0}$: is the real game as the selective security definition of NIPE scheme.

$\underline{\mathsf{Hyb}_1}$: is equivalent to $\mathsf{Hyb}_0$, except that in this hybrid the challenger chooses $\widetilde{\beta} \xleftarrow{\$} \mathbb{Z}_N$ and sets $\beta = \widetilde{\beta} - \sum_{i=1}^n \alpha^i x_i^*$ where $\vec{x}^* = (x_1^*, \cdots, x_n^*)$. The challenger sets $g^\beta = g^{\widetilde{\beta}} \prod_{i=1}^n g_i^{-x_i^*}$. Thus the ciphertext $\mathsf{ct}^* = (\mathsf{ct}_0^*, \mathsf{ct}_1^*)$ and the symmetric key $\mathsf{key}_0^*$ is computed as follows

$$\mathsf{ct}_0^* = g^s, \qquad \mathsf{ct}_1^* = g^{s \cdot \widetilde{\beta}}, \qquad \mathsf{key} = \mathsf{H}(e(g^s, u_{n+1}))$$

It is easy to see that we have $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_0}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_1}(\lambda)$

$\underline{\mathsf{Hyb}_2}$: is equivalent to $\mathsf{Hyb}_1$, except that the challenger picks $\mathsf{ct}_0$ uniformly at random in the subgroup $G_{p_1 p_2 p_3}$ instead of from $G_{p_1}$. More specifically, the challenger samples $s \xleftarrow{\$} \mathbb{Z}_N$, it computes $\mathsf{ct}_0 = (g_1 g_2 g_3)^s$, $\mathsf{ct}_1 = (g_1 g_2 g_3)^{s \cdot \widetilde{\beta}}$, and $\mathsf{key} = \mathsf{H}(e((g_1 g_2 g_3)^s, u_{n+1}))$.

**Lemma A.1.** *Under the decisional subgroup assumption* $\mathsf{DS1}$*, we have*

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

*Proof.* Given $(\mathbb{G}, g_1, g_4, T)$ where either $T \leftarrow G_{p_1}$ or $T \leftarrow G_{p_1 p_2 p_3}$, we build a reduction $\mathcal{B}_1$ working as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a vector $\vec{x}^* = (x_1^*, \cdots, x_n^*)$.

- **Setup Phase**: Pick parameters $(\alpha, \tilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, $\beta = \tilde{\beta} - \sum_{i=1}^n \alpha^i x_i^*$. Select the hash function $\mathsf{H}$, all randomness $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k = 1, \cdots, 2n$ and $R_4 \xleftarrow{\$} G_{p_4}$. It outputs the public parameters

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

  and keeps the master secret key $\mathsf{msk} = (u, \alpha, \beta, R_4)$ privately.

- **Key Extraction**: Upon a key query on vector $\vec{y} \in \mathbb{Z}_N^n$, the adversary is given a secret key $\mathsf{sk}_{\vec{y}} = \prod_{i=1}^n u_{n-i+1}^{y_i} \cdot R_4'$ where $R_4' \xleftarrow{\$} G_{p_4}$ is sampled using $R_4$.

- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$, and compute the ciphertext $\mathsf{ct}^* = (\mathsf{ct}_0^* = T^{s'}, \mathsf{ct}_1^* = (T^{s'})^{\tilde{\beta}})$ 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 $\mathsf{Hyb}_1$; when $T \leftarrow G_{p_1 p_2 p_3}$, the simulation is identical to $\mathsf{Hyb}_2$. This proves the lemma. $\square$

$\underline{\mathsf{Hyb}_{3.i}(0 \leq i \leq \lceil \log n + 1 \rceil)}$: is equivalent to $\mathsf{Hyb}_2$, except that in this hybrid the adversary is given Type $i$ parameters. In order to prove $\mathsf{Hyb}_{3.i}$ is computationally indistinguishable from $\mathsf{Hyb}_{3.i+1}$, we need two additional sub-hybrids as follows

- $\underline{\mathsf{Hyb}_{3.i.1}(0 \leq i \leq \lceil \log n + 1 \rceil)}$: is equivalent to $\mathsf{Hyb}_{3.i}$, except that the adversary is given Type $i.1$ parameters.
- $\underline{\mathsf{Hyb}_{3.i.2}(0 \leq i \leq \lceil \log n + 1 \rceil)}$: is equivalent to $\mathsf{Hyb}_{3.i.1}$, except that the adversary is given Type $i.2$ parameters.

We first prove that $\mathsf{Hyb}_2$ is computationally indistinguishable from $\mathsf{Hyb}_{3.0}$ under the decisional subgroup assumption $\mathsf{DS2}$.

**Lemma A.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 the instance $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, T)$ where either $T = u \xleftarrow{\$} G_{p_2}$ or $T = ug_2^r \xleftarrow{\$} G_{p_1p_2}$. We build a reduction $\mathcal{B}_2$ simulating the game as follows:

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a vector $\vec{x}^* = (x_1^*, \cdots, x_n^*) \in \mathbb{Z}_N^n$.
- **Setup Phase**: The challenger picks $(\alpha, \widetilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, set $\beta = \widetilde{\beta} - \sum_{i=1}^n \alpha^i \cdot x_i^*$. The challenger then selects a pairwise-independent hash function $\mathsf{H}$, and sample all randomness $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k \in [2n]$ and $R_4 \xleftarrow{\$} G_{p_4}$. Define $g_j = g^{\alpha^j}$ for $j \in [n]$ and compute each $u_k$ as
$$u_k = T^{\alpha^k} R_{4.k}$$
for each $k \in [2n]$. Proceed as $\mathsf{Hyb}_2$ to compute the public parameters
$$\mathsf{mpk} = (\mathbb{G}, g, g^\beta, \{g_j\}_{j \in [n]}, \{u_k\}_{k \in [2n] \setminus \{n+1\}}, \mathsf{H})$$
the public parameters $\mathsf{mpk}$ is sent to the adversary $\mathcal{A}$ and the challenger keeps the master secret key $\mathsf{msk} = (u, R_4, \alpha, \beta)$ privately.
- **Key Extraction Phase**: Upon query on the vector $\vec{y} \in \mathbb{Z}_N^n$, the adversary is given a secret key for the vector $\vec{y}$ as
$$\mathsf{sk}_{\vec{y}} = (\prod_{i=1}^n u_{n-i+1}^{y_i}) \cdot R_4'$$
where $R_4' \xleftarrow{\$} G_{p_4}$ is sampled using $R_4$.
- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$, and compute the ciphertext $\mathsf{ct}^* = (\mathsf{ct}_0^* = (X_1X_2X_3)^{s'}, \mathsf{ct}_1^* = ((X_1X_2X_3)^{s'})^{\tilde{\beta}})$ and the symmetric key $\mathsf{key}_0^* = \mathsf{H}(e((X_1X_2X_3)^{s'}, u_{n+1}))$. The reduction $\mathcal{B}_2$ picks random $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ and return $(\mathsf{ct}^*, \mathsf{key}_b^*)$, where $b$ is a random coin.

- **Guess**: The adversary returns a bit $b'$, the reduction $\mathcal{B}_2$ returns 1 if $b = b'$ and 0 otherwise.

Observe that if $T = u$, the simulation is identical to $\mathsf{Hyb}_2$; If $T = ug_2^r$, the simulation is identical to $\mathsf{Hyb}_{3.0}$. This proves the lemma. $\square$

Then we are left to prove $\mathsf{Hyb}_{3.i}$ is computationally indistinguishable from $\mathsf{Hyb}_{3.i+1}$ for all $i \in [1, \cdots, \lceil \log n + 1 \rceil]$ through the following two lemmas.

**Lemma A.3.** *Under the decisional subgroup assumption* DS3*, we have*

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

*for all* $i \in [1, \cdots, \lceil \log n + 1 \rceil]$.

*Proof.* Given the instance $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3, T)$ where either $T = g_2 \xleftarrow{\$} G_{p_2}$ or $T = g_2 g_3 \xleftarrow{\$} G_{p_2p_3}$. We build a reduction $\mathcal{B}_3$ working as follows:

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a vector $\vec{x}^* = (x_1^*, \cdots, x_n^*) \in \mathbb{Z}_N^n$.
- **Setup Phase**: The challenger picks $(\alpha, \widetilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, set $\beta = \widetilde{\beta} - \sum_{i=1}^n \alpha^i \cdot x_i^*$. The challenger then selects a pairwise-independent hash function $\mathsf{H}$, and sample all randomness $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k \in [2n]$ and $R_4 \xleftarrow{\$} G_{p_4}$. Define $g_j = g^{\alpha^j}$ for $j \in [n]$ and compute each $u_k$ as

$$u_k = T^{\sum_{j=1}^{2^i} r'_j \alpha_j'^k} R_{4.k}$$

  for each $k \in [2n]$, where $r'_1, \cdots, r'_{2^i}, \alpha'_1, \cdots, \alpha'_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_2}$. Proceed as $\mathsf{Hyb}_2$ to compute the public parameters

$$\mathsf{mpk} = (\mathbb{G}, g, g^\beta, \{g_j\}_{j\in[n]}, \{u_k\}_{k\in[2n]\setminus\{n+1\}}, \mathsf{H})$$

  the public parameters $\mathsf{mpk}$ is sent to the adversary $\mathcal{A}$ and the challenger keeps the master secret key $\mathsf{msk} = (u, R_4, \alpha, \beta)$ privately.
- **Key Extraction Phase**: Upon query on the vector $\vec{y} \in \mathbb{Z}_N^n$, the adversary is given a secret key for the vector $\vec{y}$ as

$$\mathsf{sk}_{\vec{y}} = (\prod_{i=1}^n u_{n-i+1}^{y_i}) \cdot R'_4$$

  where $R'_4 \xleftarrow{\$} G_{p_4}$ is sampled using $R_4$.
- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$, and compute the ciphertext $\mathsf{ct}^* = (\mathsf{ct}_0^* = (X_1X_2X_3)^{s'}, \mathsf{ct}_1^* = ((X_1X_2X_3)^{s'})^{\tilde{\beta}})$ and the symmetric key $\mathsf{key}_0^* = \mathsf{H}(e((X_1X_2X_3)^{s'}, u_{n+1}))$. The reduction $\mathcal{B}_2$ picks random $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ and return $(\mathsf{ct}^*, \mathsf{key}_b^*)$, where $b$ is a random coin.
- **Guess**: The adversary returns a bit $b'$, the reduction $\mathcal{B}_3$ returns 1 if $b = b'$ and 0 otherwise.

observe that 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 $\hat{r}_j = \hat{r}'_j \mod p_3$. This proves the lemma. $\square$

**Lemma A.4.** *Under the decisional subgroup assumption* DS4, *we have*

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

*for all* $i \in [1, \cdots, \lceil \log n + 1 \rceil]$.

*Proof.* Given the instance $(\mathbb{G}, g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, Y_2Y_4 \in G_{p_2p_4}, T)$ where either $T \in G_{p_2p_4}$ or $T = G_{p_3p_4}$. We build a reduction $\mathcal{B}_4$ working as follows.

