Almost-tight Identity Based Encryption against Selective Opening Attack

Junqing Gong* Xiaolei Dong† Zhenfu Cao‡ Jie Chen§

October 10, 2015

Abstract

The paper presented an identity based encryption (IBE) under selective opening attack (SOA) whose security is almost-tightly related to a set of computational assumptions. Our result is a combination of Bellare, Waters, and Yilek's method [TCC, 2011] for constructing (not tightly) SOA secure IBE and Hofheinz, Koch, and Striecks' technique [PKC, 2015] on building almost-tightly secure IBE in the multi-ciphertext setting. In particular, we first tuned Bellare *et al.*'s generic construction for SOA secure IBE to show that a one-bit IBE achieving ciphertext indistinguishability under chosen plaintext attack in the *multi-ciphertext* setting (with one-sided publicly openability) *tightly* implies a multi-bit IBE secure under selective opening attack. Next, we almost-tightly reduced such a one-bit IBE to static assumptions in the composite-order bilinear groups employing the technique of Hofheinz *et al.* This yielded the *first* SOA secure IBE with almost-tight reduction.

Contents

*Department of Computer Science and Engineering, Shanghai Jiao Tong University. Email: gongjunqing@126.com
†Shanghai Key Lab for Trustworthy Computing, East China Normal University. Email: dongxiaolei@sei.ecnu.edu.cn
‡Shanghai Key Lab for Trustworthy Computing, East China Normal University. Email: zfcao@sei.ecnu.edu.cn
§Shanghai Key Laboratory of Multidimensional Information Processing and Shanghai Key Lab of Trustworthy Computing, East China Normal University. Email: S080001@e.ntu.edu.sg

1 Introduction

1.1 Background and Problem

Boneh and Franklin [BF01] formalized the notion of ciphertext indistinguishability under chosen ciphertext attack (IND-CCA) for identity based encryptions (IBE) and proposed the first practical solution in the random oracle model. Since then IND-CCA security and its weakened version, ciphertext indistinguishability under chosen plaintext attack (IND-CPA), have been accepted as standard security definitions for IBE. However stronger security guarantee is required in some application scenarios. In 2011, Bellare, Waters, and Yilek proposed the selective opening security (SOA) for IBE, which may act as the basis for discussing adaptively secure multi-party computation protocol [BWY11].

Different from IND-CPA, the SOA security formalized in [BWY11] considers a communication system with multiple senders and multiple receivers. Besides eavesdropping all ciphertexts from communication channel, the adversary can also corrupt a subset of senders, extracting their plaintexts as well as random coins they used when generating corresponding ciphertexts. The SOA security ensures that ciphertexts sent by *uncorrupted* senders should not reveal any useful information on their corresponding plaintexts.

To construct an IBE achieving SOA security described above, Bellare *et al.* [BWY11] introduced a new primitive, IND-CPA one-bit IBE with one-sided publicly openability (1SPO), which is analogous to a weaker form of deniable public key encryption (PKE) [CDNO97]. Informally, the 1SPO property here requires that one can *publicly* recover the random coins used to encrypt message 1. They showed that a one-bit IBE with such a security guarantee can be generically transformed to a multi-bit IBE with SOA security in a quite straight way and provided two concrete constructions for such type of IBE based on Boyen and Waters's anonymous IBE [BW06] and De Caro, Iovino and Persiano's anonymous IBE [DCIP10], which is further based on Lewko and Waters' IBE [LW10] employing the recently developed dual system technique [Wat09]. This results in two SOA secure IBE schemes based on decisional linear assumption and general subgroup decision assumption, respectively.

However both resulting constructions are not tight, the security loss is $\mathcal{O}(k\ell)$ where k is the number of senders and ℓ is the length of each message in binary form. Namely the advantage of breaking the SOA security of these scheme is bounded by the advantage of solving some computational assumption times a factor $\mathcal{O}(k\ell)$. This means that, in order to reach certain bit security, we have to use larger security parameters to compensate the security loss, which often leads to large group size and inefficient group operation. Therefore a tightly secure construction is desirable from both theoretical and practical points of view.

Our paper is devoted to develop an IBE scheme reaching SOA security in a tighter fashion. In particular, we give a SOA secure IBE almost-tightly reduced to several static assumptions using composite order bilinear groups. The "almost-tight" means the security loss is proportional to the security parameter λ and independent of k and ℓ. In general, we consider λ as a number far smaller than $k\ell$.

1.2 Our Technique

Our work is motivated by the work of Hofheinz, Koch, and Striecks on almost-tight IBE in the multi-instance, multi-ciphertext setting [HKS15]. Roughly speaking, the so-called "ciphertext indistinguishability in the multi-instance, multi-ciphertext setting" ensures the confidentiality of multiple ciphertexts simultaneously. In the paper, we only consider a special case, i.e., the "single-instance, multi-ciphertext" setting (IND-mCPA). On the other hand, in the terms of Hofheinz *et al.*, the one-bit IBE used in Bellare *et al.*'s generic construction of SOA secure IBE is IND-CPA in the single instance, single ciphertext setting. Therefore a straight observation shows that, if we replace the one-bit IBE here with an IND-mCPA one-bit IBE also with 1SPO, this generic construction would become constantly tight. This result is quite obvious and is easy to demonstrate, we briefly present it in Section 3 to make the paper self-contained.

Having a tight generic transformation from IND-mCPA one-bit IBE to SOA secure multi-bit IBE, the remaining work is to build such a one-bit IBE whose security is almost-tightly related to some computational assumptions. Our solution is a combination of Bellare, Waters, and Yilek's second construction [BWY11] and Hofheinz, Koch, and Striecks' technique [HKS15]. Before explaining our solution, we first review these two basic work.

Dual System Technique. Both work follows the dual system technique invented by Waters [Wat09]. For an IBE scheme employing dual system technique, we define two forms for secret keys and ciphertexts, *normal* and *semi-functional*, which should be indistinguishable from each other. The normal keys and ciphertexts are used in the real system. We will say they are in the *normal space*. The semi-functional keys and ciphertexts will only be used in security proof and are always defined as normal keys and ciphertexts mixed with some additional components. We will call these components *semi-functional components* and say they are in the *semi-functional space*. Relying on certain algebraic feature, we can ensure the independence of normal space and semi-functional space in some sense, which allows us to make some changes (say, increasing entropy and breaking some algebraic structure) in the semi-functional space but avoid negative impact on the normal space, i.e., the real system.

Bellare, Waters, and Yilek's Method. Bellare *et al.*'s idea [BWY11] is originated from the work on deniable encryption [CDNO97]. From a high level, they built a one-bit IBE with both IND-CPA and *one-sided invertible sampleability* (1SIS). An encryption of message 0 has some specific algebraic structure which is detectable for secret keys (of course, related to the same identity) but is pseudo-random from view of outsiders (adversaries); an encryption of message 1 is truly random and invertible samplable using just master public key.

Assume a composite order bilinear groups $(\mathbb{G}, \mathbb{G}_T, e, N = p_1p_2p_3p_4)$, we let $\mathbb{G}_{N'}$ be the subgroup of order $N'|N$. We note that one can decompose $\mathbb{G}$ as $\mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3} \times \mathbb{G}_{p_4}$ and pairing operation on two elements from two subgroups of co-prime orders equals $1_{\mathbb{G}_T}$. Their second construction works as follows. An encryption of 0 for identity $\textsc{id} \in \mathbb{Z}_N$ is in the form of

$$\left(U_{14}^{\textsc{id}} X_{14}\right)^s g_4^{t_4}, \qquad W_{14}^s g_4^{t_4'}$$

where $U_{14}, X_{14}, W_{14} \in \mathbb{G}_{p_1p_4}, g_4 \in \mathbb{G}_{p_4}$ are parts of the master public key and $s, t_4, t_4' \in \mathbb{Z}_N$ are random coins for encryption. Since both encryption and master public key are independent in subgroup $\mathbb{G}_{p_4}$, the algebraic structure is hidden for the outsider, which can be proved following [DCIP10, LW10]. However the structure can be detected using the secret key in the form of

$$g_1^r g_3^{r_3}, \qquad \left(U_1^{\textsc{id}} X_1\right)^r g_3^{r_3'}$$

because they let $U_1, X_1, g_1 \in \mathbb{G}_{p_1}$ share the same $\mathbb{G}_{p_1}$-component with U_{14}, X_{14}, W_{14}, respectively, and subgroup $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_3}$ and $\mathbb{G}_{p_4}$ are pairwise orthogonal under pairing operations. Here the subgroup $\mathbb{G}_{p_1}$ acts as the normal space while the subgroup $\mathbb{G}_{p_2}$ is the semi-functional space. The remaining two subgroups $\mathbb{G}_{p_3}$ and $\mathbb{G}_{p_4}$ are used to additionally randomize keys and ciphertexts, respectively.

Hofheinz, Koch, and Striecks's Construction and Proof Technique. Their construction inherited the main algebraic structure of Chen and Wee's almost-tightly secure IBE in the single-ciphertext setting [CW13] which employed a variant form of Waters Hash [Wat05]. It also works on a bilinear group of composite order $N = p_1p_2p_3p_4$ as above but supports the identity space defined as the set of all n-bit binary string. An encryption of message $\textsc{m}$ for identity $\textsc{id} \in \{0,1\}^n$ is in the form of

$$g_1^s, \qquad g_1^{\sum \mathbf{w}[2i - \textsc{id}[i]] \cdot s}, \qquad e(g_1, h)^{\alpha s} \cdot \textsc{m}$$

where $g_1 \in \mathbb{G}_{p_1}, g_1^{\mathbf{w}} \in \mathbb{G}_{p_1}^{2n}$ and $e(g_1, h)^\alpha \in \mathbb{G}_T$ are parts of the master public key, and s is the random coin for encryption, while secret keys are in the form of

$$h^\alpha \cdot h_{123}^{\sum \mathbf{w}[2i - \textsc{id}[i]] \cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

where $\alpha \in \mathbb{Z}_N$, $h \in \mathbb{G}$, $h_{123} \in \mathbb{G}_{p_1p_2p_3}$, $h_{123}^{\mathbf{w}} \in \mathbb{G}_{p_1p_2p_3}$ are given in the master secret keys, and $r \in \mathbb{Z}_N, R_4, R_4' \in \mathbb{G}_{p_4}$ are random coins. Here subgroup $\mathbb{G}_{p_1}$ again acts as the normal space, the semi-functional space consists of subgroup $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$. The last subgroup $\mathbb{G}_{p_4}$ is now used to randomize keys.

We now briefly review the proof procedure. The proof begins with introducing $\mathbb{G}_{p_2}$-component into ciphertexts and conceptually inserting an random function R_0 mapping identity space to subgroup $\mathbb{G}_{p_2p_3}$ whose output is truly random but independent of the input as follows.

$$\textsc{ct}: \quad (g_1 \, g_2)^s, \qquad (g_1 \, g_2)^{\sum \mathbf{w}[2i - \textsc{id}[i]] \cdot s}, \qquad e((g_1 \, g_2)^s, h^\alpha \cdot \boxed{\mathsf{R}_0(\textsc{id})}) \cdot \textsc{m}$$

$$\textsc{sk}: \quad h^\alpha \cdot \boxed{\mathsf{R}_0(\textsc{id})} \cdot h_{123}^{\sum \mathbf{w}[2i - \textsc{id}[i]] \cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

The next step is to gradually increase the dependence of R_0's output on its input, from 0 bit to n bits. Here we need n computationally arguments and arise $\mathcal{O}(n)$ security loss but unrelated to the number of secret keys or ciphertexts. Technically, each step relies on the orthogonality of subgroup $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ to independently adopt the proof technique introduced by Chen and Wee [CW13] in each of the two subgroups. Finally, we will reach the scenario with

$$\textsc{ct}: \quad (g_1g_2)^s, \qquad (g_1g_2)^{\sum \mathbf{w}[2i-\textsc{id}[i]]\cdot s}, \qquad e((g_1g_2)^s, h^\alpha \cdot \boxed{\mathsf{R}_n(\textsc{id})}) \cdot \textsc{m}$$

$$\textsc{sk}: \quad h^\alpha \cdot \boxed{\mathsf{R}_n(\textsc{id})} \cdot h_{123}^{\sum \mathbf{w}[2i-\textsc{id}[i]]\cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

where now R_n's output depends on all bits of identity $\textsc{id}$. Combining the feature of random function and the entropy of s, the random coin for encryption, we can readily show the pseudo-randomness of term $e((g_1g_2)^s, h^\alpha \cdot \mathsf{R}_n(\textsc{id}))$ in all ciphertexts simultaneously in a tight fashion.

Our Attempts and Solution. At first glance, we can apply Bellare *et al.*'s idea directly to Hofheinz, Koch, and Striecks' construction. We may remove the master secret key h^α and drop payload mask $e(g_1,h)^\alpha$, put the structure into a bilinear group of order $N = p_1p_2p_3p_4p_5$ and use the extra subgroup $\mathbb{G}_{p_5}$ to hide the structure of ciphertexts as [DCIP10, BWY11]. The result is as follows. An encryption of message 0 for identity $\textsc{id} \in \{0,1\}^n$ looks like

$$g_1^s g_5^{t_5}, \qquad g_1^{\sum \mathbf{w}[2i-\textsc{id}[i]]\cdot s} g_5^{t_5'}$$

where $g_5^{t_5}, g_5^{t_5'} \in \mathbb{G}_{p_5}$ function as $g_4^{t_4}$ and $g_4^{t_4'}$, while $g_1 \in \mathbb{G}_{p_1}$ and $g_1^{\mathbf{w}} \in \mathbb{G}_{p_1}^{2n}$ should be multiplied by a random element of $\mathbb{G}_{p_5}$ before they are published with g_5 as the master public key. On the other hand, secret keys for $\textsc{id}$ should look like

$$h_{123}^{\sum \mathbf{w}[2i-\textsc{id}[i]]\cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

It is easy to verify the correctness of this construction. However the proof technique for proving IND-mCPA is not applicable now. We remark that it is the master secret α that allows us to introduce the random function into the system after introducing $\mathbb{G}_{p_2}$-components into all ciphertexts.

A natural idea to overcome this problem may be to take h^α back in secret keys and put an additional term $e(g_1,h)^{\alpha s}$ in ciphertexts. However we still don't know how to prove the resulting scheme to be IND-mCPA yet even though Hofheinz *et al.*'s proof technique now works. To see that we remind the reader of the fact that our technical goal is to show all components in ciphertexts of message 0 are pseudo-random. The proof technique here finally allows us to argue the pseudo-randomness of $e(g_1,h)^{\alpha s}$, but not $g_1^{\sum \mathbf{w}[2i-\textsc{id}[i]]\cdot s} g_5^{t_5'}$, since no additional entropy has been introduced in it during the proof. It is worth noting that Attrapadung *et al.* [AHY15] showed how to tightly obtain anonymity from a similar structure (using prime-order bilinear groups), i.e., the IND-mCPA in our setting. However they required the key generation procedure to be somewhat deterministic. Namely they use the same random coins for generating secret keys for the same identity. In our paper, we continue to work with the classical definition where each secret key is created from fresh random coins.