- **Commitment Phase**: The adversary $\mathcal{A}$ commits to a vector $\vec{x}^* = (x_1^*, \cdots, x_n^*) \in \mathbb{Z}_N^n$.
- **Setup Phase**: The challenger picks $(\alpha, \widetilde{\beta}, u) \xleftarrow{\$} \mathbb{Z}_N^2 \times G_{p_1}$, set $\beta = \widetilde{\beta} - \sum_{i=1}^n \alpha^i \cdot x_i^*$. The challenger then selects a pairwise-independent hash function $\mathsf{H}$, and sample all randomness $R_{4.k} \xleftarrow{\$} G_{p_4}$ for $k \in [2n]$ and $R_4 \xleftarrow{\$} G_{p_4}$. Define $g_j = g^{\alpha^j}$ for $j \in [n]$ and compute each $u_k$ as

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

  for each $k \in [2n]$,

  where $r'_1, \cdots, r'_{2^i}, \alpha'_1, \cdots, \alpha'_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \xleftarrow{\$} \mathbb{Z}_{p_2}$. Proceed as $\mathsf{Hyb}_2$ to compute the public parameters

$$\mathsf{mpk} = (\mathbb{G}, g, g^\beta, \{g_j\}_{j\in[n]}, \{u_k\}_{k\in[2n]\setminus\{n+1\}}, \mathsf{H})$$

  the public parameters $\mathsf{mpk}$ is sent to the adversary $\mathcal{A}$ and the challenger keeps the master secret key $\mathsf{msk} = (u, R_4, \alpha, \beta)$ privately.
- **Key Extraction Phase**: Upon query on the vector $\vec{y} \in \mathbb{Z}_N^n$, the adversary is given a secret key for the vector $\vec{y}$ as

$$\mathsf{sk}_{\vec{y}} = (\prod_{i=1}^n u_{n-i+1}^{y_i}) \cdot R'_4$$

  where $R'_4 \xleftarrow{\$} G_{p_4}$ is sampled using $R_4$.
- **Challenge Phase**: Sample the random value $s' \xleftarrow{\$} \mathbb{Z}_N$, and compute the ciphertext $\mathsf{ct}^* = (\mathsf{ct}_0^* = (X_1X_2X_3)^{s'}, \mathsf{ct}_1^* = ((X_1X_2X_3)^{s'})^{\widetilde{\beta}})$ and the symmetric key $\mathsf{key}_0^* = \mathsf{H}(e((X_1X_2X_3)^{s'}, u_{n+1}))$. The reduction $\mathcal{B}_2$ picks random $\mathsf{key}_1^* \xleftarrow{\$} \{0,1\}^\lambda$ and return $(\mathsf{ct}^*, \mathsf{key}_b^*)$, where $b$ is a random coin.
- **Guess**: The adversary returns a bit $b'$, the reduction $\mathcal{B}_4$ returns 1 if $b = b'$ and 0 otherwise.

Observe that if $T \in G_{p_3p_4}$, the simulation is identical to $\mathsf{Hyb}_{3.i.1}$ and if $T \in G_{p_2p_4}$, then the simulation is identical to $\mathsf{Hyb}_{3.i.2}$. □

It is easy to notice that $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i.2}}(\lambda)$ is identical to $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_{3.i+1}}(\lambda)$ 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 $\mathsf{RF}(\cdot)$ is a truly random function, which determines the $G_{p_2}$ component of each $u_k$.

We argue that any legitimate adversary's view remains statistically close to that of $\mathsf{Hyb}_{3.\lceil \log n+1 \rceil}$. To see this, we first notice that the $G_{p_2}$ component of the secret keys contain linear combinations of $\mathsf{RF}(k)$ in the exponent excluding $\mathsf{RF}(n+1)$. Recall that the adversary can only make private key queries on vectors $\vec{y}$ under the restriction that $\langle \vec{y}, \vec{x} \rangle = 0$. In order to generate a secret key for the vector $\vec{y}$, the challenger set $\beta = \widetilde{\beta} - \sum_{i=1}^{n} \alpha^i \cdot x_i^*$ to create a $G_{p_1}$ component with the exponent $(\sum_{i=1}^{n} y_i \cdot \alpha^{n-i+1}) \cdot (\widetilde{\beta} - \sum_{i=1}^{n} \alpha^i \cdot x_i^*)$. Note that the coefficient of $\alpha^{n+1}$ is the inner product of $\vec{x}$ and $\vec{y}$, which is 0 for all legal private key queries. Hence, each $\mathsf{sk}_{\vec{y}}$ can be computed without using $u_{n+1}$, ensuring that $\mathsf{RF}(n+1)$ remains completely independent of any information revealed to the adversary. As a result, the distribution of

$$\begin{aligned}
\mathsf{key}_0^* &= e\left((X_1 X_2 X_3)^s, u_{n+1}\right) \\
&= e\left((X_1 X_2 X_3)^s, u^{\alpha^{n+1}} g_2^{\mathsf{RF}(n+1)} R_{4.n+1}\right) \\
&= e\left((X_1 X_2 X_3)^s, u^{\alpha^{n+1}}\right) \cdot e\left((X_1 X_2 X_3)^s, g_2^{\mathsf{RF}(n+1)}\right)
\end{aligned}$$

is statistically close to the uniform distribution over $\{0,1\}^\lambda$ as long as the $G_{p_2}$ component of $\mathsf{ct}_0^* = (X_1 X_2 X_3)^s$ is not 1. This follows from the fact that if $e\left((X_1 X_2 X_3)^s, g_2\right) \neq 1_{G_T}$, the $G_{p_2}$ component of $e\left((X_1 X_2 X_3)^s, g_2^{\mathsf{RF}(n+1)}\right)$ has $\log p_2 = \Theta(\lambda)$ bits of min-entropy. Since $\mathsf{H}$ is a pairwise-independent hash function, the left-over hash lemma ensures that the distribution of $\mathsf{key}_0^*$ is within $2^{-\lambda}$ from the uniform distribution over $\{0,1\}^\lambda$. This implies that

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

Now since the random bit $b \in \{0,1\}$ is completely hidden, we have $\mathbf{Adv}_{\mathcal{A}}^{\mathsf{Hyb}_4}(\lambda) = 0$. This finishes the proof. □

## Appendix B Proof of Our FC Scheme

In this section we prove the function binding property of our functional commitment scheme for linear functions.

*Proof.* We first assume that $y' \neq y \mod p_i$ for each $i \in \{1,2,3,4\}$. To see why, we recall the binding game, in which in order to break the binding property, the adversary $\mathcal{A}$ must come up with a commitment $\mathsf{C} \in G$, a vector $\vec{x} = (x_1, \cdots, x_n) \in \mathbb{Z}_N^n$ and a pair $y, y' \in \mathbb{Z}_N$, $y \neq y'$ and $\mathsf{W}_y, \mathsf{W}'_y \in G$ such that

$$e(\mathsf{C}, \prod_{i=1}^{n} u_{n-i+1}^{x_i}) = e(g, u_{n+1})^y \cdot e(g, \mathsf{W}_y) \tag{4}$$

$$e(\mathsf{C}, \prod_{i=1}^{n} u_{n-i+1}^{x_i}) = e(g, u_{n+1})^{y'} \cdot e(g, \mathsf{W}_{y'}) \tag{5}$$

By dividing both sides of the above two equations, we have $e(g, u_{n+1})^{y-y'} = e(g, \mathsf{W}_{y'}/\mathsf{W}_y) \Leftrightarrow e(g, u_{n+1}) = e(g, (\mathsf{W}_{y'}/\mathsf{W}_y))^{1/(y-y')} = e(g, \widehat{\mathsf{W}})$.

If $\gcd(y'-y, N) = 1$, then $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$ is of the form $u^{\alpha^{n+1}} \cdot g_2^{r_2} \cdot g_3^{r_3} \cdot g_4^{r_4}$ for some random $r_2 \in \mathbb{Z}_{p_2}$, $r_3 \in \mathbb{Z}_{p_3}$ and $r_4 \in \mathbb{Z}_{p_4}$. In the following context, we denote by $\widetilde{\mathsf{W}} = u^{\alpha^{n+1}} \cdot g_2^{r_2} \cdot g_4^{r_4}$ that can be seen as a *semi-functional* trapdoor in that it is equivalent to a product of the normal trapdoor key $\mathsf{tk}$ with a $G_{p_2}$ component. $\widehat{\mathsf{W}}$ can be seen as a *shadow semi-functional* trapdoor in that it is equivalent to a product of the normal trapdoor key $\mathsf{tk}$ with a $G_{p_2}$ component and a $G_{p_3}$ component.

Before we introduce the hybrid arguments used in this proof, we first introduce the following kinds of attacks that will be used.

- **Type-I Attack**: these attacks are those for which $(\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$ lives in the subgroup $G_{p_1 p_4}$.
- **Type-II.i Attack**: these attacks are those such that $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$ has a $G_{p_2}$ component, in which $r_2 \neq 0$, and is thus a semi-functional trapdoor. More concretely, $r_2$ is determined by the function family $\{F_{2^i}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$

$$F_{2^i}(k) = \sum_{j=1}^{2^i} r_j \cdot \alpha_j^k \mod p_2, \quad \forall i \in [1, \lceil \log n + 1 \rceil] \qquad F_0(k) = 0$$

  where $r_1, \cdots, r_{2^i}, \alpha_1, \cdots \alpha_{2^i} \overset{\$}{\leftarrow} \mathbb{Z}_{p_2}$ are chosen at random for each $i$ by the challenger that generates $\mathsf{ck}$ for the adversary.
- **Type-II.i.1 Attack**: these attacks are those such that $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$ has a $G_{p_2}$ component and an additional $G_{p_3}$ component, in which $r_2 \neq 0$ and $r_3 \neq 0$, and is thus a shadow semi-functional trapdoor. Thus $\widehat{\mathsf{W}} = u^{\alpha^{n+1}} \cdot g_2^{F_{2^i}(n+1)} \cdot g_3^{\widehat{F_{2^i}}(n+1)} \cdot R_4''$ for some $R_4'' \in G_{p_4}$. More concretely, $r_2$ is determined by the function family $\{F_{2^i}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ defined above and $r_3$ is determined by the function family $\{\widehat{F_{2^i}}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$

$$\widehat{F_{2^i}}(k) = \sum_{j=1}^{2^i} \hat{r}_j \cdot \hat{\alpha}_j^k \mod p_2, \quad \forall i \in [1, \lceil \log n + 1 \rceil] \qquad \widehat{F_0}(k) = 0$$

  where $\hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \overset{\$}{\leftarrow} \mathbb{Z}_{p_3}$ are chosen at random for each $i$ by the challenger that generates $\mathsf{ck}$ for the adversary.
- **Type-II.i.2 Attack**: these attacks are those such that $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$ has a $G_{p_2}$ component, in which $r_2 \neq 0$, and is thus a semi-functional trapdoor. More concretely, $r_2$ is determined by a new function family $\{\widehat{F_{2^i}}'(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$ such that for all $k \in [2n]$

$$\widehat{F_{2^i}}'(k) = \sum_{j=1}^{2^i} r_j \cdot \alpha_j^k + \hat{r}_j \cdot \hat{\alpha}_j^k, \quad \forall i \in [1, \lceil \log n + 1 \rceil] \qquad \widehat{F_0}'(k) = 0$$

  where $r_1, \cdots, r_{2^i}, \alpha_1, \cdots \alpha_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i} \overset{\$}{\leftarrow} \mathbb{Z}_{p_2}$ are chosen at random for each $i$ by the challenger that generates $\mathsf{ck}$ for the adversary.