Our solution is to turn to the original version of Waters Hash [Wat05] where the encoding of identity $\textsc{id}$ is not linear but affine. In particular, we define an encryption for message 0 as

$$g_1^s g_5^{t_5}, \qquad g_1^{(u+\sum \mathbf{w}[2i-\textsc{id}[i]])\cdot s} g_5^{t_5'}$$

where the newly introduced $g_1^u \in \mathbb{G}_{p_1}$ is also published as a part of master public key after hidden by a random element of $\mathbb{G}_{p_5}$; secret keys are defined as

$$h_{123}^{(u+\sum \mathbf{w}[2i-\textsc{id}[i]])\cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

where u is also introduced to maintain correctness. At this time, we can insert a random function relying on the entropy of $u \in \mathbb{Z}_N$ and continue the proof procedure, which circumvents the issues in our first attempt. In particular, the first step of the proof now results in the following ciphertexts and secret keys

$$\textsc{ct}: \quad (g_1 g_2)^s g_5^{t_5}, \qquad (g_1 g_2)^{(u+\sum \mathbf{w}[2i-\textsc{id}[i]])\cdot s} g_5^{t_5'} \cdot \boxed{g_2^{\mathsf{R}_0(\textsc{id})\cdot s}}$$

$$\textsc{sk}: \quad \boxed{h_{23}^{\mathsf{R}_0(\textsc{id})\cdot r}} \cdot h_{123}^{(u+\sum \mathbf{w}[2i-\textsc{id}[i]])\cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

where we define the random function mapping from identity space to $\mathbb{Z}_N$ instead of $\mathbb{G}_{p_2p_3}$. We note that, different from Hofheinz *et al.*'s proof, the random function in our proof is associated with random coin r in secret keys instead of standing alone, and is associated with random coin s directly in ciphertexts instead of connected through bilinear map e. Finally, we will reach the following configuration using Hofheinz *et al.*'s technique with tiny technical tuning due to our different definition of random function R_0 and R_n.

$$\text{CT}: \quad (g_1g_2)^s g_5^{t_5}, \qquad (g_1g_2)^{(u+\sum \mathbf{w}[2i-\text{ID}[i]])\cdot s} g_5^{t_5'} \cdot \boxed{g_2^{\mathsf{R}_n(\text{ID})\cdot s}}$$

$$\text{SK}: \quad \boxed{h_{23}^{\mathsf{R}_n(\text{ID})\cdot r}} \cdot h_{123}^{(u+\sum \mathbf{w}[2i-\text{ID}[i]])\cdot r} \cdot R_4, \qquad h_{123}^r \cdot R_4'$$

At this moment, we see that the pseudo-random terms $g_2^{\mathsf{R}_n(\text{ID})\cdot s}$ and $h_{23}^{\mathsf{R}_n(\text{ID})\cdot r}$ are bound with $g_1^{(u+\sum \mathbf{w}[2i-\text{ID}[i]])\cdot s}$ and $h_{123}^{(u+\sum \mathbf{w}[2i-\text{ID}[i]])\cdot r}$ respectively, which facilitates the proof of pseudo-randomness of the entire ciphertexts and circumvents the problem in our second attempt. We show the formal description of this construction and its security analysis in Section 4.

1.3 Related Work

Identity Based Encryption. The notion of identity based encryption was introduced by Shamir [Sha85] to alleviate the cost of key management in traditional PKI framework. The first practical solution with formal security analysis for this promising primitive was found by Boneh and Franklin [BF01] in 2001 using bilinear groups. A bunch of constructions are proposed in the next several years, including classical Boneh-Boyen [BB04b, BB04a], Waters [Wat05], and Gentry [Gen06] construction.

A recent breakthrough in this field is the introduction of dual system technique by Waters [Wat09], which is now widely used for building various types of adaptively secure functional encryptions [LW10, LOS+10, OT10, LW11, CLL+13, Lew12, LW12, OT12b, OT12a, RCS12, JR13]. There also appears general frameworks investigating the dual system technique, especially its application in functional encryption [Att14, BKP14, Wee14, CGW15].

In 2013, Chen and Wee [CW13] eastablished the first IBE scheme with almost-tight reduction to static assumptions in the standard model, combining the dual system technique with Naor-Reingold technique for pseudo-random functions [NR04]. The method was extended to the "multi-instance, multi-ciphertext" setting by Hofheinz, Koch, and Striecks [HKS15]. Two very recent work [AHY15, GCD+15] gave the prime-order realization of Hofheinz *et al.*'s composite-order construction. Besides that the work by Attrapadung *et al.* [AHY15] also proposed a framework for building almost-tight IBE from broadcast encodings [Att14], which leads to a lot of new constructions with diverse features.

Selective Opening Security. The study of selective opening security started from the field of public key encryption. Since the work of Bellare, Hofheinz, and Yilek [BHY09], the community obtained several solutions [FHKW10, HLOV11, Hof12, HLQ13, HJKS15, LP15] covering various settings such as chosen plaintext attack (SO-CPA) and chosen ciphertext attack (SO-CCA). In 2011, Bellare, Hofheinz, and Yilek brought SOA security into the field of IBE [BWY11] by considering SO-CPA security. The blank of SO-CCA security was recently filled by Lai *et al.* [LDL+14] relying on several newly introduced primitives. Our paper only considers SO-CPA following [BWY11]. Very recently, He *et al.* [HLL+15] gave an IBE scheme achieving indistinguishability-based SOA security under chosen-plaintext attack in the selective identity model, which is weaker than the security model used in [BWY11, LDL+14] and ours as well.

1.4 Roadmap

The remainder of our paper is organized in three parts. We first define some notations and review several concepts and security models in Section 2. Section 3 shows a generic construction from IND-mCPA one-bit IBE to SOA-secure multi-bit IBE with tight reduction. Such a one-bit IBE is presented in Section 4 followed by a series of static assumptions in composite-order bilinear groups. Finally, we conclude the paper in Section 5.

2 Preliminaries

2.1 Notations

We employ $x := f(y)$ to denote the process of assigning to x the result of $f(y)$ for some formula $f(\cdot)$ and some value y. For any positive number n, we define $[n] := \{1, \ldots, n\}$. For any list or vector $\mathbf{w}$, we let $\mathbf{w}[i]$ denote the ith entry of $\mathbf{w}$. Similarly, for any binary string $\textsc{id} \in \{0,1\}^*$, we use $\textsc{id}[i]$ to indicate the ith bit of $\textsc{id}$. The notation $y \leftarrow \mathsf{Alg}(x_1,\ldots,x_n; r_1,\ldots,r_m)$ or $\mathsf{Alg}(x_1,\ldots,x_n; r_1,\ldots,r_m) \rightarrow y$ refer to the process of running algorithm Alg with inputs $x_1,\ldots,x_n$ and random coins $r_1,\ldots,r_m$, then assigning the result to variable y. The random coins may be omitted for brevity. We may also use a more compact form $y \leftarrow \mathsf{Alg}(\mathbf{x};\mathbf{r})$ where $\mathbf{x} := (x_1,\ldots,x_n)$ and $\mathbf{r} := (r_1,\ldots,r_m)$. For any fixed input $\mathbf{x}$, the set $[\mathsf{Alg}(\mathbf{x})]$ is defined as $\{y : \exists \mathbf{r} \text{ s.t. } \mathsf{Alg}(\mathbf{x};\mathbf{r}) = y\}$.

Given a finite cyclic group $\mathbb{G}$, we use $X \leftarrow \mathbb{G}$ to denote the process of sampling a random element in $\mathbb{G}$. In particular, the notation is for the so-called "lazy sampling". Assuming $g \in \mathbb{G}$ is a generator of group $\mathbb{G}$ of order N, we sample X from $\mathbb{G}$ by randomly sampling x from $\mathbb{Z}_N$ and set $X := g^x$. As [BWY11], we consider sampling random element from $\mathbb{Z}_N$ (for any $N \in \mathbb{Z}_{>0}$) as the unique random source in our system, and denote this atomic process by $x \leftarrow \mathbb{Z}_N$. For any element $g \in \mathbb{G}$ and list or vector $\mathbf{w} = (w_1,\ldots,w_n) \in \mathbb{Z}_N^n$, we define $g^{\mathbf{w}} := (g^{w_1},\ldots,g^{w_n}) \in \mathbb{G}^n$. Given $\mathbf{g} = (g_1,\ldots,g_n)$ and $\mathbf{h} = (h_1,\ldots,h_n)$, two vectors over $\mathbb{G}$, we define $\mathbf{g}\cdot\mathbf{h} := (g_1\cdot h_1,\ldots,g_n\cdot h_n) \in \mathbb{G}^n$.

2.2 Code-Based Games

As [BWY11], we employ code based games [BR06] for our security definitions and proofs. A code based game is defined by an **Initialize** procedure and a **Finalize** procedure plus a series of procedures, which will be used to answer adversary $\mathscr{A}$'s queries and depends on the security notion we concern. The game begins with running **Initialize** procedure and transmitting the result to adversary $\mathscr{A}$. During the game, $\mathscr{A}$ is allowed to make various types of queries in any order. In general, $\mathscr{A}$ is capable of making polynomial-many queries. Finally, $\mathscr{A}$ is expected to return an output before it halts. The output of the game is then obtained by invoking **Finalize** procedure on $\mathscr{A}$'s output. As usual, we let $\mathsf{Game}_{\mathscr{A}}(\lambda) = y$ denote the event that the output of executing Game with adversary $\mathscr{A}$ on security parameter 1^λ is y.

2.3 Identity Based Encryptions

Algorithms. An identity-based encryption scheme with identity space IdSp and message space MsgSp consists of four (probabilistic) polynomial time algorithms defined as follows:

- $\mathsf{Setup}(1^\lambda) \rightarrow (\textsc{mpk},\textsc{msk})$. The setup algorithm takes as input a security parameter 1^λ, and outputs a master public key $\textsc{mpk}$ and the corresponding master secret key $\textsc{msk}$.
- $\mathsf{KeyGen}(\textsc{mpk},\textsc{msk},\textsc{id}) \rightarrow \textsc{sk}$. The key generation algorithm takes as input a master public key $\textsc{mpk}$, a master secret key $\textsc{msk}$ and an identity $\textsc{id} \in \mathsf{IdSp}$, and outputs a secret key $\textsc{sk}$.
- $\mathsf{Enc}(\textsc{mpk},\textsc{id},\textsc{m}) \rightarrow \textsc{ct}$. The encryption algorithm takes as input a master public key $\textsc{mpk}$, an identity $\textsc{id} \in \mathsf{IdSp}$ and a message $\textsc{m} \in \mathsf{MsgSp}$, outputs a ciphertext $\textsc{ct}$.
- $\mathsf{Dec}(\textsc{mpk},\textsc{sk},\textsc{ct}) \rightarrow \textsc{m}$. The decryption algorithm takes as input a master public key $\textsc{mpk}$, a secret key $\textsc{sk}$ and a ciphertext $\textsc{ct}$, outputs a message $\textsc{m}$ or a failure symbol $\perp$.

Correctness. The correctness (with negligible failure probability) requires that, for all security parameter λ, for all $(\textsc{mpk},\textsc{msk}) \in [\mathsf{Setup}(1^\lambda)]$, all $\textsc{id} \in \mathsf{IdSp}$, and all $\textsc{m} \in \mathsf{MsgSp}$, it holds that

$$\Pr[\mathsf{Dec}(\textsc{mpk},\mathsf{KeyGen}(\textsc{mpk},\textsc{msk},\textsc{id}),\mathsf{Enc}(\textsc{mpk},\textsc{id},\textsc{m})) = \textsc{m}] \geqslant 1-\epsilon,$$

where ϵ is negligible in λ and the probability space is defined by random coins consumed by algorithm KeyGen and Enc.

More Notations. We now define two sets for an IBE scheme. We let $\mathsf{Coins}(\textsc{mpk},\textsc{m})$ be the set of random coins used to encrypt message $\textsc{m}$ for all $(\textsc{mpk},\textsc{msk}) \in [\mathsf{Setup}(1^\lambda)]$. We also use $\mathsf{Coins}(\textsc{mpk},\textsc{id},\textsc{ct},\textsc{m})$ to denote $\{r : \textsc{ct} = \mathsf{Enc}(\textsc{mpk},\textsc{id},\textsc{m};r)\}$, the set of all random coins which makes algorithm Enc to produce $\textsc{ct}$ as the ciphertext for message $\textsc{m}$ under $\textsc{mpk}$ and $\textsc{id}$.

2.4 Security against Selective Opening Attacks

Bellare *et al.* [BWY11] formally defined selective opening security in the setting of IBE based on the work of [BHY09]. This subsection review their definition through two games: SOAREAL and SOASIM, defined in Figure 1 and Figure 2, respectively. In the figures, $\mathscr{M}$ is called (k,ℓ)-message sampler which takes $\alpha \in \{0,1\}^*$ as input and returns $\mathbf{m}$ such that $|\mathbf{m}| = k$ and $\mathbf{m}[i] \in \{0,1\}^\ell$ for all $i \in [k]$, and $\mathscr{R}$ is any randomized algorithm with binary output. In both games, an adversary must make one **NewMesg** query before one **Corrupt** query. Besides, game SOAREAL additionally allows the adversary to adaptively make polynomially-many **Extract** queries.

Initialize	
$(\textsc{mpk}, \textsc{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$	**Extract**$(\textsc{id})$
return $\textsc{mpk}$	**if** $\textsc{id} \in \mathsf{ChID}$ **then return** $\perp$
	$\mathsf{ExID} := \mathsf{ExID} \cup \{\textsc{id}\}$
NewMesg$(\mathbf{id}, \alpha)$	**return** $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \textsc{id})$
if $\mathbf{id} \cap \mathsf{ExID} \neq \emptyset$ **then return** $\perp$	
$\mathsf{ChID} := \mathsf{ChID} \cup \mathbf{id}$	**Corrupt**(I)
$\mathbf{m} \leftarrow \mathscr{M}(\alpha)$	**return** $\mathbf{r}[I], \mathbf{m}[I]$
for $i \in [k]$ **do**	
$\mathbf{r}[i] \leftarrow \mathsf{Coins}(\textsc{mpk}, \mathbf{m}[i])$	**Finalize**$(\textsc{out})$
$\mathbf{ct}[i] \leftarrow \mathsf{Enc}(\textsc{mpk}, \mathbf{id}[i], \mathbf{m}[i]; \mathbf{r}[i])$	**return** $\mathscr{R}(\mathbf{m}, \mathsf{ChID}, I, \textsc{out})$
return ct	

Figure 1: SOAREAL Game

Initialize	
return $\perp$	**Corrupt**(I)
	return $\mathbf{m}[I]$
NewMesg$(\mathbf{id}, \alpha)$	
$\mathsf{ChID} := \mathsf{ChID} \cup \mathbf{id}$	**Finalize**$(\textsc{out})$
$\mathbf{m} \leftarrow \mathscr{M}(\alpha)$	**return** $\mathscr{R}(\mathbf{m}, \mathsf{ChID}, I, \textsc{out})$
return $\perp$	

Figure 2: SOASIM Game

The SOA-advantage function of SOA-adversary $\mathscr{A}$ against an SOA-simulator $\mathscr{S}$ with respect to $\mathscr{M}$ and $\mathscr{R}$ is defined as follows.

$$\mathsf{Adv}^{\mathsf{SOA}}_{\mathscr{A},\mathscr{S},\mathscr{M},\mathscr{R}}(\lambda) := \left|\Pr\left[\mathsf{SOAREAL}_{\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1\right] - \Pr\left[\mathsf{SOASIM}_{\mathscr{S},\mathscr{M},\mathscr{R}}(\lambda) = 1\right]\right|.$$

An IBE scheme is SOA secure if and only if, for any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any probabilistic polynomial time SOA-adversary $\mathscr{A}$, there exists an SOA-simulator $\mathscr{S}$ such that the advantage function $\mathsf{Adv}^{\mathsf{SOA}}_{\mathscr{A},\mathscr{S},\mathscr{M},\mathscr{R}}(\lambda)$ is negligible in λ.

2.5 Indistinguishability with One-sided Publicly Openability

The ciphertext indistinguishability under chosen plaintext attack [BF01] (IND-CPA) is one of well-known security definitions for IBE. In the model, given master public key, an adversary is able to obtain polynomially-many secret keys adaptively and a *single* challenge ciphertext for one of challenge messages, and is asked to guess a secret bit (which indicates which challenge message is used to generate challenge ciphertext). We consider IND-CPA in the multi-ciphertext setting (IND-mCPA) where the adversary now has access to *polynomially-many* challenge ciphertexts, which is recently proposed and investigated in more general multi-instance, multi-ciphertext setting [HKS15].

More formally, we review the notion of IND-mCPA [HKS15] through game INDmCPA shown in Figure 3. The advantage function of adversary $\mathscr{A}$ in game INDmCPA is defined as

$$\mathsf{Adv}^{\mathsf{INDmCPA}}_{\mathscr{A}}(\lambda) := \left|\Pr\left[\mathsf{INDmCPA}_{\mathscr{A}}(\lambda) = 1\right] - 1/2\right|.$$

$$
\begin{array}{ll}
\underline{\textbf{Initialize}} & \\
(\text{MPK}, \text{MSK}) \leftarrow \mathsf{Setup}(1^\lambda) & \underline{\textbf{Challenge}(\text{ID}^*, \text{M}_0^*, \text{M}_1^*)} \\
\beta \leftarrow \{0,1\} & \textbf{if } \text{ID}^* \in \mathsf{ExID} \textbf{ then return } \perp \\
\textbf{return } \text{MPK} & \mathsf{ChID} := \mathsf{ChID} \cup \{\text{ID}^*\} \\
 & \textbf{return } \mathsf{Enc}(\text{MPK}, \text{ID}^*, \text{M}_\beta^*) \\
\underline{\textbf{Extract}(\text{ID})} & \\
\textbf{if } \text{ID} \in \mathsf{ChID} \textbf{ then return } \perp & \underline{\textbf{Finalize}(\beta')} \\
\mathsf{ExID} := \mathsf{ExID} \cup \{\text{ID}\} & \textbf{return } (\beta = \beta') \\
\textbf{return } \mathsf{KeyGen}(\text{MPK}, \text{MSK}, \text{ID}) &
\end{array}
$$

Figure 3: INDmCPA Game

An IBE scheme is said to be IND-mCPA if and only if the advantage function $\mathsf{Adv}_{\mathscr{A}}^{\mathsf{INDmCPA}}(\lambda)$ is negligible in λ for any probabilistic polynomial time adversary $\mathscr{A}$.

However the IND-mCPA alone is not sufficient for realizing SOA security. We require an additional property, called *One-Sided Publicly Openability* (1SPO), proposed by Bellare *et al.* [BWY11] based on the concept of deniable PKE [CDNO97]. Roughly speaking, 1SPO for an IBE with $\mathsf{MsgSp} = \{0,1\}$ means that one can publicly open a ciphertext for message 1 by presenting its random coins. More formally, we review the following algorithm.

- $\mathsf{OpenToOne}(\text{MPK}, \text{ID}, \text{CT}) \to r$. The setup algorithm takes as input a master public key MPK, an identity $\text{ID} \in \mathsf{IdSp}$ and a ciphertext CT of 1, outputs a random coin r such that $\text{CT} = \mathsf{Enc}(\text{MPK}, \text{ID}, 1; r)$.

We allow algorithm OpenToOne to output failure symbol $\perp$ with probability δ and require that, for all $\text{MPK} \in [\mathsf{Setup}(1^\lambda)]$, all $\text{ID} \in \mathsf{IdSp}$, all $\text{CT} \in [\mathsf{Enc}(\text{MPK}, \text{ID}, 1)]$, and all $\widehat{r} \in \mathsf{Coins}(\text{MPK}, \text{ID}, \text{CT}, 1)$,

$$
\Pr\left[r \leftarrow \mathsf{OpenToOne}(\text{MPK}, \text{ID}, \text{CT}) : r = \widehat{r} | r \neq \perp\right] = \frac{1}{|\mathsf{Coins}(\text{MPK}, \text{ID}, \text{CT}, 1)|}
$$

The algorithm is called δ-1SP opener and an IBE with a δ-1SP opener is called δ-one-sided publicly openable (δ-1SPO).

3 Tightly Reducing SOA Security to IND-CPA with δ-1SPO

Bellare *et al.* [BWY11] presented a trivial construction for ℓ-bit IBE IBE^ℓ from one-bit IBE IBE, and proved that IBE^ℓ achieves SOA security if the underlying IBE is IND-CPA with δ-1SPO. Their reduction has *polynomial security loss*. We review this trivial construction (with different identity space) and prove that IBE^ℓ achieves SOA security if the underlying IBE is IND-mCPA with δ-1SPO. We emphasize that our reduction only has *constant security loss*.

3.1 Construction

Assume an one-bit IBE $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with $\mathsf{IdSp} = \{0,1\}^n$ and $\mathsf{MsgSp} = \{0,1\}$. The ℓ-bit IBE $\mathsf{IBE}^\ell = \left(\mathsf{Setup}^\ell, \mathsf{KeyGen}^\ell, \mathsf{Enc}^\ell, \mathsf{Dec}^\ell\right)$ with $\mathsf{IdSp} = \{0,1\}^n$ (identical to IBE) and $\mathsf{MsgSp} = \{0,1\}^\ell$ is defined as in Figure 4.

$$
\begin{array}{ll}
\underline{\mathsf{Setup}^\ell(1^\lambda, n)} & \underline{\mathsf{KeyGen}^\ell(\text{MPK}, \text{MSK}, \text{ID})} \\
\textbf{return } \mathsf{Setup}(1^\lambda, n) & \textbf{return } \mathsf{KeyGen}(\text{MPK}, \text{MSK}, \text{ID}) \\
 & \\
\underline{\mathsf{Dec}^\ell(\text{MPK}, \text{SK}, \mathbf{ct})} & \underline{\mathsf{Enc}^\ell(\text{MPK}, \text{ID}, \text{M})} \\
\textbf{for } i \in [\ell] \textbf{ do} & \textbf{for } i \in [\ell] \textbf{ do} \\
\quad \text{M}[i] \leftarrow \mathsf{Dec}(\text{MPK}, \text{SK}, \mathbf{ct}[i]) & \quad \mathbf{ct}[i] \leftarrow \mathsf{Enc}(\text{MPK}, \text{ID}, \text{M}[i]) \\
\textbf{return } \text{M} & \textbf{return } \mathbf{ct}
\end{array}
$$

Figure 4: From IBE to IBE^ℓ.

It is quite clear that the correctness of the resulting IBE follows from the underlying one. More concretely, if IBE is correct with failure probability ϵ, IBE^ℓ is correct with failure probability $\ell \cdot \epsilon$.

3.2 Security Analysis

We want to prove the following theorem.

Theorem 1 *For any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any probabilistic polynomial time SOA-adversary $\mathscr{A}$ making at most Q_K* **Extract** *queries, there exists a probabilistic polynomial time SOA-simulator $\mathscr{S}$ and adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathsf{SOA}}_{\mathscr{A},\mathscr{S},\mathscr{M},\mathscr{R}}(\lambda) \le 2 \cdot \mathsf{Adv}^{\mathsf{INDmCPA}}_{\mathscr{B}}(\lambda) + k\ell \cdot \delta.$$

The proof begins with constructing a SOA-simulator $\mathscr{S}$ in SOASIM game, which takes any probabilistic polynomial time SOA-adversary $\mathscr{A}$ as oracle. Our proof continues employing Bellare *et al.*'s construction and we recall it in Figure 5. The SOA-simulator $\mathscr{S}$ prepares a fresh master public/secret key pair ($\textsc{mpk}, \textsc{msk}$) and initializes a SOA-adversary $\mathscr{A}$ using $\textsc{mpk}$. To obtain $\mathscr{A}$'s output, $\mathscr{S}$ answers all queries made by it, i.e., simulating $\mathsf{SOAREAL}$ game for $\mathscr{A}$. The **Extract** queries are directly answered using $\textsc{msk}$. For the **NewMesg** query, SOA-simulator $\mathscr{S}$ returns k encryptions of message 0^ℓ. In the figure, we use $\mathbf{NewMesg}_{\mathscr{S}}$ to denote the **NewMesg** oracle for $\mathscr{S}$ in SOASIM game, which returns nothing to $\mathscr{S}$. For the **Corrupt** query, $\mathscr{S}$ opens corrupted messages correctly relying on the power of $\mathsf{OpenToOne}$ algorithm. In particular, if the corrupted message bit is 0, the random coin used when answering **NewMesg** query could be returned directly; if the bit is 1 instead, the random coin must be resampled so as to explain the corresponding ciphertext $\mathscr{S}$ has given to $\mathscr{A}$, which is originally an encryption of 0, to be an encryption of 1.

$\underline{\mathsf{main}()}$ $(\textsc{mpk}, \textsc{msk}) \leftarrow \mathsf{Setup}^\ell(1^\lambda)$ **return** $\mathscr{A}(\textsc{mpk})$ $\underline{\mathbf{NewMesg}(\mathbf{id}, \alpha)}$ $\underline{\mathbf{NewMesg}_{\mathscr{S}}(\mathbf{id}, \alpha)}$ **for** $i \in [k]$ **do** $\quad \mathbf{r}[i] \leftarrow \mathsf{Coins}(\textsc{mpk}, 0^\ell)$ $\quad \mathbf{ct}[i] \leftarrow \mathsf{Enc}^\ell(\textsc{mpk}, \textsc{id}[i], 0^\ell; \mathbf{r}[i])$ **return ct**	$\underline{\mathbf{Corrupt}(I)}$ $\mathbf{m}[I] \leftarrow \mathbf{Corrupt}_{\mathscr{S}}(I)$ **for** $i \in I$ **and** $j \in [\ell]$ **do** $\quad$**if** $\mathbf{m}[i][j] = 1$ **then** $\quad\quad \mathbf{r}[i][j] \leftarrow \mathsf{OpenToOne}(\textsc{mpk}, \mathbf{id}[i], \mathbf{ct}[i][j])$ **return** $\mathbf{r}[I], \mathbf{m}[I]$ $\underline{\mathbf{Extract}(\textsc{id})}$ **return** $\mathsf{KeyGen}^\ell(\textsc{mpk}, \textsc{msk}, \textsc{id})$

Figure 5: SOA-simulator $\mathscr{S}$.

One may see that the simulation of $\mathscr{S}$ is different from the specification of $\mathsf{SOAREAL}$. Therefore the next step of the proof is to show that the simulation described in Figure 5 and the real $\mathsf{SOAREAL}$ are computationally indistinguishable from the viewpoint of $\mathscr{A}$. In detail, we employ hybrid argument using the game sequence shown in Figure 6.

We have the following lemmas immediately.

Lemma 1 *For any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any adversary $\mathscr{A}$,*

$$\left|\Pr[\mathsf{Game}_{0,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1] - \Pr[\mathsf{Game}_{1,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1]\right| \le k\ell \cdot \delta.$$

Lemma 2 *For any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any adversary $\mathscr{A}$,*

$$\Pr[\mathsf{Game}_{2,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1] = \Pr[\mathsf{Game}_{3,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1].$$

Lemma 3 *For any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any adversary $\mathscr{A}$,*

$$\Pr[\mathsf{Game}_{3,\mathscr{A}}(\lambda) = 1] = \Pr[\mathsf{SOASIM}_{\mathscr{S},\mathscr{M},\mathscr{R}}(\lambda) = 1].$$

The first lemma follows from the fact that the algorithm $\mathsf{OpenToOne}$ is not perfect, its failure probability is δ, and there are at most $k\ell$ applications of $\mathsf{OpenToOne}$. For the second lemma, since we can answer **NewMesg** queries without knowing anything about $\mathbf{m}$, it is safe to defer the sampling of $\mathbf{m}$. The last lemma follows the observation that the SOA-simulator described in Figure 5 is able to simulate Game_3 and $\mathscr{S}$'s output always equals $\mathscr{A}$'s.

To finish the proof of the theorem, we must fill the gap between Game_1 and Game_2. Especially, we prove the following lemma.

Game_0, Game_1, $\boxed{\mathsf{Game}_2}$, Game_3

Initialize
$(\text{MPK}, \text{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$
return MPK

Extract(ID)
if ID $\in \mathsf{ChID}$ **then return** $\perp$
$\mathsf{ExID} := \mathsf{ExID} \cup \{\text{ID}\}$
return $\mathsf{KeyGen}(\text{MPK}, \text{MSK}, \text{ID})$

Finalize(OUT)
return $\mathscr{R}(\mathbf{m}, \mathsf{ChID}, I, \text{OUT})$

NewMesg($\mathbf{id}, \alpha$)
if $\mathbf{id} \cap \mathsf{ExID} \neq \emptyset$ **then return** $\perp$
$\mathsf{ChID} := \mathsf{ChID} \cup \mathbf{id}$
$\mathbf{m} \leftarrow \mathscr{M}(\alpha)$
$\mathbf{m} \leftarrow \perp$
for $i \in [k]$ **and** $j \in [\ell]$ **do**
  $\mathbf{r}[i][j] \leftarrow \mathsf{Coins}(\text{MPK}, \mathbf{m}[i][j])$
  $\boxed{\mathbf{r}[i][j] \leftarrow \mathsf{Coins}(\text{MPK}, 0)}$
  $\mathbf{ct}[i][j] \leftarrow \mathsf{Enc}(\text{MPK}, \mathbf{id}[i], \mathbf{m}[i][j]; \mathbf{r}[i][j])$
  $\boxed{\mathbf{ct}[i][j] \leftarrow \mathsf{Enc}(\text{MPK}, \mathbf{id}[i], 0; \mathbf{r}[i][j])}$
return ct

Corrupt(I)
$\mathbf{m} \leftarrow \mathscr{M}(\alpha)$
for $i \in I$ **and** $j \in [\ell]$ **do**
  if $\mathbf{m}[i][j] = 1$**then**
    $\mathbf{r}[i][j] \leftarrow \mathsf{OpenToOne}(\text{MPK}, \mathbf{id}[i], \text{CT}[i][j])$
return $\mathbf{r}[I], \mathbf{m}[I]$

Figure 6: Game Sequence for SOA security

Lemma 4 ($\mathsf{Game}_1 \approx \mathsf{Game}_2$) *For any message sampler $\mathscr{M}$, any binary relation $\mathscr{R}$, any probabilistic polynomial time adversary $\mathscr{A}$ making at most Q_K key extraction queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\Pr\left[\mathsf{Game}_{1,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1\right] - \Pr\left[\mathsf{Game}_{2,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1\right]\right| \leq 2 \cdot \mathsf{Adv}_{\mathscr{B}}^{\mathsf{INDmCPA}}(\lambda).$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (k\ell + Q_K) \cdot \mathsf{poly}(\lambda)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. Given MPK and oracle access to $\mathbf{Extract}_{\mathscr{B}}$ and **Challenge**, algorithm $\mathscr{B}$ proceeds as follows:

Initialize Return MPK.

Extract(ID) Return $\mathbf{Extract}_{\mathscr{B}}(\text{ID})$.

NewMesg($\mathbf{id}, \alpha$) Return $\perp$ when $\mathbf{id} \cap \mathsf{ExID} \neq \emptyset$. Update $\mathsf{ChID} := \mathsf{ChID} \cup \mathbf{id}$. Sample $\mathbf{m} \leftarrow \mathscr{M}(\alpha)$. For $i \in [k]$ and $j \in [\ell]$, if $\mathbf{m}[i][j] = 1$, set

$$\mathbf{ct}[i][j] \leftarrow \mathbf{Challenge}(\mathbf{id}[i], 0, 1),$$

otherwise, sample

$$\mathbf{r}[i][j] \leftarrow \mathsf{Coins}(\text{MPK}, 0) \quad \text{and} \quad \mathbf{ct}[i][j] \leftarrow \mathsf{Enc}(\text{MPK}, \mathbf{id}[i], 0; \mathbf{r}[i][j]).$$

Return **ct**.

Corrupt(I) For $i \in I$ and $j \in [\ell]$, if $\mathbf{m}[i][j] = 1$, set

$$\mathbf{r}[i][j] \leftarrow \mathsf{OpenToOne}(\text{MPK}, \mathbf{id}[i], \mathbf{ct}[i][j]).$$

Return $(\mathbf{r}[I], \mathbf{m}[I])$.

When $\mathscr{A}$ halts with output OUT, algorithm $\mathscr{B}$ outputs $\mathscr{R}(\mathbf{m}, \mathsf{ChID}, I, \text{OUT})$.

Observe that if $\beta = 0$, the outputs of **Challenge** are encryptions of 0 and the simulation is identical to Game_2. On the other hand, if $\beta = 1$, the outputs of **Challenge** are encryptions of 1 and the simulation is identical to Game_1. Therefore we can conclude that $\left|\Pr\left[\mathsf{Game}_{1,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1\right] - \Pr\left[\mathsf{Game}_{2,\mathscr{A},\mathscr{M},\mathscr{R}}(\lambda) = 1\right]\right| \leq 2 \cdot \mathsf{Adv}_{\mathscr{B}}^{\mathsf{INDmCPA}}(\lambda)$. ☐

4 Tightly Reducing IND-CPA IBE with δ-1SPO to Static Assumptions

4.1 Composite-order Bilinear Groups

We assume a group generator GrpGen for composite-order bilinear groups, which takes as input a security parameter 1^λ, outputs $(N, \mathbb{G}, \mathbb{G}_T, e, p_1, \ldots, p_5)$ where $N = p_1 \cdots p_5$ and all p_i are prime numbers in $[2^{\lambda-1}, 2^\lambda - 1]$, both $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an admissible bilinear map. We let $\mathbb{G}$ and $\mathbb{G}_T$ also contain their generators, denoted by $g \in \mathbb{G}$ and $g_T \in \mathbb{G}_T$, respectively. We define public group description $\mathcal{G} = (N, \mathbb{G}, \mathbb{G}_T, e)$ and take factoring of N, i.e., $p_1, \ldots, p_5$, as secret information. It is the secret information that allow us to derive the generator of each subgroup from the generator of the entire group, i.e., $g \in \mathbb{G}$, and sample random element from it.