Furthermore, we define the following types of parameters to be used in this proof.

- **Type-i Parameters**: are parameters where elements $u_1, \cdots, u_{2n}$ now have a $G_{p_2}$ component $r_2$ determined by $\{F_{2^i}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$. Namely $u_k = u^{\alpha^k} g_2^{r_2} R_{4.k}$ for $k \in [2n]$. These elements induce a modified joint distribution of $\mathsf{ck}$, which contains the group elements $\{u_k\}_{k \in [2n]} \backslash \{n+1\}$ and $\mathsf{tk} = u_{n+1}$.
- **Type-i.1 Parameters**: are parameters where elements $u_1, \cdots, u_{2n}$ now have an additional $G_{p_3}$ component $r_3$ determined by the function family $\{\widehat{F_{2^i}}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$. Namely $u_k = u^{\alpha^k} \cdot g_2^{r_2} \cdot g_3^{r_3} \cdot R_{4.k}$ for $k \in [2n]$. These elements induce a modified joint distribution of $\mathsf{ck}$, which contains the group elements $\{u_k\}_{k \in [2n] \backslash \{n+1\}}$ and $\mathsf{tk} = u_{n+1}$.
- **Type-i.2 Parameters**: are parameters where elements $\{u_k\}_{k \in [2n]}$ now have $G_{p_2}$ component $r_2'$ determined by the function family $\{\widehat{F_{2^i}}'(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$. Namely $u_k = u^{\alpha^k} \cdot g_2^{r_2'} \cdot R_{4.k}$ for $k \in [2n]$. These

elements induce a modified joint distribution of $\mathsf{ck}$, which contains the group elements $\{u_k\}_{k\in[2n]\setminus\{n+1\}}$ and $\mathsf{tk} = u_{n+1}$.

The sequence of games begins with the real attack game, where the adversary is given a normal $\mathsf{ck}$. Then we gradually modify the distribution of $\mathsf{ck}$ and prove that, unless either assumption $\mathsf{DS1}, \mathsf{DS2}, \mathsf{DS3}$ or $\mathsf{DS4}$ is false, the adversary $\mathcal{A}$ will produce a Type-II.i attack when fed with Type-i parameters. In the last game, $\mathsf{ck}$ consists of Type-II.$\lceil \log n + 1\rceil$.2 parameters and we argue that the adversary $\mathcal{A}$'s advantage in producing Type-II.$\lceil \log n + 1\rceil$.2 attack is statistically negligible. Following the work of [LRY16], we denote by $\mathsf{Win}_\square$ as the event that the adversary $\mathcal{A}$ wins in game $\square$ and $\mathsf{E}_\square$ as the even that the adversary $\mathcal{A}$ mounts a Type-II.i attack (including Type-II.i.1 attack and Type-II.i.2 attack) when $\mathsf{ck}$ is generated using Type-i parameters (including Type-i.1 parameters and Type-i.2 parameters), for each $i \in [0, \lceil \log n + 1\rceil]$. For example, we denote by $\mathsf{Win}_{2.i.1}$ as the event that the adversary wins in game or hybrid $\mathsf{Hyb}_{2.i.1}$ and $\mathsf{E}_{II.i.1}$ as the event that the adversary mounts a Type-II.i.1 attack when $\mathsf{ck}$ is generated using Type-i.1 parameters

$\underline{\mathsf{Hyb}_1}$: In this hybrid the adversary $\mathcal{A}$ receives a commitment key $\mathsf{ck}$ which is as in the real scheme. The following lemma B.1 shows that if $\mathsf{DS1}$ holds, any PPT adversary $\mathcal{A}$ cannot produce anything but a Type-I attack.

**Lemma B.1.** *Under the decisional subgroup assumption* $\mathsf{DS1}$, *we have*

$$\Pr[\mathsf{Win}_1 \wedge \neg \mathsf{E}_I] \leq \boldsymbol{Adv}^{\mathsf{DS1}}_{\mathcal{B}_1}(\lambda)$$

*Proof.* Let $\mathcal{A}$ be an adversary that mounts a Type-II attack (for any $k$) when fed with a commitment key $\mathsf{ck}$ in the normal form. We will build a reduction $\mathcal{B}_1$ that takes as input the tuple $(g \in G_{p_1}, g_4 \in G_{p_4}, T)$ and finds an element $\gamma$ of $G_{p_2p_4}$ with a non-trivial $G_{p_2}$ component. In turn such an $\gamma \in G_{p_2p_4}$ allows deciding whether $T \in G_{p_1}$ or $T \in G_{p_1p_2}$ since $e(\gamma, T) = 1_{G_T}$ when $T \in G_{p_1}$. The reduction $\mathcal{B}_1$ works as follows.