For positive integer N' with $N'|N$, we use $\mathbb{G}_{N'}$ to indicate the unique subgroup of order N'. Given $g \in \mathbb{G}_n$ and $h \in \mathbb{G}_m$ such that $\gcd(n, m) = 1$, we have $e(g, h) = 1_{\mathbb{G}_T}$. In fact, we can decompose $\mathbb{G}$ as $\mathbb{G}_{p_1} \times \cdots \times \mathbb{G}_{p_5}$ and write g^x for $x \in \mathbb{Z}_N$ as $\prod_{i=1}^{5} g_i^{x_i}$ where $\mathbb{G}_{p_i}$ is generated by $g_i \in \mathbb{G}_{p_i}$ and $x_i \in \mathbb{Z}_N$ is unique modulo p_i. Under the representation, we call $\prod_{i \in I} g_i^{x_i}$ for some $I \subseteq [5]$ the $\mathbb{G}_{N'}$-part of g^x where $N' = \prod_{i \in I} p_i$.

We need the following six computational assumptions to reach IND-mCPA security. Assumption 1, 2, 5, 6 are concrete instantiations of *General Subgroup Decision Assumption* formalized by Bellare *et al.* [BWY11] which is inspired by several concrete assumptions used in [BGN05, DCIP10, LW10]. Assumption 3 and 4 are modified from Hofheinz *et al.*'s *Dual System Assumption 3* [HKS15] and Chen and Wee's second assumption in [CW13], respectively. In fact the Dual System Assumption 3 [HKS15] was also derived from Chen and Wee's second assumption. All of them could be viewed as Diffie-Hellman Assumption on subgroups of composite-order $\mathbb{G}$.

Assumption 1 *For any probabilistic polynomial time adversary $\mathcal{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{SD1}}(\lambda) := \left|\Pr\left[\mathcal{A}(D, T_0) = 1\right] - \Pr\left[\mathcal{A}(D, T_1) = 1\right]\right|$$

where

$$\begin{aligned}
&(N, \mathbb{G}, \mathbb{G}_T, e, p_1, \ldots, p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathcal{G} = (N, \mathbb{G}, \mathbb{G}_T, e);\\
&g_1 \leftarrow \mathbb{G}_{p_1};\ g_4 \leftarrow \mathbb{G}_{p_4};\ g_5 \leftarrow \mathbb{G}_{p_5};\ X_1X_2X_3 \leftarrow \mathbb{G}_{p_1p_2p_3};\\
&D \leftarrow (\mathcal{G}, g_1, g_4, g_5, X_1X_2X_3);\\
&T_0 \leftarrow \mathbb{G}_{p_1};\ T_1 \leftarrow \mathbb{G}_{p_1p_2}.
\end{aligned}$$

Assumption 2 *For any probabilistic polynomial time adversary $\mathcal{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{SD2}}(\lambda) := \left|\Pr\left[\mathcal{A}(D, T_0) = 1\right] - \Pr\left[\mathcal{A}(D, T_1) = 1\right]\right|$$

where

$$\begin{aligned}
&(N, \mathbb{G}, \mathbb{G}_T, e, p_1, \ldots, p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathcal{G} = (N, \mathbb{G}, \mathbb{G}_T, e);\\
&g_1 \leftarrow \mathbb{G}_{p_1};\ g_4 \leftarrow \mathbb{G}_{p_4};\ g_5 \leftarrow \mathbb{G}_{p_5};\ X_2X_5 \leftarrow \mathbb{G}_{p_2p_5};\ Y_2Y_3 \leftarrow \mathbb{G}_{p_2p_3};\\
&D \leftarrow (\mathcal{G}, g_1, g_4, g_5, X_2X_5, Y_2Y_3);\\
&T_0 \leftarrow \mathbb{G}_{p_2p_5};\ T_1 \leftarrow \mathbb{G}_{p_3p_5}.
\end{aligned}$$

Assumption 3 *For any probabilistic polynomial time adversary $\mathcal{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{DDH1}}(\lambda) := \left|\Pr\left[\mathcal{A}(D, T_0) = 1\right] - \Pr\left[\mathcal{A}(D, T_1) = 1\right]\right|$$

where

$$\begin{aligned}
&(N, \mathbb{G}, \mathbb{G}_T, e, p_1, \ldots, p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathcal{G} = (N, \mathbb{G}, \mathbb{G}_T, e);\\
&g_1 \leftarrow \mathbb{G}_{p_1};\ g_2 \leftarrow \mathbb{G}_{p_2};\ g_3 \leftarrow \mathbb{G}_{p_3};\ g_4 \leftarrow \mathbb{G}_{p_4};\ g_5 \leftarrow \mathbb{G}_{p_5};\\
&\widetilde{X}_4, \widetilde{Y}_4, \overline{X}_4, \overline{Y}_4 \leftarrow \mathbb{G}_{p_4};\ x, y, z \leftarrow \mathbb{Z}_N^*;\\
&D = \left(\mathcal{G}, g_1, g_2, g_3, g_4, g_5, g_2^x\widetilde{X}_4, g_2^y\widetilde{Y}_4, g_3^x\overline{X}_4, g_3^y\overline{Y}_4\right);\\
&T_0 \leftarrow \left(g_2^{xy}, g_3^{xy}\right);\ T_1 \leftarrow \left(g_2^{xy+z}, g_3^{xy+z}\right).
\end{aligned}$$

Assumption 4 *For any probabilistic polynomial time adversary $\mathscr{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}^{\mathsf{DDH2}}_{\mathscr{A}}(\lambda) := \left|\Pr\left[\mathscr{A}(D,T_0)=1\right] - \Pr\left[\mathscr{A}(D,T_1)=1\right]\right|$$

where

$$\begin{aligned}
&(N,\mathbb{G},\mathbb{G}_T,e,p_1,\ldots,p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathscr{G}=(N,\mathbb{G},\mathbb{G}_T,e);\\
&g_1 \leftarrow \mathbb{G}_{p_1};\ g_2 \leftarrow \mathbb{G}_{p_2};\ g_3 \leftarrow \mathbb{G}_{p_3};\ g_4 \leftarrow \mathbb{G}_{p_4};\ g_5 \leftarrow \mathbb{G}_{p_5};\\
&\widetilde{X}_5, \widetilde{Y}_5 \leftarrow \mathbb{G}_{p_5};\ x,y,z \leftarrow \mathbb{Z}_N^*;\\
&D = \left(\mathscr{G}, g_1, g_2, g_3, g_4, g_5, g_2^x\widetilde{X}_5, g_2^y\widetilde{Y}_5\right);\\
&T_0 \leftarrow g_2^{xy};\ T_1 \leftarrow g_2^{xy+z}.
\end{aligned}$$

Assumption 5 *For any probabilistic polynomial time adversary $\mathscr{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}^{\mathsf{SD3}}_{\mathscr{A}}(\lambda) := \left|\Pr\left[\mathscr{A}(D,T_0)=1\right] - \Pr\left[\mathscr{A}(D,T_1)=1\right]\right|$$

where

$$\begin{aligned}
&(N,\mathbb{G},\mathbb{G}_T,e,p_1,\ldots,p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathscr{G}=(N,\mathbb{G},\mathbb{G}_T,e);\\
&g_2 \leftarrow \mathbb{G}_{p_2};\ g_3 \leftarrow \mathbb{G}_{p_3};\ g_4 \leftarrow \mathbb{G}_{p_4};\ g_5 \leftarrow \mathbb{G}_{p_5};\\
&X_1X_5 \leftarrow \mathbb{G}_{p_1p_5};\ Y_1Y_2Y_3 \leftarrow \mathbb{G}_{p_1p_2p_3};\\
&D = (\mathscr{G}, g_2, g_3, g_4, g_5, X_1X_5, Y_1Y_2Y_3);\\
&T_0 \leftarrow \mathbb{G}_{p_2p_5};\ T_1 \leftarrow \mathbb{G}_{p_1p_2p_5}.
\end{aligned}$$

Assumption 6 *For any probabilistic polynomial time adversary $\mathscr{A}$, the advantage function defined as follows are negligible in λ.*

$$\mathsf{Adv}^{\mathsf{SD4}}_{\mathscr{A}}(\lambda) := \left|\Pr\left[\mathscr{A}(D,T_0)=1\right] - \Pr\left[\mathscr{A}(D,T_1)=1\right]\right|$$

where

$$\begin{aligned}
&(N,\mathbb{G},\mathbb{G}_T,e,p_1,\ldots,p_5) \leftarrow \mathsf{GrpGen}(1^\lambda);\ \mathscr{G}=(N,\mathbb{G},\mathbb{G}_T,e);\\
&g_1 \leftarrow \mathbb{G}_{p_1};\ g_2 \leftarrow \mathbb{G}_{p_2};\ g_5 \leftarrow \mathbb{G}_{p_5};\ X_2X_3X_4 \leftarrow \mathbb{G}_{p_2p_3p_4};\\
&D \leftarrow (\mathscr{G}, g_1, g_2, g_5, X_2X_3X_4);\\
&T_0 \leftarrow \mathbb{G}_{p_1p_2p_5};\ T_1 \leftarrow \mathbb{G}.
\end{aligned}$$

To realize the *One-sided Publicly Openability*, we need $\mathbb{G}$ to be equipped with *publicly reversible sampling* [BWY11, FHKW10, LDL+14]. Formally, there exist two algorithms defined as follows.

- $\mathsf{Sample}_{\mathbb{G}}() \rightarrow G$. The publicly reversible sampler takes no input and outputs a random element from group $\mathbb{G}$ with probability $1-\zeta$ or outputs a failure symbol $\perp$ with probability ζ. In particular, for all $G' \in \mathbb{G}$, we require that

  $$\Pr\left[G=G' | G \neq \perp\right] = 1/|\mathbb{G}|,$$

  where the probability space is defined by random coins consumed by $\mathsf{Sample}_{\mathbb{G}}$.

- $\mathsf{Sample}^{-1}_{\mathbb{G}}(G) \rightarrow r$. The public re-sampler takes an element $G \in \mathbb{G}$ as input and outputs random coins r with probability $1-\theta$ such that $\mathsf{Sample}_{\mathbb{G}}(;r)=G$ or outputs a failure symbol $\perp$ with probability θ. In particular, we require that, for all $r' \in R_G$ where $R_G := \{\widehat{r} : \mathsf{Sample}_{\mathbb{G}}(;\widehat{r}) = G\}$,

  $$\Pr\left[r=r' | r \neq \perp\right] = 1/|R_G|,$$

  where *the probability space is defined by random coins consumed by* $\mathsf{Sample}^{-1}_{\mathbb{G}}$.

Bellare, Waters, and Yilek [BWY11] had realized these two algorithms for bilinear group $\mathbb{G}$ with $\zeta \approx 1/2^\rho$ and $\theta \approx 1/2^\rho$ where ρ is an independent parameter, based on the technique for hashing into elliptic curve groups [BLS01]. We will invoke $\mathsf{Sample}_{\mathbb{G}}$ and $\mathsf{Sample}^{-1}_{\mathbb{G}}$ as black box, their detailed specifications can be found in [BWY11].

$$
\begin{array}{ll}
\underline{\mathsf{Setup}(1^\lambda, n)} & \\
\mathcal{G} := (N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{GrpGen}(1^\lambda) & \mathsf{IdSp} = \{0,1\}^n,\ \mathsf{MsgSp} = \{0,1\} \\
\textbf{for } i \in [5] \textbf{ do } g_i \leftarrow \mathbb{G}_{p_i} & \\
v_5 \leftarrow \mathbb{Z}_N;\ G := g_1 g_5^{v_5};\ \widehat{G} := g_1 g_2 g_3 & \underline{\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \textsc{id})} \\
u, u_5 \leftarrow \mathbb{Z}_N;\ \mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n} & r, r_4, r_4' \leftarrow \mathbb{Z}_N \\
U := g_1^u g_5^{u_5};\ \mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}; & K := \widehat{G}^r g_4^{r_4} \\
\widehat{U} := (g_1 g_2 g_3)^u;\ \widehat{\mathbf{W}} := (g_1 g_2 g_3)^{\mathbf{w}} & K' := \left(\widehat{U} \prod_{i=1}^n \widehat{\mathbf{W}}[2i - \textsc{id}[i]]\right)^r g_4^{r_4'} \\
\textsc{mpk} := \left(\mathcal{G}, G, U, \mathbf{W}, g_5\right) & \textbf{return } (K, K') \\
\textsc{msk} := \left(\widehat{G}, \widehat{U}, \widehat{\mathbf{W}}, g_4\right) & \\
\textbf{return } \textsc{mpk}, \textsc{msk} & \underline{\mathsf{Enc}(\textsc{mpk}, \textsc{id}, \textsc{m})} \\
 & \textbf{if } \textsc{m} = 0 \textbf{ then} \\
\underline{\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \textsc{ct})} & \quad s, s_5, s_5' \leftarrow \mathbb{Z}_N \\
\textbf{if } e(C, K) = e(C', K') \textbf{ then return } 0 & \quad C := \left(U \prod_{i=1}^n \mathbf{W}[2i - \textsc{id}[i]]\right)^s g_5^{s_5} \\
\textbf{else return } 1 & \quad C' := G^s g_5^{s_5'} \\
 & \textbf{else} \\
\underline{\mathsf{OpenToOne}(\textsc{mpk}, \textsc{id}, \textsc{ct})} & \quad C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}() \\
\mathbf{r} \leftarrow \mathsf{Sample}_{\mathbb{G}}^{-1}(C) & \textbf{return } (C, C') \\
\mathbf{r}' \leftarrow \mathsf{Sample}_{\mathbb{G}}^{-1}(C') & \\
\textbf{return } (\mathbf{r}, \mathbf{r}') &
\end{array}
$$

Figure 7: Our Main Construction: IBE with 1SPO

4.2 Construction

Assuming a group generator GrpGen described in previous subsection, our main construction is shown in Figure 7, an IBE scheme with identity space $\mathsf{IdSp} = \{0,1\}^n$ and message space $\mathsf{MsgSp} = \{0,1\}$ which is also equipped with algorithm OpenToOne for achieving One-Sided Publicly Openability.

The correctness of our construction is obvious. Fix an $\textsc{id} \in \mathsf{IdSp}$. For the case $(C, C') \in [\mathsf{Enc}(\textsc{mpk}, \textsc{id}, 0)]$, we have

$$
\begin{aligned}
e(C, K) &= e\left(\left(U \prod_{i=1}^n \mathbf{W}[2i - \textsc{id}[i]]\right)^s g_5^{s_5}, \widehat{G}^r g_4^{r_4}\right) \\
&= e\left(g_1^{(u + \sum_{i=1}^n \mathbf{w}[2i - \textsc{id}[i]])s}, g_1^r\right) = e\left(g_1^s, g_1^{(u + \sum_{i=1}^n \mathbf{w}[2i - \textsc{id}[i]])r}\right) \\
&= e\left(G^s g_5^{s_5'}, \left(\widehat{U} \prod_{i=1}^n \widehat{\mathbf{W}}[2i - \textsc{id}[i]]\right)^r g_4^{r_4'}\right) = e(C', K'),
\end{aligned}
$$

which means $\Pr[\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \mathsf{Enc}(\textsc{mpk}, \textsc{id}, 0)) = 0] = 1$. For the case $(C, C') \in [\mathsf{Enc}(\textsc{mpk}, \textsc{id}, 1)]$, we let $C = g^c$ and $C' = g^{c'}$ where $c, c' \in \mathbb{Z}_N$, and we have, with the probability space defined by sampling $r, r_4, r_4', c, c' \leftarrow \mathbb{Z}_N$,

$$
\begin{aligned}
& \Pr[\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \mathsf{Enc}(\textsc{mpk}, \textsc{id}, 1)) = 0] \\
= & \Pr\left[e(C, K) = e(C', K')\right] \\
= & \Pr\left[e\left(g^c, \widehat{G}^r g_4^{r_4}\right) = e\left(g^{c'}, \left(\widehat{U} \prod_{i=1}^n \widehat{\mathbf{W}}[2i - \textsc{id}[i]]\right)^r g_4^{r_4'}\right)\right] \\
= & \Pr\left[e\left((g_1 g_2 g_3 g_4)^c, (g_1 g_2 g_3)^r g_4^{r_4}\right) = e\left((g_1 g_2 g_3 g_4)^{c'}, (g_1 g_2 g_3)^{(u + \sum \mathbf{w}[2i - \textsc{id}[i]])r} g_4^{r_4'}\right)\right] \\
\leqslant & \ 64/2^{4\lambda}
\end{aligned}
$$

Therefore we conclude that $\Pr[\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \mathsf{Enc}(\textsc{mpk}, \textsc{id}, 1)) = 1] \geqslant 1 - 64/2^{4\lambda}$.