The reduction $\mathcal{B}_1$ honestly generates the commitment key $\mathsf{ck}$ using its input elements $g \in G_{p_1}$ and $g_4 \in G_{p_4}$. The commitment key $\mathsf{ck}$ is given to the adversary $\mathcal{A}$ and it will submits a commitment $\mathsf{C} \in G$, a pair of linear function results $y, y' \in \mathbb{Z}_N$ and two witnesses $\mathsf{W}_y, \mathsf{W}_{y'} \in G$ such that they satisfy the relation (1) and (2) and the reduction computes $\widetilde{\mathsf{W}} = (\mathsf{W}_y/\mathsf{W}_{y'})^{1/(y-y')}$ has a non-trivial $G_{p_2}$ element. Moreover the reduction $\mathcal{B}_1$ could divide $u^{\alpha^{n+1}}$ to cancel out the $G_{p_1}$ element inside of $\widetilde{\mathsf{W}}$ by computing $\gamma = \widetilde{\mathsf{W}}/u^{\alpha^{n+1}}$, which is an element of $G_{p_2p_4}$ with a non-trivial $G_{p_2}$ component. Finally the reduction $\mathcal{B}_1$ outputs 1 if $e(\gamma, T) = 1_{G_T}$ and outputs 0 otherwise. ☐

Now we are left to bound the term $\Pr[\mathsf{Win}_1 \wedge \mathsf{E}_I]$ since $\Pr[\mathsf{Win}_1] = \Pr[\mathsf{Win}_1 \wedge \mathsf{E}_I] + \Pr[\mathsf{Win}_1 \wedge \neg \mathsf{E}_I]$. Now we define the hybrid $\mathsf{Hyb}_{2.i}$.

$\underline{\mathsf{Hyb}_{2.i}}$: In this game, for each $0 \leq i \leq \lceil \log n + 1\rceil$, the commitment key $\mathsf{ck}$ and the trapdoor key $\mathsf{tk} = u_{n+1}$ now have a modified distribution obtained by having the challenger generate Type-i parameters before giving $\mathsf{ck}$ to the adversary $\mathcal{A}$. Thus $\{u_k\}_{k\in[2n]}$ now have a $G_{p_2}$ component determined by the function family $\{F_{2^i}(\cdot)\}_{i=0}^{\lceil \log n+1\rceil}$. $u_k = u^{\alpha^k} g_2^{F_{2^i}(k)} R_{4.k}$ for all $k \in [2n]$.

The following lemma B.2 shows that, under the decisional subgroup assumption $\mathsf{DS2}$, the probability that $\mathcal{A}$'s attack reveals a semi-functional trapdoor key $\mathsf{tk}$ of the same type as $\mathsf{ck}$ is about the same in $\mathsf{Hyb}_1$ and $\mathsf{Hyb}_{2.0}$.

**Lemma B.2.** *Under the decisional subgroup assumption* $\mathsf{DS2}$, *we have*

$$|\Pr[\mathsf{Win}_1 \wedge \mathsf{E}_I] - \Pr[\mathsf{Win}_{2.0} \wedge \mathsf{E}_{II.0}]| \leq \boldsymbol{Adv}^{\mathsf{DS2}}_{\mathcal{B}_2}(\lambda)$$

*Proof.* To prove this, we first assume that there exists an adversary $\mathcal{A}$ such that $\epsilon = |\Pr[\mathsf{Win}_I \wedge \mathsf{E}_I] - \Pr[\mathsf{Win}_{II.0} \wedge \mathsf{E}_{II.0}]|$ is non-negligible. We build a reduction $\mathcal{B}_2$ with advantage no less than $\epsilon$ to break the decisional subgroup assumption $\mathsf{DS2}$ successfully.

Given the tuple $(g \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, T)$, the reduction $\mathcal{B}_2$ uses the adversary $\mathcal{A}$ to decide $T \in G_{p_1}$ or $T \in G_{p_1p_2}$. The reduction $\mathcal{B}_2$ generates the commitment key $\mathsf{ck}$ and the trapdoor key $\mathsf{tk}$ as follows. It picks the random parameter $\alpha \xleftarrow{\$} \mathbb{Z}_N$ and define $g_j = g^{\alpha^j}$ for each $j \in [n]$. It chooses $\alpha_1, r_1 \xleftarrow{\$} \mathbb{Z}_N$ and computes $u_k = T^{\alpha^k} R_{4.k}$ for each $k \in [2n]$, where $R_{4.k} \xleftarrow{\$} G_{p_4}$, and it chooses a random element $R_4 \xleftarrow{\$} G_{p_4}$. Then the commitment key $\mathsf{ck} = (g, \{g_j\}_{j=1}^n, \{u_k\}_{k\in[2n]\setminus\{n+1\}}, R_4)$ is given to the adversary $\mathcal{A}$ while the reduction $\mathcal{B}_2$ keeps the trapdoor $\mathsf{tk} = u_{n+1} \cdot R_{4.n+1}$ privately. Then $\mathcal{A}$ is expected to output a commitment $\mathsf{C} \in G$, $y, y' \in \mathbb{Z}_N$ and two witnesses $\mathsf{W}_y, \mathsf{W}_{y'} \in G$ such that $y \neq y'$ that satisfy the relation (1) and (2). Finally the reduction $\mathcal{B}_2$ computes $\widehat{\mathsf{W}} = (W_{y'}/W_y)^{1/(y-y')}$, which must be of the form $u^{\alpha^{n+1}} g_2^{r_2} g_4^{r_4}$, and checks $e(X_1X_2X_3, \widehat{\mathsf{W}}/u^{\alpha^{n+1}}) = 1_{G_T}$. If it holds, this means that $\widehat{\mathsf{W}} \in G_{p_1p_4}$ and $T \in G_{p_1}$, it outputs 1. Otherwise $\widehat{\mathsf{W}} \in G_{p_1p_2p_4}$ and it outputs 0. □

Now we are left to prove that $|\Pr[\mathsf{Win}_{2.i} \wedge \mathsf{E}_{II.i}] - \Pr[\mathsf{Win}_{2.i-1} \wedge \mathsf{E}_{II.i-1}]|$ for all $i \in [1, \lceil \log n + 1 \rceil]$ is negligible. Before we prove this, we introduce the following two sub-hybrid arguments for each $i \in [1, \lceil \log n + 1 \rceil]$.

$\underline{\mathsf{Hyb}_{2.i.1}}$: In this game the commitment key $\mathsf{ck}$ and the trapdoor key $\mathsf{tk} = u_{n+1}$ now have a modified distribution obtained by having the challenger generate Type-i.1 parameters before given $\mathsf{ck}$ to the adversary $\mathcal{A}$. Thus the elements $\{u_k\}_{k\in[2n]}$ now have an additional $G_{p_3}$ component determined by the function family $\{\widehat{F_{2^i}}(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$. That is $u_k = u^{\alpha^k} g_2^{F_{2^i}(k)} g_3^{\widehat{F_{2^i}}(k)} R_{4.k}$ for all $k \in [2n]$.

$\underline{\mathsf{Hyb}_{2.i.2}}$: In this game the commitment key $\mathsf{ck}$ and the trapdoor key $\mathsf{tk} = u_{n+1}$ now have a modified distribution obtained by having the challenger generate Type-i.2 parameters before giving $\mathsf{ck}$ to the adversary $\mathcal{A}$. The elements $\{u_k\}_{k\in[2n]}$ now have a $G_{p_2}$ component (but no $G_{p_3}$ component) determined by the family of functions $\{\widehat{F_{2^i}}'(\cdot)\}_{i=0}^{\lceil \log n+1 \rceil}$. That is $u_k = u^{\alpha^k} g_2^{\widehat{F_{2^i}}'(k)} R_{4.k}$ for each $k \in [2n]$.

In the following context, the lemma B.3 shows that for each $i \in [1, \lceil \log n + 1 \rceil]$, under the decisional subgroup assumption $\mathsf{DS3}$, the probability that $\mathcal{A}$'s attack reveals a shadow semi-functional trapdoor key $\mathsf{tk}$ of the same type as $\mathsf{ck}$ is about the same in $\mathsf{Hyb}_{2.i}$ and $\mathsf{Hyb}_{2.i.1}$. The lemma B.4 shows that for each $i \in [1, \lceil \log n + 1 \rceil]$, under the decisional subgroup assumption $\mathsf{DS4}$, the probability that $\mathcal{A}$'s attack reveals a shadow semi-functional trapdoor key $\mathsf{tk}$ of the same type as $\mathsf{ck}$ is about the same in $\mathsf{Hyb}_{2.i.1}$ and $\mathsf{Hyb}_{2.i.2}$.

**Lemma B.3.** *Under the decisional subgroup assumption* $\mathsf{DS3}$, *we have*

$$|\Pr[\mathsf{Win}_{2.i} \wedge \mathsf{E}_{II.i}] - \Pr[\mathsf{Win}_{2.i.1} \wedge \mathsf{E}_{II.i.1}]| \leq \boldsymbol{Adv}_{\mathcal{B}_3}^{\mathsf{DS3}}(\lambda)$$

*Proof.* Assume that there exists $i \in [1, \cdots, \lceil \log n + 1 \rceil]$ and PPT adversaries $\mathcal{A}$ such that

$$\epsilon = |\Pr[\mathsf{Win}_{2.i} \wedge \mathsf{E}_{II.i}] - \Pr[\mathsf{Win}_{2.i.1} \wedge \mathsf{E}_{II.i.1}]|$$

is non-negligible. We build a distinguisher $\mathcal{B}_3$ with advantage at least $\epsilon$ against the decisional subgroup assumption $\mathsf{DS3}$.

Given $(g_1 \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, T)$, the reduction $\mathcal{B}_3$ uses $\mathcal{A}$ to decide $T \in G_{p_2}$ or $T \in G_{p_2p_3}$. The reduction $\mathcal{B}_3$ generates the commitment key $\mathsf{ck}$ and the trapdoor $\mathsf{tk}$ as follows: $\alpha \xleftarrow{\$} \mathbb{Z}_N$, $g_j = g^{\alpha^j}$ for $j \in [1, n]$. Choose $\alpha'_1, \cdots, \alpha'_{2^i}, r'_1, \cdots, r'_{2^i} \xleftarrow{\$} \mathbb{Z}_N$ and computes $u_k = u^{\alpha^k} T^{\sum_{j=1}^{2^i} r'_j \alpha'^k_j} R_{4.k}$ for each $k \in [1, \lceil \log n+1 \rceil]$, $R_{4.k} \xleftarrow{\$} G_{p_4}$, $R_4 \xleftarrow{\$} G_{p_4}$. Then the commitment key $\mathsf{ck} = (g, \{g_j\}_{j=1}^n, \{u_k\}_{k\in[2n]\setminus\{n+1\}}, R_4)$ is given to the adversary $\mathcal{A}$ while $\mathcal{B}_3$ keeps the trapdoor key $\mathsf{tk} = u_{n+1} \cdot R_{4.n+1}$ privately. The adversary $\mathcal{A}$ is expected to output the commitment $\mathsf{C} \in G$, $y, y' \in \mathbb{Z}_N$ and $\mathsf{W}_y, \mathsf{W}_{y'} \in G$ such that $y \neq y'$ and which satisfy the relation. At this point, $\mathcal{B}_3$ computes $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$, which must be of the form $\widehat{\mathsf{W}} = u^{\alpha^{n+1}} g_2^{r_2} g_3^{r_3} g_4^{r_4}$. The reduction check $e(X_1X_2X_3, \widehat{\mathsf{W}}/u_{n+1}) = 1_{G_T}$. If so, it outputs 1, which means $T \in G_{p_2}$. Otherwise it outputs 0, which means $T \in G_{p_2p_3}$. □