As [BWY11], algorithm OpenToOne indeed satisfies our requirement defined in Section 2 by the property of algorithm $\mathsf{Sample}_{\mathbb{G}}$ and $\mathsf{Sample}_{\mathbb{G}}^{-1}$ shown in previous subsection. We remark that algorithm OpenToOne runs $\mathsf{Sample}_{\mathbb{G}}^{-1}$ twice independently and thus has failure probability $\delta \leqslant 2\theta$.

4.3 Security Analysis

We prove the following theorem in this subsection.

Theorem 2 (Main Theorem) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exist adversaries $\mathscr{B}_1$, $\mathscr{B}_2$, $\mathscr{B}_3$, $\mathscr{B}_4$, $\mathscr{B}_5$ and $\mathscr{B}_6$ such that*

$$\begin{aligned}\mathsf{Adv}_{\mathscr{A}}^{\mathsf{INDmCPA}}(\lambda) \leqslant\ & \mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{SD1}}(\lambda)+2n\cdot\mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{SD2}}(\lambda)+(n+1)\cdot\mathsf{Adv}_{\mathscr{B}_3}^{\mathrm{DDH1}}(\lambda)\\ &+\mathsf{Adv}_{\mathscr{B}_4}^{\mathrm{DDH2}}(\lambda)+\mathsf{Adv}_{\mathscr{B}_5}^{\mathrm{SD3}}(\lambda)+\mathsf{Adv}_{\mathscr{B}_6}^{\mathrm{SD4}}(\lambda)+2\zeta.\end{aligned}$$

and $\max_{i\in[6]}\mathsf{Time}(\mathscr{B}_i)\approx\mathsf{Time}(\mathscr{A})+(Q_C+Q_K)\cdot\mathsf{poly}(\lambda,n)$ *where* poly *is independent of* $\mathscr{A}$.

4.3.1 Organization of Proof

Before we proceed, we define two functions for brevity:

$$h(u,\mathbf{w},\textsc{id}):=u+\sum_{i=1}^{n}\mathbf{w}[2i-\textsc{id}[i]]\ \text{ and }\ H(U,\mathbf{W},\textsc{id}):=U\cdot\prod_{i=1}^{n}\mathbf{W}[2i-\textsc{id}[i]] \tag{1}$$

where $\textsc{id}\in\{0,1\}^n$, $u\in\mathbb{Z}_N$, $\mathbf{w}\in\mathbb{Z}_N^{2n}$, $U\in\mathbb{G}$, and $\mathbf{W}\in\mathbb{G}^{2n}$. When we set $U=g^u$ and $\mathbf{W}=g^{\mathbf{w}}$ for some $g\in\mathbb{G}$, we immediately have the following relation

$$H(U,\mathbf{W},\textsc{id})=g^{h(u,\mathbf{w},\textsc{id})}.$$

We also need a family of random functions $\{\mathsf{R}_i(\cdot)\}_{i\in\{0\}\cup[n]}$ defined as follows:

$$\mathsf{R}_i(\textsc{id}):\textsc{id}|_i\to\mathbb{Z}_N,\ \forall i\in\{0\}\cup[n],$$

where $\textsc{id}|_i$ denotes the i-bit prefix of $\textsc{id}$. We note that the Chinese Reminder Theorem implies that $\mathsf{R}_i(\textsc{id}) \bmod p_2$ and $\mathsf{R}_i(\textsc{id}) \bmod p_3$ are distributed independently, which correspond to $\widehat{\mathsf{R}}_i$ and $\widetilde{\mathsf{R}}_i$, respectively, in [HKS15]. Although the range of random functions is $\mathbb{Z}_N$, the actual working range is $\mathsf{R}_i(\textsc{id}) \bmod p_2$ and $\mathsf{R}_i(\textsc{id}) \bmod p_3$ corresponding to semi-functional space $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$, respectively.

The proof follows hybrid arguments using a series of games, which can be roughly divided into four phases. We describe these games in a phase-by-phase fashion in Figure 8, Figure 9, Figure 10, Figure 11, respectively. In particular, we state that

- In phase 1, we introduce semi-functional components (elements in $\mathbb{G}_{p_2}$) into ciphertexts and introduce random function R_0 into the system.
- In phase 2, we increase the entropy of random function we have introduced following the method of Hofheinz *et al.* [HKS15]. In particular, by executing phase 2 for n times, we replace the initial random function R_0 with R_n, whose output depends on all bits of $\textsc{id}$.
- In phase 3, we handle the multiple occurrences of a single identity in challenge ciphertexts as well as in secret keys. We finally argue that multiple ciphertexts for a single identity are computationally independent in semi-functional space and so do secret keys. The proof will also follows Hofheinz *et al.*'s idea for full security [HKS15].
- In phase 4, we show that all ciphertexts for $\textsc{m}=0$ are computationally indistinguishable from those for $\textsc{m}=1$, which is truly random following the method used in [DCIP10, BWY11].

The remaining of the section includes four parts corresponding to four phases in order. Each part begins with games involved and then proves a series of lemmas showing computational indistinguishability of these games. Putting them together, we immediately obtain the main theorem. In the proof, we define the advantage function of adversary $\mathscr{A}$ in $\mathsf{Game}_{x.x.x}$ as

$$\mathsf{Adv}_{\mathscr{A}}^{x.x.x}(\lambda):=\left|\Pr\left[\mathsf{Game}_{x.x.x,\mathscr{A}}(\lambda)=1\right]-1/2\right|.$$

4.3.2 Phase 1: Prelude

All games used in Phase 1 is defined in Figure 8.

Lemma 5 ($\mathsf{Game}_0\approx\mathsf{Game}_1$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}_{\mathscr{A}}^{0}(\lambda)-\mathsf{Adv}_{\mathscr{A}}^{1}(\lambda)\right|\leqslant\mathsf{Adv}_{\mathscr{B}}^{\mathrm{SD1}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B})\approx\mathsf{Time}(\mathscr{A})+(Q_C+Q_K)\cdot\mathsf{poly}(\lambda,n)$ *where* poly *is independent of* $\mathscr{A}$.

Initialize
$\mathcal{G} := (N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{GrpGen}(1^\lambda)$
for $i \in [5]$ **do** $g_i \leftarrow \mathbb{G}_{p_i}$
$v_5 \leftarrow \mathbb{Z}_N$
$G := g_1 g_5^{v_5}$; $\boxed{\widetilde{G} := (g_1 g_2)\, g_5^{v_5}}$
$\widehat{G} := g_1 g_2 g_3$
$u, u_5 \leftarrow \mathbb{Z}_N$; $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$
$U := g_1^u g_5^{u_5}$; $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$
$\boxed{\widetilde{U} := (g_1 g_2)^u\, g_5^{u_5}; \widetilde{\mathbf{W}} := (g_1 g_2)^{\mathbf{w}}\, g_5^{\mathbf{w}_5}}$
$\widehat{U} := (g_1 g_2 g_3)^u$; $\widehat{\mathbf{W}} := (g_1 g_2 g_3)^{\mathbf{w}}$
$\beta \leftarrow \{0,1\}$
return $(\mathcal{G}, G, U, \mathbf{W}, g_5)$

Extract(ID)
if ID $\in$ ChID **then return** $\perp$
ExID := ExID $\cup$ {ID}
$r, r_4, r_4' \leftarrow \mathbb{Z}_N$
$K := \widehat{G}^r g_4^{r_4}$
$K' := H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$
$K' := (g_2 g_3)^{\mathsf{R}_0(\text{ID}) \cdot r} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$
return (K, K')

Game$_0$, $\boxed{\text{Game}_1}$, Game$_{2.0}$
$\mathsf{R}_0 : \mathsf{IdSp} \rightarrow \mathbb{Z}_N$

Challenge(ID*, M$_0^*$, M$_1^*$)
if ID* $\in$ ExID **then return** $\perp$
ChID := ChID $\cup$ {ID*}
if M$_\beta^* = 0$ **then**
$s, s_5, s_5' \leftarrow \mathbb{Z}_N$
$C := H(U, \mathbf{W}, \text{ID})^s g_5^{s_5}$
$\boxed{C := H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID})^s g_5^{s_5}}$
$C := g_2^{\mathsf{R}_0(\text{ID}^*) \cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID})^s g_5^{s_5}$
$C' := G^s g_5^{s_5'}$
$\boxed{C' := \widetilde{G}^s g_5^{s_5'}}$
else
$C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$
return (C, C')

Finalize(β')
return ($\beta = \beta'$)

Figure 8: Game$_0$, Game$_1$, Game$_{2.0}$

Proof. Given $(\mathcal{G}, g_1, g_4, g_5, X_1X_2X_3, T)$ where T is a random element of either $\mathbb{G}_{p_1}$ or $\mathbb{G}_{p_1p_2}$, adversary $\mathcal{B}$ simulates the procedures as follows. We assume $g_2 \leftarrow \mathbb{G}_{p_2}$, $g_3 \leftarrow \mathbb{G}_{p_3}$ and implicitly parse $X_1X_2X_3 = (g_1g_2g_3)^x$ for some $x \in \mathbb{Z}_N$, and either $T = g_1^t$ or $T = (g_1g_2)^t$ from some $t \in \mathbb{Z}_N$.

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 \cdot g_5^{v_5}$. We implicitly define $\widetilde{G} := (g_1g_2) \cdot g_5^{v_5}$ and $\widehat{G} := g_1g_2g_3$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$, $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$ and set $U := g_1^u g_5^{u_5}$, $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$. We implicitly define $\widetilde{U} := (g_1g_2)^u g_5^{u_5}$, $\widetilde{\mathbf{W}} := (g_1g_2)^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widehat{U} := (g_1g_2g_3)^u$, $\widehat{\mathbf{W}} := (g_1g_2g_3)^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output MPK := $(\mathcal{G}, G, U, \mathbf{W}, g_5)$.

Extract(ID) Return $\perp$ when ID $\in$ ChID. Update ExID := ExID $\cup$ {ID}. Sample $r', r_4, r_4' \leftarrow \mathbb{Z}_N$ and set

$$K := (X_1X_2X_3)^{r'} \cdot g_4^{r_4} \quad \text{and} \quad K' := (X_1X_2X_3)^{h(u,\mathbf{w},\text{ID}) \cdot r'} \cdot g_4^{r_4'}.$$

Output (K, K'). Here we implicitly set $r := xr' \in \mathbb{Z}_N$.

Challenge(ID*, M$_0^*$, M$_1^*$) Return $\perp$ when ID* $\in$ ExID. Update ChID := ChID $\cup$ {ID*}. If M$_\beta^* = 0$, sample $s', s_5'', s_5''' \leftarrow \mathbb{Z}_N$ and set

$$C := T^{h(u,\mathbf{w},\text{ID}) \cdot s'} \cdot g_5^{s_5''} \quad \text{and} \quad C' := T^{s'} \cdot g_5^{s_5'''}.$$

If M$_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C'). Here we implicitly set $s \in \mathbb{Z}_N$ such that $s = ts' \bmod p_1p_2$.

Finalize(β') Output ($\beta = \beta'$).

The algorithm $\mathcal{B}$ outputs 1 if and only if **Finalize**(β') = 1, i.e., the adversary $\mathcal{A}$ wins the game.

Observe that if $T = g_1^t \in \mathbb{G}_{p_1}$, the simulation described above is identical to Game$_0$; if $T = (g_1g_2)^t \in \mathbb{G}_{p_1p_2}$, the simulation is identical to Game$_1$. Therefore we can conclude that $\left|\mathsf{Adv}_{\mathcal{A}}^0(\lambda) - \mathsf{Adv}_{\mathcal{A}}^1(\lambda)\right| \leqslant \mathsf{Adv}_{\mathcal{B}}^{\mathrm{SD1}}(\lambda)$. □

Lemma 6 (Game$_1$ = Game$_{2.0}$) *For any adversary $\mathcal{A}$, we have* $\mathsf{Adv}_{\mathcal{A}}^1(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{2.0}(\lambda)$.

Proof. Since random function $\mathsf{R}_0(\cdot)$ is consistent on all possible identities in IdSp, R_0(ID) for all ID $\in$ IdSp is a single random variable independently distributed over $\mathbb{Z}_N$. If we sample $u' \leftarrow \mathbb{Z}_N$ and set, in **Initialize** procedure in Game$_1$,

$$U := g_1^{u'} g_5^{u_5} \quad \text{and} \quad \widetilde{U} := (g_1g_2)^{u'}\, g_2^{\mathsf{R}_0} g_5^{u_5} \quad \text{and} \quad \widehat{U} := (g_1g_2g_3)^{u'}\, (g_2g_3)^{\mathsf{R}_0},$$

the resulting game remains unchanged, since we in fact implicitly define

$$u = u' \bmod p_1 \;\text{ and }\; u = u' + \mathsf{R}_0 \bmod p_2p_3,$$

which is still distributed over $\mathbb{Z}_N$ as we required in Game_1. Observe that the simulation is also identical to $\mathsf{Game}_{2.0}$ with $u = u'$ except the definition of $\widetilde{U}$ and $\widehat{U}$ in **Initialize** procedure. However we note that the difference is just conceptual as both $\widetilde{U}$ and $\widehat{U}$ are not given to adversary in MPK. Therefore we can conclude that Game_1 and $\mathsf{Game}_{2.0}$ are statistically identical. $\square$

4.3.3 Phase 2: From R_0 to R_n

All games used in Phase 2 is defined in Figure 9.

Initialize
$\mathcal{G} := (N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{GrpGen}(1^\lambda, 5)$
for $i \in [5]$ **do** $g_i \leftarrow \mathbb{G}_{p_i}$
$v_5 \leftarrow \mathbb{Z}_N$; $G := g_1 g_5^{v_5}$
$\widetilde{G} := (g_1g_2)\, g_5^{v_5}$; $\boxed{\overline{G} := (g_1g_3)\, g_5^{v_5}}$
$\widehat{G} := g_1g_2g_3$
$u, u_5 \leftarrow \mathbb{Z}_N$; $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$
$U := g_1^u g_5^{u_5}$; $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$
$\widetilde{U} := (g_1g_2)^u g_5^{u_5}$; $\widetilde{\mathbf{W}} := (g_1g_2)^{\mathbf{w}} g_5^{\mathbf{w}_5}$
$\boxed{\overline{U} := (g_1g_3)^u g_5^{u_5};\ \overline{\mathbf{W}} := (g_1g_3)^{\mathbf{w}} g_5^{\mathbf{w}_5}}$
$\widehat{U} := (g_1g_2g_3)^u$; $\widehat{\mathbf{W}} := (g_1g_2g_3)^{\mathbf{w}}$
$\beta \leftarrow \{0,1\}$
return $(\mathcal{G}, G, U, \mathbf{W}, g_5)$

Extract(ID)
if ID $\in$ ChID **then return** $\perp$
ExID := ExID $\cup$ {ID}
$r, r_4, r_4' \leftarrow \mathbb{Z}_N$
$K := \widehat{G}^r g_4^{r_4}$
$K' := (g_2g_3)^{\mathsf{R}_i(\text{ID})\cdot r} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$
$K' := (g_2g_3)^{\mathsf{R}_{i+1}(\text{ID})\cdot r} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$
return (K, K')

Finalize(β')
return $(\beta = \beta')$

$\mathsf{Game}_{2.i}$, $\boxed{\mathsf{Game}_{2.i.1}}$, $\mathsf{Game}_{2.i.2}$
$\mathsf{R}_i : \mathsf{IdSp} \rightarrow \mathbb{Z}_N$

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$)
if $\text{ID}^* \in$ ExID **then return** $\perp$
ChID := ChID $\cup$ {ID*}
if $\text{M}_\beta^* = 0$ **and** $\text{ID}^*[i+1] = 0$ **then**
  $s, s_5, s_5' \leftarrow \mathbb{Z}_N$;
  $C := g_2^{\mathsf{R}_i(\text{ID}^*)\cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID}^*)^s g_5^{s_5}$
  $C := g_2^{\mathsf{R}_{i+1}(\text{ID}^*)\cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID}^*)^s g_5^{s_5}$
  $C' = \widetilde{G}^s g_5^{s_5'}$
elif $\text{M}_\beta^* = 0$ **and** $\text{ID}^*[i+1] = 1$ **then**
  $s, s_5, s_5' \leftarrow \mathbb{Z}_N$
  $C := g_2^{\mathsf{R}_i(\text{ID}^*)\cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID}^*)^s g_5^{s_5}$
  $\boxed{C := g_3^{\mathsf{R}_i(\text{ID}^*)\cdot s} \cdot H(\overline{U}, \overline{\mathbf{W}}, \text{ID}^*)^s g_5^{s_5}}$
  $C := g_3^{\mathsf{R}_{i+1}(\text{ID}^*)\cdot s} \cdot H(\overline{U}, \overline{\mathbf{W}}, \text{ID}^*)^s g_5^{s_5}$
  $C' := \widetilde{G}^s g_5^{s_5'}$
  $\boxed{C' := \overline{G}^s g_5^{s_5'}}$
else
  $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$
return (C, C')

Figure 9: $\mathsf{Game}_{2.i}$ $(i \in [0,n])$, $\mathsf{Game}_{2.i.1}$ $(i \in [0,n-1])$, $\mathsf{Game}_{2.i.2}$ $(i \in [0,n-1])$

Lemma 7 ($\mathsf{Game}_{2.i} \approx \mathsf{Game}_{2.i.1}$) *For any probabilistic polynomial time adversary $\mathcal{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathcal{B}$ such that*

$$\left|\mathsf{Adv}_{\mathcal{A}}^{2.i}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{2.i.1}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathcal{B}}^{\mathrm{SD2}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + (Q_C + Q_K) \cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of $\mathcal{A}$.*

Proof. Given $(\mathcal{G}, g_1, g_4, g_5, X_2X_5, Y_2Y_3, T)$ where T is a random element of either $\mathbb{G}_{p_2p_5}$ or $\mathbb{G}_{p_3p_5}$, adversary $\mathcal{B}$ simulates the procedures as follows. We assume $g_2 \leftarrow \mathbb{G}_{p_2}$, $g_3 \leftarrow \mathbb{G}_{p_3}$ and implicitly parse $X_2X_5 = (g_2g_5)^x$ for some $x \in \mathbb{Z}_N$, $Y_2Y_3 = (g_2g_3)^y$ for some $y \in \mathbb{Z}_N$ and either $T = (g_2g_5)^t$ or $T = (g_3g_5)^t$ for some $t \in \mathbb{Z}_N$.

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 g_5^{v_5}$. We implicitly define $\widetilde{G} := (g_1g_2)\, g_5^{v_5}$, $\overline{G} := (g_1g_3)\, g_5^{v_5}$ and $\widehat{G} := g_1g_2g_3$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$ and $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$, and set $U := g_1^u g_5^{u_5}$ and $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$. We implicitly define $\widetilde{U} := (g_1g_2)^u g_5^{u_5}$, $\widetilde{\mathbf{W}} := (g_1g_2)^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\overline{U} := (g_1g_3)^u g_5^{u_5}$, $\overline{\mathbf{W}} := (g_1g_3)^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widehat{U} := (g_1g_2g_3)^u$ and $\widehat{\mathbf{W}} := (g_1g_2g_3)^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output MPK $:= (\mathcal{G}, G, U, \mathbf{W}, g_5)$. $\mathcal{B}$ also maintains a random function $\mathsf{R}_i(\cdot)$ in an on-the-fly manner.

Extract(ID) Return $\perp$ when $\text{ID} \in \mathsf{ChID}$. Update $\mathsf{ExID} := \mathsf{ExID} \cup \{\text{ID}\}$. Sample $r', r'', r_4, r_4' \leftarrow \mathbb{Z}_N$ and set

$$K := g_1^{r'} \cdot (Y_2 Y_3)^{r''} \cdot g_4^{r_4} \text{ and } K' := g_1^{h(u,\mathbf{w},\text{ID})\cdot r'} \cdot (Y_2 Y_3)^{(\mathsf{R}_i(\text{ID})+h(u,\mathbf{w},\text{ID}))\cdot r''} \cdot g_4^{r_4'}.$$

Output (K, K'). Here we implicitly set $r \in \mathbb{Z}_N$ such that $r = r' \bmod p_1$ and $r = yr'' \bmod p_2 p_3$.

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$) Return $\perp$ when $\text{ID}^* \in \mathsf{ExID}$. Update $\mathsf{ChID} := \mathsf{ChID} \cup \{\text{ID}^*\}$. If $\text{M}_\beta^* = 0$ and $\text{ID}^*[i+1] = 0$, sample $s', s'', s_5'', s_5''' \leftarrow \mathbb{Z}_N$ and set

$$C := g_1^{h(u,\mathbf{w},\text{ID}^*)\cdot s'} \cdot (X_2 X_5)^{(\mathsf{R}_i(\text{ID}^*)+h(u,\mathbf{w},\text{ID}^*))\cdot s''} \cdot g_5^{s_5''} \text{ and } C' := g_1^{s'} \cdot (X_2 X_5)^{s''} \cdot g_5^{s_5'''}.$$

Here we implicitly set $s \in \mathbb{Z}_N$ such that $s = s' \bmod p_1$ and $s = xs'' \bmod p_2$. If $\text{M}_\beta^* = 0$ and $\text{ID}^*[i+1] = 1$, sample $s', s'', s_5'', s_5''' \leftarrow \mathbb{Z}_N$ and set

$$C := g_1^{h(u,\mathbf{w},\text{ID}^*)\cdot s'} \cdot T^{(\mathsf{R}_i(\text{ID}^*)+h(u,\mathbf{w},\text{ID}^*))\cdot s''} \cdot g_5^{s_5''} \text{ and } C' := g_1^{s'} \cdot T^{s''} \cdot g_5^{s_5'''}.$$

Here we implicitly set $s \in \mathbb{Z}_N$ such that $s = s' \bmod p_1$, and $s = ts'' \bmod p_2 p_3$. Finally, if $\text{M}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathscr{B}$ outputs 1 if and only if **Finalize**$(\beta') = 1$, i.e., the adversary $\mathscr{A}$ wins the game.

Observe that if $T = (g_2 g_5)^t \in \mathbb{G}_{p_2 p_5}$, the simulation described above is identical to $\mathsf{Game}_{2.i}$; if $T = (g_3 g_5)^t$, the simulation is identical to $\mathsf{Game}_{2.i.1}$. Therefore we can conclude that $\left|\mathsf{Adv}_{\mathscr{A}}^{2.i}(\lambda) - \mathsf{Adv}_{\mathscr{A}}^{2.i.1}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{SD2}}(\lambda)$. □

Lemma 8 ($\mathsf{Game}_{2.i.1} \approx \mathsf{Game}_{2.i.2}$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}_{\mathscr{A}}^{2.i.1}(\lambda) - \mathsf{Adv}_{\mathscr{A}}^{2.i.2}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{DDH1}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (Q_C + Q_K) \cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. Given $\left(\mathscr{G}, g_1, g_2, g_3, g_4, g_5, g_2^x \widetilde{X}_4, g_2^y \widetilde{Y}_4, g_3^x \overline{X}_4, g_3^y \overline{Y}_4, T\right)$ where T is either $\left(g_2^{xy}, g_3^{xy}\right)$ or $\left(g_2^{xy+z}, g_3^{xy+z}\right)$, adversary $\mathscr{B}$ generates Q_K tuples (using *Many Tuple Lemma* [CW13])

$$\left(g_2^{x_j} \widetilde{X}_{4,j}, g_3^{x_j} \overline{X}_{4,j}, T_j\right), \ \forall j \in [Q_K]$$

where $\widetilde{X}_{4,j}, \overline{X}_{4,j} \leftarrow \mathbb{G}_{p_4}$ and T_j is either $\left(g_2^{x_j y}, g_3^{x_j y}\right)$ or $\left(g_2^{x_j y + z_j}, g_3^{x_j y + z_j}\right)$ for $x_j, z_j \leftarrow \mathbb{Z}_N$, and then simulates the procedures as follows.

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 g_5^{v_5}$, $\widetilde{G} := (g_1 g_2)\, g_5^{v_5}$, $\overline{G} := (g_1 g_3)\, g_5^{v_5}$ and $\widehat{G} := g_1 g_2 g_3$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$ and set $U := g_1^u g_5^{u_5}$, $\widetilde{U} := (g_1 g_2)^u g_5^{u_5}$, $\overline{U} := (g_1 g_3)^u g_5^{u_5}$ and $\widehat{U} := (g_1 g_2 g_3)^u$. Sample $\mathbf{w}', \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$ and set $\mathbf{W} := g_1^{\mathbf{w}'} g_5^{\mathbf{w}_5}$. We implicitly define

$$\widetilde{\mathbf{W}} := (g_1 g_2)^{\mathbf{w}'} g_2^{y\mathbf{e}_{2(i+1)-1}} g_5^{\mathbf{w}_5} \text{ and } \overline{\mathbf{W}} := (g_1 g_3)^{\mathbf{w}'} g_3^{y\mathbf{e}_{2(i+1)-0}} g_5^{\mathbf{w}_5} \text{ and } \widehat{\mathbf{W}} := (g_1 g_2 g_3)^{\mathbf{w}'} g_2^{y\mathbf{e}_{2(i+1)-1}} g_3^{y\mathbf{e}_{2(i+1)-0}}.$$

Randomly pick $\beta \leftarrow \{0,1\}$ and output $\text{MPK} := (\mathscr{G}, G, U, \mathbf{W}, g_5)$. Note that we implicitly set $\mathbf{w} \in \mathbb{Z}_N^{2n}$ such that

$$\mathbf{w} = \mathbf{w}' \bmod p_1 \text{ and } \mathbf{w} = \mathbf{w}' + y\mathbf{e}_{2(i+1)-1} \bmod p_2 \text{ and } \mathbf{w} = \mathbf{w}' + y\mathbf{e}_{2(i+1)-0} \bmod p_3,$$

and note that neither $\widetilde{\mathbf{W}}[2(i+1)-1]$ nor $\overline{\mathbf{W}}[2(i+1)-0]$ is known to $\mathscr{B}$. Besides that, $\mathscr{B}$ also maintains a random function $\mathsf{R}_i(\cdot)$ in an on-the-fly manner.

Extract(ID) Return $\perp$ when $\text{ID} \in \mathsf{ChID}$. Update $\mathsf{ExID} := \mathsf{ExID} \cup \{\text{ID}\}$. If $\mathsf{R}_i(\text{ID})$ has not been used before, sample $r', r_4'', r_4''' \leftarrow \mathbb{Z}_N$ and set

$$\begin{aligned} K &:= (g_1 g_2 g_3)^{r'} \cdot \left(g_2^{x_j} \widetilde{X}_{4,j}\right) \cdot \left(g_3^{x_j} \overline{X}_{4,j}\right) \cdot g_4^{r_4''} \\ K' &:= \begin{cases} g_1^{-\mathsf{R}_i(\text{ID})\cdot r'} \cdot K^{\mathsf{R}_i(\text{ID})+h(u,\mathbf{w}',\text{ID})} \cdot \left(g_3^y \overline{Y}_4\right)^{r'} \cdot T_j[2] \cdot g_4^{r_4'''}, & \text{if } \text{ID}[i+1] = 0 \\ g_1^{-\mathsf{R}_i(\text{ID})\cdot r'} \cdot K^{\mathsf{R}_i(\text{ID})+h(u,\mathbf{w}',\text{ID})} \cdot \left(g_2^y \widetilde{Y}_4\right)^{r'} \cdot T_j[1] \cdot g_4^{r_4'''}, & \text{if } \text{ID}[i+1] = 1 \end{cases} \end{aligned}$$

where $T_j[1]$ and $T_j[2]$ refers to the first and second entry of T_j, respectively. Here we implicitly set $r \in \mathbb{Z}_N$ such that $r = r' \bmod p_1$ and $r = r' + x_j \bmod p_2p_3$. Output (K, K'). On the other hand, if $\mathsf{R}_i(\textsc{id})$ has been touched, we find out the index j and random coins r', r_4'', r_4''' used at the first time $\mathsf{R}_i(\textsc{id})$ was met, and create (K, K') using the old index and these random coins but with the new $\textsc{id}$ following the above method. Then we sample $r'' \leftarrow \mathbb{Z}_N$ and output $\left(K^{r''}, (K')^{r''}\right)$ as reply to the query. In this case, we implicitly set r such that $r = r'r'' \bmod p_1$ and $r = \left(r' + x_j\right) r'' \bmod p_2p_3$.

Challenge($\textsc{id}^*, \textsc{m}_0^*, \textsc{m}_1^*$) Return $\perp$ when $\textsc{id}^* \in \mathsf{ExID}$. Update $\mathsf{ChID} := \mathsf{ChID} \cup \{\textsc{id}^*\}$. If $\textsc{m}_\beta^* = 0$ and $\textsc{id}^*[i+1] = 0$, sample $s, s_5, s_5' \leftarrow \mathbb{Z}_N$ and set

$$C := g_2^{\mathsf{R}_i(\textsc{id}^*)\cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \textsc{id}^*)^s \cdot g_5^{s_5} \text{ and } C' := \widetilde{G}^s \cdot g_5^{s_5'}.$$

Note that we do not need $\widetilde{\mathbf{W}}[2(i+1)-1]$ which is unknown. If $\textsc{m}_\beta^* = 0$ and $\textsc{id}^*[i+1] = 1$, sample $s, s_5, s_5' \leftarrow \mathbb{Z}_N$ and set

$$C := g_3^{\mathsf{R}_i(\textsc{id}^*)\cdot s} \cdot H(\overline{U}, \overline{\mathbf{W}}, \textsc{id}^*)^s \cdot g_5^{s_5} \text{ and } C' := \widetilde{G}^s \cdot g_5^{s_5'}.$$

Note that we do not need $\overline{\mathbf{W}}[2(i+1)-0]$ which is unknown. In a word, even though $\mathscr{B}$ does not know all elements in $\widetilde{\mathbf{W}}$ and $\overline{\mathbf{W}}$, it still can compute ciphertext for $\textsc{m}_\beta^* = 0$ as usual. Finally, if $\textsc{m}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathscr{B}$ outputs 1 if and only if **Finalize**(β') $= 1$, i.e., the adversary $\mathscr{A}$ wins.

Clearly, if $T_j = \left(g_2^{x_j y}, g_3^{x_j y}\right)$ for all $j \in [q]$, the simulation described above is identical to $\mathsf{Game}_{2.i.1}$. On the other hand, if $T_j = \left(g_2^{x_j y + z_j}, g_3^{x_j y + z_j}\right)$ for all $j \in [q]$, we implicitly set

$$\mathsf{R}_{i+1}(\textsc{id}) := \begin{cases} \mathsf{R}_i(\textsc{id}) \bmod p_2, & \text{if } \textsc{id}[i+1] = 0 \\ \mathsf{R}_i(\textsc{id}) + z_j/(r' + x_j) \bmod p_2, & \text{if } \textsc{id}[i+1] = 1 \end{cases} \quad \text{and}$$

$$\mathsf{R}_{i+1}(\textsc{id}) := \begin{cases} \mathsf{R}_i(\textsc{id}) + z_j/(r' + x_j) \bmod p_3, & \text{if } \textsc{id}[i+1] = 0 \\ \mathsf{R}_i(\textsc{id}) \bmod p_3, & \text{if } \textsc{id}[i+1] = 1 \end{cases}$$

where index $j \in [q]$ and index $r' \in \mathbb{Z}_N$ were selected at the first time $\mathsf{R}_i(\textsc{id})$ was met in the simulation. Therefore the simulation in this case is identical to $\mathsf{Game}_{2.i.2}$. Therefore we can conclude that $\left|\mathsf{Adv}_{\mathscr{A}}^{2.i.1}(\lambda) - \mathsf{Adv}_{\mathscr{A}}^{2.i.2}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{DDH1}}(\lambda)$. ☐

Lemma 9 ($\mathsf{Game}_{2.i.2} \approx \mathsf{Game}_{2.(i+1)}$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}_{\mathscr{A}}^{2.i.2}(\lambda) - \mathsf{Adv}_{\mathscr{A}}^{2.(i+1)}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{SD2}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (Q_C + Q_K) \cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. The proof is almost the same as the proof for Lemma 7. The only difference is to employ the high-entropy random function R_{i+1} instead of the low-entroy R_i in the simulation. ☐

4.3.4 Phase 3: Handling multi-ciphertexts, multi-keys setting

All games used in Phase 3 is defined in Figure 10.