**Lemma B.4.** *Under the decisional subgroup assumption* $\mathsf{DS4}$*, we have*

$$|\Pr[\mathsf{Win}_{2.i.1} \wedge \mathsf{E}_{II.i.1}] - \Pr[\mathsf{Win}_{2.i.2} \wedge \mathsf{E}_{II.i.2}]| \leq \boldsymbol{Adv}^{\mathsf{DS4}}_{\mathcal{B}_4}(\lambda)$$

*Proof.* Assume that there exists $i \in [1, \lceil \log n + 1 \rceil]$ and a PPT adversary $\mathcal{A}$ such that

$$\epsilon = |\Pr[\mathsf{Win}_{II.i.1} \wedge \mathsf{E}_{II.i.1}] - \Pr[\mathsf{Win}_{II.i.2} \wedge \mathsf{E}_{II.i.2}]|$$

is non-negligible, we build a reduction $\mathcal{B}_4$ with advantage at least $\epsilon$ against the decisional subgroup assumption $\mathsf{DS4}$.

Given the tuple $(g_1 \in G_{p_1}, g_4 \in G_{p_4}, X_1X_2X_3 \in G_{p_1p_2p_3}, Y_2Y_4 \in G_{p_2p_4}, T)$, the reduction $\mathcal{B}_4$ uses the adversary $\mathcal{A}$ to decide if $T \in G_{p_2p_4}$ or $T \in G_{p_3p_4}$. The reduction $\mathcal{B}_4$ generates the commitment key $\mathsf{ck}$ and the trapdoor key $\mathsf{tk}$ as follows: It samples $\alpha \xleftarrow{\$} \mathbb{Z}_N$, define $g_j = g^{\alpha^j}$ for each $j \in [n]$. It samples $\alpha'_1, \cdots, \alpha'_{2^i}, r'_1, \cdots, r'_{2^i}, \hat{\alpha}_1, \cdots, \hat{\alpha}_{2^i}, \hat{r}_1, \cdots, \hat{r}_{2^i} \xleftarrow{\$} \mathbb{Z}_N$. Compute $u_k = u^{\alpha^k}(Y_2Y_4)^{\sum_{j=1}^{2^i} r'_j \alpha'^k_j} T^{\sum_{j=1}^{2^i} \hat{r}'_j \hat{\alpha}'^k_j} R_{4.k}$ for each $k \in [2n]$. $R_{4.k} \xleftarrow{\$} G_{p_4}$. The commitment key $\mathsf{ck} = (g, \{g_j\}_{j=1}^n, \{u_k\}_{k \in [2n] \setminus \{n+1\}}, R_4)$ is given to the adversary $\mathcal{A}$ while the reduction $\mathcal{B}_4$ keeps the trapdoor key $\mathsf{tk} = u_{n+1}$ privately. Then the adversary $\mathcal{A}$ is expected to output a commitment $\mathsf{C} \in G$, $y, y' \in \mathbb{Z}_N$, $\mathsf{W}_y, \mathsf{W}_{y'} \in G$ such that $y \neq y'$ and which satisfy relations. At this point $\widehat{\mathsf{W}} = (\mathsf{W}_{y'}/\mathsf{W}_y)^{1/(y-y')}$, which must be of the form $\widehat{\mathsf{W}} = u^{\alpha^{n+1}} g_2^{r_2} g_3^{r_3} g_4^{r_4}$. Finally the reduction check $e(X_1X_2X_3, \widehat{\mathsf{W}}/u_{n+1}) = 1_{G_T}$. If so, it outputs 1, which means that $T \in G_{p_2p_4}$. Otherwise it outputs 0, which means that $T \in G_{p_3p_4}$. □

It is easy to see that $\mathsf{Hyb}_{2.i.2}$ is equivalent to $\mathsf{Hyb}_{2.i+1}$ by setting $\alpha_{2^i+j} = \hat{\alpha}_j$ and $r_{2^i+j} = \hat{r}_j$ for all $j \in [2^i]$ because all $r_j$ and all $r'_j$ are i.i.d variables in $\mathsf{Hyb}_{2.i.2}$.

We conclude the proof by observe that $\Pr[\mathsf{Win}_{2.\lceil \log n+1 \rceil} \wedge \mathsf{E}_{II.\lceil \log n+1 \rceil}] \leq 1/p_2$ which is negligible. To see this, it suffices to observe that the function $F_{2^{\lceil \log n+1 \rceil}}(\cdot)$ is a random function in the adversary's view, as shown in the core lemma 6.1. Hence the function evaluation $F_{2^{\lceil \log n+1 \rceil}}(n+1)$ is statistically indistinguishable from the uniform distribution over $\mathbb{Z}_{p_2}$. This will lead to a $2^{-\Omega(\lambda)}$ security loss due to the left-over hash lemma. This finishes the proof. □