Lemma 10 ($\mathsf{Game}_{2.n} \approx \mathsf{Game}_3$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}_{\mathscr{A}}^{2.n}(\lambda) - \mathsf{Adv}_{\mathscr{A}}^{3}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{DDH2}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (Q_C + Q_K) \cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. Given $\left(\mathscr{G}, g_1, g_2, g_3, g_4, g_5, g_2^x\widetilde{X}_5, g_2^y\widetilde{Y}_5, T\right)$ where T is either g_2^{xy} or g_2^{xy+z}, adversary $\mathscr{B}$ generates Q_C tuples (using *Many Tuple Lemma* [CW13])

$$\left(g_2^{x_j}\widetilde{X}_{5,j}, T_j\right), \ \forall j \in [Q_C]$$

where $\widetilde{X}_{5,j} \leftarrow \mathbb{G}_{p_5}$ and T_j is either $g_2^{x_j y}$ or $g_2^{x_j y + z_j}$ for $x_j, z_j \leftarrow \mathbb{Z}_N$, and then simulates the procedures as follows.

Initialize

$\mathcal{G} := (N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{GrpGen}(1^\lambda)$
for $i \in [5]$ **do** $g_i \leftarrow \mathbb{G}_{p_i}$
$v_5 \leftarrow \mathbb{Z}_N$; $G := g_1 g_5^{v_5}$
$\widetilde{G} := (g_1 g_2)\, g_5^{v_5}$; $\widetilde{G} := \perp$
$\widehat{G} := g_1 g_2 g_3$; $\widehat{G} := g_1$
$u, u_5 \leftarrow \mathbb{Z}_N$; $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$
$U := g_1^u g_5^{u_5}$; $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$
$\widetilde{U} := (g_1 g_2)^u\, g_5^{u_5}$; $\widetilde{\mathbf{W}} := (g_1 g_2)^{\mathbf{w}}\, g_5^{\mathbf{w}_5}$
$\widetilde{U} := \perp$; $\widetilde{\mathbf{W}} := \perp$
$\widehat{U} := (g_1 g_2 g_3)^u$; $\widehat{\mathbf{W}} := (g_1 g_2 g_3)^{\mathbf{w}}$
$\widehat{U} := g_1^u$; $\widehat{\mathbf{W}} := g_1^{\mathbf{w}}$
$\beta \leftarrow \{0,1\}$
return $(\mathcal{G}, G, U, \mathbf{W}, g_5)$

Extract(ID)

if ID $\in$ ChID **then return** $\perp$
ExID := ExID $\cup$ {ID}
$r, r_4, r_4' \leftarrow \mathbb{Z}_N$; $r' \leftarrow \mathbb{Z}_N$; $r'' \leftarrow \mathbb{Z}_N$
$K := \widehat{G}^r g_4^{r_4}$
$K := (g_2 g_3)^{r''} \cdot \widehat{G}^r g_4^{r_4}$
$K' := (g_2 g_3)^{\mathsf{R}_n(\text{ID})\cdot r} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$
$K' := (g_2 g_3)^{r'} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$

$\mathsf{Game}_{2.n}$, $\boxed{\mathsf{Game}_3}$, Game_4, Game_5
$\mathsf{R}_n : \mathsf{IdSp} \to \mathbb{Z}_N$

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$)

if $\text{ID}^* \in$ ExID **then return** $\perp$
ChID := ChID $\cup$ {ID*}
if $\text{M}_\beta^* = 0$ **then**
$s, s_5, s_5' \leftarrow \mathbb{Z}_N$
$\boxed{s' \leftarrow \mathbb{Z}_N;}$ $s'' \leftarrow \mathbb{Z}_N$
$C := g_2^{\mathsf{R}_n(\text{ID}^*)\cdot s} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID})^s g_5^{s_5}$
$\boxed{C := g_2^{s'} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \text{ID})^s g_5^{s_5}}$
$C := g_2^{s'} \cdot H(U, \mathbf{W}, \text{ID})^s g_5^{s_5}$
$C' := \widetilde{G}^s g_5^{s_5'}$
$C' := g_2^{s''} \cdot G^s g_5^{s_5'}$
else
$C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$
return (C, C')

Finalize(β')

return $(\beta = \beta')$

Figure 10: $\mathsf{Game}_{2.n}$, Game_3, Game_4, Game_5

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 g_5^{v_5}$, $\widetilde{G} := (g_1 g_2)\, g_5^{v_5}$ and $\widehat{G} := g_1 g_2 g_3$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$ and $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$, and set $U := g_1^u g_5^{u_5}$, $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widetilde{U} := (g_1 g_2)^u g_5^{u_5}$, $\widetilde{\mathbf{W}} := (g_1 g_2)^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widehat{U} := (g_1 g_2 g_3)^u$ and $\widehat{\mathbf{W}} := (g_1 g_2 g_3)^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output MPK $:= (\mathcal{G}, G, U, \mathbf{W}, g_5)$. Besides that, $\mathcal{B}$ also maintains a random function $\mathsf{R}_n(\cdot)$ in an on-the-fly manner.

Extract(ID) Return $\perp$ when ID $\in$ ChID. Update ExID := ExID $\cup$ {ID}. Sample $r, r_4, r_4' \leftarrow \mathbb{Z}_N$ and set

$$K := \widehat{G}^r g_4^{r_4} \quad \text{and} \quad K' := (g_2 g_3)^{\mathsf{R}_n(\text{ID})\cdot r} \cdot H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r g_4^{r_4'}$$

Output (K, K').

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$) Return $\perp$ when $\text{ID}^* \in$ ExID. Update ChID := ChID $\cup$ {ID*}. We maintain another independent random function $\mathsf{R}' : \mathsf{IdSp} \to \mathbb{Z}_N$ whose output depends on all bits of ID. If $\text{M}_\beta^* = 0$, pick a new $j \in [q]$, sample $s', s_5'', s_5''' \leftarrow \mathbb{Z}_N$ and set

$$C := g_1^{h(u,\mathbf{w},\text{ID}^*)s'} \cdot \left(g_2^{x_j} \widetilde{X}_{5,j}\right)^{h(u,\mathbf{w},\text{ID}^*)} \cdot T_j^{\mathsf{R}'(\text{ID}^*)} \cdot g_5^{s_5''} \quad \text{and} \quad C' := g_1^{s'} \cdot \left(g_2^{x_j} \widetilde{X}_{5,j}\right) \cdot g_5^{s_5'''}.$$

Here we implicitly set $s \in \mathbb{Z}_N$ such that $s = s' \bmod p_1$, $s = x_j \bmod p_2$ and $\mathsf{R}_n(\text{ID}^*) = y \cdot \mathsf{R}'(\text{ID}^*) \bmod p_2$ for $\text{ID}^* \in$ ChID. We note that the assignment for R_n is always consistent since the simulation ensures that ExID $\cap$ ChID $= \emptyset$. If $\text{M}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathcal{B}$ outputs 1 if and only if **Finalize**$(\beta') = 1$, i.e., the adversary $\mathcal{A}$ wins the game.

Observe that if $T_j = g_2^{x_j y}$ for all $j \in [q]$, the simulation described above is identical to $\mathsf{Game}_{2.n}$. On the other hand, if $T_j = g_2^{x_j y + z_j}$ for all $j \in [q]$, the simulation is identical to Game_3 where we implicitly set $s' = \mathsf{R}'(\text{ID}) \cdot (x_j y + z_j)$. Therefore we can conclude that $\left|\mathsf{Adv}_{\mathcal{A}}^{2.n}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{3}(\lambda)\right| \leqslant \mathsf{Adv}_{\mathcal{B}}^{\mathrm{DDH2}}(\lambda)$. $\square$

Lemma 11 ($\mathsf{Game}_3 \approx \mathsf{Game}_4$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}^3_{\mathscr{A}}(\lambda) - \mathsf{Adv}^4_{\mathscr{A}}(\lambda)\right| \leqslant \mathsf{Adv}^{\mathrm{DDH1}}_{\mathscr{B}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (Q_C + Q_K)\cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. Given $\left(\mathscr{G}, g_1, g_2, g_3, g_4, g_5, g_2^x\widetilde{X}_4, g_2^y\widetilde{Y}_4, g_3^x\overline{X}_4, g_3^y\overline{Y}_4, T\right)$ where T is either $\left(g_2^{xy}, g_3^{xy}\right)$ or $\left(g_2^{xy+z}, g_3^{xy+z}\right)$, adversary $\mathscr{B}$ generates a shared term and Q_K tuples (using *Many Tuple Lemma* [CW13])

$$(g_2g_3)^y\, Y_4 \text{ and } \left((g_2g_3)^{x_j} X_{4,j}, T_j\right),\ \forall j \in [Q_K]$$

where $Y_4, X_{4,j} \leftarrow \mathbb{G}_{p_4}$ and T_j is either $(g_2g_3)^{x_j y}$ or $(g_2g_3)^{x_j y + z_j}$ for $x_j, z_j \leftarrow \mathbb{Z}_N$, and then simulates the procedures as follows.

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 g_5^{v_5}$, $\widetilde{G} := (g_1g_2)\, g_5^{v_5}$ and $\widehat{G} := g_1g_2g_3$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$ and $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$, and set $U := g_1^u g_5^{u_5}$, $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widetilde{U} := (g_1g_2)^u g_5^{u_5}$, $\widetilde{\mathbf{W}} := (g_1g_2)^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widehat{U} := (g_1g_2g_3)^u$ and $\widehat{\mathbf{W}} := (g_1g_2g_3)^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output $\textsc{mpk} := (\mathscr{G}, G, U, \mathbf{W}, g_5)$.

Extract($\textsc{id}$) Return $\perp$ when $\textsc{id} \in \mathsf{ChID}$. Update $\mathsf{ExID} := \mathsf{ExID} \cup \{\textsc{id}\}$. We maintain a random function $\mathsf{R}' : \mathsf{IdSp} \to \mathbb{Z}_N$ whose output depends on all bits of $\textsc{id}$. Pick a new $j \in [q]$, sample $r', r_4'', r_4''' \leftarrow \mathbb{Z}_N$ and set

$$K := g_1^{r'} \cdot \left((g_2g_3)^{x_j} X_{4,j}\right) \cdot g_4^{r_4''} \text{ and } K' := g_1^{h(u,\mathbf{w},\textsc{id})r'} \cdot \left((g_2g_3)^{x_j} X_{4,j}\right)^{h(u,\mathbf{w},\textsc{id})} \cdot T_j^{\mathsf{R}'(\textsc{id})} \cdot g_4^{r_4'''}$$

Here we implicitly set $r \in \mathbb{Z}_N$ such that $r = r' \bmod p_1$, $r = x_j \bmod p_2p_3$ and define $\mathsf{R}_n(\textsc{id}) = y \cdot \mathsf{R}'(\textsc{id}) \bmod p_2p_3$ for $\textsc{id} \in \mathsf{ExID}$. We note that the assignment for R_n is consistent since $\mathsf{ExID} \cap \mathsf{ChID} = \emptyset$.

Challenge($\textsc{id}^*, \textsc{m}_0^*, \textsc{m}_1^*$) Return $\perp$ when $\textsc{id}^* \in \mathsf{ExID}$. Update $\mathsf{ChID} := \mathsf{ChID} \cup \{\textsc{id}^*\}$. If $\textsc{m}_\beta^* = 0$, sample $s, s', s_5, s_5' \leftarrow \mathbb{Z}_N$ and set

$$C := g_2^{s'} \cdot H(\widetilde{U}, \widetilde{\mathbf{W}}, \textsc{id}^*)^s \cdot g_5^{s_5} \text{ and } C' := \widetilde{G}^s \cdot g_5^{s_5'}.$$

If $\textsc{m}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathscr{B}$ outputs 1 if and only if **Finalize**(β') $= 1$, i.e., the adversary $\mathscr{A}$ wins the game.

Clearly, if $T_j = \left(g_2^{x_j y}, g_3^{x_j y}\right)$ for all $j \in [q]$, the simulation described above is identical to Game_3. On the other hand, if $T_j = \left(g_2^{x_j y + z_j}, g_3^{x_j y + z_j}\right)$ for all $j \in [q]$, the simulation is identical to Game_4 where we implicitly set $r' = \mathsf{R}'(\textsc{id}) \cdot (x_j y + z_j)$. Therefore we can conclude that $\left|\mathsf{Adv}^3_{\mathscr{A}}(\lambda) - \mathsf{Adv}^4_{\mathscr{A}}(\lambda)\right| \leqslant \mathsf{Adv}^{\mathrm{DDH1}}_{\mathscr{B}}(\lambda)$. ☐

Lemma 12 ($\mathsf{Game}_4 = \mathsf{Game}_5$) *For any adversary $\mathscr{A}$,* $\mathsf{Adv}^4_{\mathscr{A}}(\lambda) = \mathsf{Adv}^5_{\mathscr{A}}(\lambda)$.

Proof. The transformation from Game_4 to Game_5 is just conceptual following the Chinese Remainder Theorem. In both games, the $\mathbb{G}_{p_2p_3}$-parts of K and K' are independent, and so do the $\mathbb{G}_{p_2}$-part of C and C'. Of course, the $\mathbb{G}_{p_1}$-parts of them are still structural. ☐

4.3.5 Phase 4: Epilogue

All games used in Phase 4 is defined in Figure 11.

Lemma 13 ($\mathsf{Game}_5 \approx \mathsf{Game}_6$) *For any probabilistic polynomial time adversary $\mathscr{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}^5_{\mathscr{A}}(\lambda) - \mathsf{Adv}^6_{\mathscr{A}}(\lambda)\right| \leqslant \mathsf{Adv}^{\mathrm{SD3}}_{\mathscr{B}}(\lambda)$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (Q_C + Q_K)\cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathscr{A}$.

Proof. Given $(\mathscr{G}, g_2, g_3, g_4, g_5, X_1X_5, Y_1Y_2Y_3, T)$ where T is a random element of either $\mathbb{G}_{p_2p_5}$ or $\mathbb{G}_{p_1p_2p_5}$, adversary $\mathscr{B}$ simulates the procedures as follows. We implicitly set $g_1 = X_1$ and parse $X_1X_5 = g_1 g_5^x$ from some $x \in \mathbb{Z}_N$, $Y_1Y_2Y_3 = (g_1g_2g_3)^y$ for some $y \leftarrow \mathbb{Z}_N$ and either $T = (g_2g_5)^t$ or $T = (g_1g_2g_5)^t$ for some $t \leftarrow \mathbb{Z}_N$.

Game$_5$, $\boxed{\text{Game}_6}$, Game_7, Game_{Fin}

Initialize
$\mathcal{G} := (N, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{GrpGen}(1^\lambda)$
for $i \in [5]$ **do** $g_i \leftarrow \mathbb{G}_{p_i}$
$v_5 \leftarrow \mathbb{Z}_N$; $G := g_1 g_5^{v_5}$; $\widehat{G} := g_1$
$u, u_5 \leftarrow \mathbb{Z}_N$; $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$
$U := g_1^u g_5^{u_5}$; $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$
$\widehat{U} := g_1^u$; $\widehat{\mathbf{W}} := g_1^{\mathbf{w}}$
$\beta \leftarrow \{0,1\}$
return $(\mathcal{G}, G, U, \mathbf{W}, g_5)$

Extract(ID)
if ID $\in$ ChID **then return** $\perp$
ExID := ExID $\cup$ {ID}
$r, r', r'', r_4, r_4' \leftarrow \mathbb{Z}_N$
$K := \widehat{G}^r \cdot (g_2 g_3)^{r''} g_4^{r_4}$
$K' := H(\widehat{U}, \widehat{\mathbf{W}}, \text{ID})^r \cdot (g_2 g_3)^{r'} g_4^{r_4'}$
return (K, K')

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$)
if $\text{ID}^* \in$ ExID **then return** $\perp$
ChID := ChID $\cup$ {ID*}
if $\text{M}_\beta^* = 0$ **then**
  $s, s', s'', s_5, s_5' \leftarrow \mathbb{Z}_N$
  $C := H(U, \mathbf{W}, \text{ID})^s \cdot g_2^{s'} g_5^{s_5}$
  $C' := G^s \cdot g_2^{s''} g_5^{s_5'}$
  $\boxed{C, C' \leftarrow \mathbb{G}_{p_1 p_2 p_5}}$
  $C, C' \leftarrow \mathbb{G}$
  $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$
else
  $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$
return (C, C')

Finalize(β')
return $(\beta = \beta')$

Figure 11: Game$_5$, Game$_6$, Game$_7$

Initialize Set $G := X_1 X_5$. Sample $u, u_5' \leftarrow \mathbb{Z}_N$ and $\mathbf{w}, \mathbf{w}_5' \leftarrow \mathbb{Z}_N^{2n}$, and set $U := (X_1 X_5)^u g_5^{u_5'}$ and $\mathbf{W} := (X_1 X_5)^{\mathbf{w}} g_5^{\mathbf{w}_5'}$. We implicitly define $\widehat{G} := g_1$, $\widehat{U} := g_1^u$ and $\widehat{\mathbf{W}} := g_1^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output MPK $:= (\mathcal{G}, G, U, \mathbf{W}, g_5)$.

Extract(ID) Return $\perp$ when ID $\in$ ChID. Update ExID := ExID $\cup$ {ID}. Sample $\widehat{r}, \widehat{r}', \widehat{r}'', r_4, r_4' \leftarrow \mathbb{Z}_N$ and set

$$K := (Y_1 Y_2 Y_3)^{\widehat{r}} \cdot (g_2 g_3)^{\widehat{r}''} g_4^{r_4} \quad \text{and} \quad K' := (Y_1 Y_2 Y_3)^{h(u, \mathbf{w}, \text{ID}) \cdot \widehat{r}} \cdot (g_2 g_3)^{\widehat{r}'} \cdot g_4^{r_4'}.$$

Output (K, K'). Here we implicitly set $r = y\widehat{r}$.

Challenge($\text{ID}^*, \text{M}_0^*, \text{M}_1^*$) Return $\perp$ when $\text{ID}^* \in$ ExID. Update ChID := ChID $\cup$ {ID*}. If $\text{M}_\beta^* = 0$, sample $s, t', t'' \leftarrow \mathbb{Z}_N$ and set

$$C := H(U, \mathbf{W}, \text{ID}^*)^s \cdot T^{t'} \quad \text{and} \quad C' = G^s \cdot T^{t''}.$$

If $\text{M}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathcal{B}$ outputs 1 if and only if **Finalize**(β') = 1, i.e., the adversary $\mathcal{A}$ wins the game.

Observe that if $T = (g_2 g_5)^t \in \mathbb{G}_{p_2 p_5}$, the simulation described above is identical to Game$_5$; if $T = (g_1 g_2 g_5)^t$, the simulation is identical to Game$_6$ where the $\mathbb{G}_{p_1}$-parts of C and C' are hidden by $g_1^{tt'}$ and $g_1^{tt''}$, respectively. Therefore we can conclude that $\left|\mathsf{Adv}_{\mathcal{A}}^5(\lambda) - \mathsf{Adv}_{\mathcal{A}}^6(\lambda)\right| \leqslant \mathsf{Adv}_{\mathcal{B}}^{\mathrm{SD3}}(\lambda)$. ☐

Lemma 14 (Game$_6 \approx$ Game$_7$) *For any probabilistic polynomial time adversary $\mathcal{A}$ making Q_K key extraction queries and Q_C challenge queries, there exists an adversary $\mathcal{B}$ such that*

$$\left|\mathsf{Adv}_{\mathcal{A}}^6(\lambda) - \mathsf{Adv}_{\mathcal{A}}^7(\lambda)\right| \leqslant \mathsf{Adv}_{\mathcal{B}}^{\mathrm{SD4}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + (Q_C + Q_K) \cdot \mathsf{poly}(\lambda, n)$ *where* poly *is independent of* $\mathcal{A}$.

Proof. Given $(\mathcal{G}, g_1, g_2, g_5, X_2 X_3 X_4, T)$ where T is a random element of either $\mathbb{G}_{p_1 p_2 p_5}$ or $\mathbb{G}$, adversary $\mathcal{B}$ simulates the procedures as follows. We implicitly parse $X_2 X_3 X_4 = (g_2 g_3 g_4)^x$ for some $x \in \mathbb{Z}_N$ and either $T = (g_1 g_2 g_5)^t$ or $T = g^t$ for some $t \in \mathbb{Z}_N$.

Initialize Sample $v_5 \leftarrow \mathbb{Z}_N$ and set $G := g_1 g_5^{v_5}$ and $\widehat{G} := g_1$. Sample $u, u_5 \leftarrow \mathbb{Z}_N$ and $\mathbf{w}, \mathbf{w}_5 \leftarrow \mathbb{Z}_N^{2n}$, and set $U := g_1^u g_5^{u_5}$, $\mathbf{W} := g_1^{\mathbf{w}} g_5^{\mathbf{w}_5}$, $\widehat{U} := g_1^u$ and $\widehat{\mathbf{W}} := g_1^{\mathbf{w}}$. Randomly pick $\beta \leftarrow \{0,1\}$ and output $\textsc{mpk} := (\mathcal{G}, G, U, \mathbf{W}, g_5)$.

Extract($\textsc{id}$) Return $\perp$ when $\textsc{id} \in \mathsf{ChID}$. Update $\mathsf{ExID} := \mathsf{ExID} \cup \{\textsc{id}\}$. Sample $r, r', r'' \leftarrow \mathbb{Z}_N$ and set

$$K := \widehat{G}^r \cdot (X_2 X_3 X_4)^{r'} \quad \text{and} \quad K' := H(\widehat{U}, \widehat{\mathbf{W}}, \textsc{id})^r \cdot (X_2 X_3 X_4)^{r''}.$$

Output (K, K').

Challenge($\textsc{id}^*, \textsc{m}_0^*, \textsc{m}_1^*$) Return $\perp$ when $\textsc{id}^* \in \mathsf{ExID}$. Update $\mathsf{ChID} := \mathsf{ChID} \cup \{\textsc{id}^*\}$. If $\textsc{m}_\beta^* = 0$, sample $t', t'' \leftarrow \mathbb{Z}_N$ and set

$$C := T^{t'} \quad \text{and} \quad C' := T^{t''}.$$

If $\textsc{m}_\beta^* = 1$, sample $C, C' \leftarrow \mathsf{Sample}_{\mathbb{G}}()$. Output (C, C').

Finalize(β') Output $(\beta = \beta')$.

The algorithm $\mathscr{B}$ outputs 1 if and only if **Finalize**$(\beta') = 1$, i.e., the adversary $\mathscr{A}$ wins game.

Observe that if $T = (g_1 g_2 g_5)^t \in \mathbb{G}_{p_1 p_2 p_5}$, the simulation described above is identical to Game_6; if $T = g^t$, the simulation is identical to Game_7. Therefore we can conclude that $\left|\mathsf{Adv}^6_{\mathscr{A}}(\lambda) - \mathsf{Adv}^7_{\mathscr{A}}(\lambda)\right| \leqslant \mathsf{Adv}^{\mathrm{SD4}}_{\mathscr{B}}(\lambda)$. ☐

Lemma 15 ($\mathsf{Game}_7 \approx \mathsf{Game}_{\mathbf{Fin}}$) *For adversary $\mathscr{A}$, $\left|\mathsf{Adv}^7_{\mathscr{A}}(\lambda) - \mathsf{Adv}^{\mathrm{Fin}}_{\mathscr{A}}(\lambda)\right| \leqslant 2\zeta$.*

Proof. These two games are exactly the same until publicly reversible sampler $\mathsf{Sample}_{\mathbb{G}}$ outputs $\perp$ when encrypting message 0 in $\mathsf{Game}_{\mathrm{Fin}}$. Clearly we can bound the probability of this event by 2ζ where ζ is the error probability of $\mathsf{Sample}_{\mathbb{G}}$. Therefore we can conclude that $\left|\mathsf{Adv}^7_{\mathscr{A}}(\lambda) - \mathsf{Adv}^{\mathrm{Fin}}_{\mathscr{A}}(\lambda)\right| \leqslant 2\zeta$. ☐

Final Analysis. In the last game $\mathsf{Game}_{\mathrm{Fin}}$, ciphertexts for message 0 and 1 are produced following the same procedure, i.e., invoking $\mathsf{Sample}_{\mathbb{G}}()$ twice. Therefore we have $\mathsf{Adv}^{\mathrm{Fin}}_{\mathscr{A}}(\lambda) = 0$ for any adversary $\mathscr{A}$. Combining all lemmas above together, we have proved the main theorem.

5 Conclusion

The paper proposed the first identity based encryption scheme secure against selective opening attack with almost-tight security reduction. The work applied Hofheinz *et al.*'s recent technique for almost-tight secure IBE in the multi-ciphertext setting to Bellare *et al.*'s framework for reaching selective opening security from standard IND-CPA security in the IBE setting. The resulting construction works with bilinear groups whose order is the product of five distinct prime numbers.

Acknowledgement. We want to thank Zhengan Huang for useful suggestion. This work is supported by the National Natural Science Foundation of China (Grant Nos. 61321064, 61371083, 61373154, 61411146001, and 61472142), the Specialized Research Fund for the Doctoral Program of Higher Education of China through the Prioritized Development Projects under Grant 20130073130004, and Science and Technology Commission of Shanghai Municipality (Grant Nos. 14YF1404200, 13JC1403500).

References

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. *IACR Cryptology ePrint Archive*, 2015. 4, 5

[Att14] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *Advances in Cryptology–EUROCRYPT 2014*, pages 557–577. Springer, 2014. 5

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, pages 223–238, 2004. 5

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004*, pages 443–459, 2004. 5

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology-CRYPTO 2001*, pages 213–229. Springer, 2001. 2, 5, 7

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of cryptography*, pages 325–341. Springer, 2005. 11

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *Advances in Cryptology-EUROCRYPT 2009*, pages 1–35. Springer, 2009. 5, 7

[BKP14] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. (hierarchical) identity-based encryption from affine message authentication. In *Advances in Cryptology–CRYPTO 2014*, pages 408–425. Springer, 2014. 5

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *Advances in Cryptology-ASIACRYPT 2001*, pages 514–532. Springer, 2001. 12

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology-EUROCRYPT 2006*, pages 409–426. Springer, 2006. 6

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Advances in Cryptology-CRYPTO 2006*, pages 290–307. Springer, 2006. 2

[BWY11] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In *Theory of Cryptography*, pages 235–252. Springer, 2011. 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14

[CDNO97] Rein Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Advances in Cryptology-CRYPTO'97*, pages 90–104. Springer, 1997. 2, 3, 8

[CGW15] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system abe in prime-order groups via predicate encodings. In *Advances in Cryptology-EUROCRYPT 2015*, pages 595–624. Springer, 2015. 5

[CLL+13] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter ibe and signatures via asymmetric pairings. In *Pairing-Based Cryptography–Pairing 2012*, pages 122–140. Springer, 2013. 5

[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. 3, 4, 5, 11, 17, 18, 20

[DCIP10] Angelo De Caro, Vincenzo Iovino, and Giuseppe Persiano. Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts. In *Pairing-Based Cryptography-Pairing 2010*, pages 347–366. Springer, 2010. 2, 3, 4, 11, 14

[FHKW10] Serge Fehr, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Encryption schemes secure against chosen-ciphertext selective opening attacks. In *Advances in Cryptology–EUROCRYPT 2010*, pages 381–402. Springer, 2010. 5, 12

[GCD+15] Junqing Gong, Jie Chen, Xiaolei Dong, Zhenfu Cao, and Shaohua Tang. Extended nested dual system groups, revisited. *IACR Cryptology ePrint Archive*, 2015. 5

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2006*, pages 445–464, 2006. 5

[HJKS15] Felix Heuer, Tibor Jager, Eike Kiltz, and Sven Schäge. On the selective opening security of practical public-key encryption schemes. In *Public-Key Cryptography–PKC 2015*, pages 27–51. Springer, 2015. 5

[HKS15] Dennis Hofheinz, Jessica Koch, and Christoph Striecks. Identity-based encryption with (almost) tight security in the multi-instance, multi-ciphertext setting. In *Public-Key Cryptography–PKC 2015*, pages 799–822. Springer, 2015. 2, 5, 7, 11, 14

[HLL+15] Jingnan He, Bao Li, Xianhui Lu, Dingding Jia, Haiyang Xue, and Xiaochao Sun. Identity-based lossy encryption from learning with errors. In *Advances in Information and Computer Security*, pages 3–20. Springer, 2015. 5

[HLOV11] Brett Hemenway, Benoît Libert, Rafail Ostrovsky, and Damien Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In *Advances in Cryptology–ASIACRYPT 2011*, pages 70–88. Springer, 2011. 5

[HLQ13] Zhengan Huang, Shengli Liu, and Baodong Qin. Sender-equivocable encryption schemes secure against chosen-ciphertext attacks revisited. In *Public-Key Cryptography–PKC 2013*, pages 369–385. Springer, 2013. 5

[Hof12] Dennis Hofheinz. All-but-many lossy trapdoor functions. In *Advances in Cryptology–EUROCRYPT 2012*, pages 209–227. Springer, 2012. 5

[JR13] Charanjit S Jutla and Arnab Roy. Shorter quasi-adaptive nizk proofs for linear subspaces. In *Advances in Cryptology-ASIACRYPT 2013*, pages 1–20. Springer, 2013. 5

[LDL+14] Junzuo Lai, Robert H Deng, Shengli Liu, Jian Weng, and Yunlei Zhao. Identity-based encryption secure against selective opening chosen-ciphertext attack. In *Advances in Cryptology–EUROCRYPT 2014*, pages 77–92. Springer, 2014. 5, 12

[Lew12] Allison Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *Advances in Cryptology–EUROCRYPT 2012*, pages 318–335. Springer, 2012. 5

[LOS+10] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology–EUROCRYPT 2010*, pages 62–91. Springer, 2010. 5

[LP15] Shengli Liu and Kenneth G Paterson. Simulation-based selective opening cca security for pke from key encapsulation mechanisms. In *Public-Key Cryptography–PKC 2015*, pages 3–26. Springer, 2015. 5

[LW10] Allison Lewko and Brent Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *Theory of Cryptography*, pages 455–479. Springer, 2010. 2, 3, 5, 11

[LW11] Allison Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In *Advances in Cryptology–EUROCRYPT 2011*, pages 547–567. Springer, 2011. 5

[LW12] Allison Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *Advances in Cryptology–CRYPTO 2012*, pages 180–198. Springer, 2012. 5

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. 5

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *Advances in Cryptology–CRYPTO 2010*, pages 191–208. Springer, 2010. 5

[OT12a] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *Advances in Cryptology–EUROCRYPT 2012*, pages 591–608. Springer, 2012. 5

[OT12b] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In *Advances in Cryptology–ASIACRYPT 2012*, pages 349–366. Springer, 2012. 5

[RCS12] Somindu C Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of waters' dual system primitives using asymmetric pairings. In *Public Key Cryptography–PKC 2012*, pages 298–315. Springer, 2012. 5

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in cryptology*, pages 47–53. Springer, 1985. 5

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology–EUROCRYPT 2005*, pages 114–127. Springer, 2005. 3, 4, 5

[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. 2, 5

[Wee14] Hoeteck Wee. Dual system encryption via predicate encodings. In *Theory of Cryptography*, pages 616–637. Springer, 2014. 5