




# Fully Collusion Resistant Traceable Identity-Based Inner Product Functional Encryption

Subhranil Dutta<sup>1</sup> , Tapas Pal<sup>2</sup> , Amit Kumar Singh<sup>3</sup>  and  
Sourav Mukhopadhyay<sup>4</sup> 

<sup>1</sup> University of St. Gallen, Switzerland

<sup>2</sup> Karlsruhe Institute of Technology, KASTEL SRL, Germany

<sup>3</sup> Siksha 'O' Anusandhan (Deemed to be) University, India

<sup>4</sup> Indian Institute of Technology Kharagpur, India

**Abstract.** We present the *first* fully collusion resistant traceable functional encryption (TFE) scheme for identity-based inner product FE (IBIPFE) that directly traces user identities through an efficient tracing procedure. We name such a scheme as *embedded identity TIBIPFE* (EI-TIBIPFE) where secret keys and ciphertexts are computed for vectors, and decryption recovers the inner product between the vectors given the key and ciphertext are associated with the same group identity. Additionally, a secret key corresponds to a user identity for the purpose of tracing. Suppose some of the users linked to a particular group team up and create a pirate decoder that is capable of decrypting the content of the group, then the tracing algorithm extracts the identities of the dishonest users' given black-box access to the decoder. Previously, such schemes were designed for usual public key encryptions. In this work, we construct a fully collusion resistant EI-TIBIPFE scheme from pairings in the standard model. The ciphertext size of our scheme grows sub-linearly with the number of users in the system. We achieve many-target security of tracing, namely the adversary is allowed to ask for multiple secret keys corresponding to many functions, which notably solves an open problem raised by Do, Phan, and Pointcheval [CT-RSA'2020].

**Keywords:** embedded identity · traitor tracing · inner product functional encryption · identity-based inner product functional encryption

## 1 Introduction

A traditional *traitor tracing* (TT) [CFN94] scheme is a multi-receiver system that helps to detect malicious users who deceive the broadcasters by creating a pirate decryption box. More specifically, contents are encrypted under a public key  $\text{mpk}$  and each authorized user indexed with  $j$  is given a sophisticated secret key  $\text{sk}_j$  to recover the contents. Consider a scenario where a collection of dishonest users, called *traitors*, embeds their secret keys into a pirate decoder which decrypts the ciphertext for unauthorized users, thereby causing a significant loss to the content providers. To prevent such impermissible theft, there is a tracing algorithm that uses a dedicated tracing key  $\text{key}$  to identify the traitors in the system. The tracing algorithm is called *public* or *private*, depending on whether the key is available publicly or kept secret by the central authority.

In literature, TT schemes are mainly explored in the context of usual *public key encryption* (PKE) [BF99, TT01, SW98, KD98, KY02a, CPP05, ABP<sup>+</sup>17, FT01, SSW01, BZ14, KY02b, KY02c, LPSS14] or *identity-based encryption* (IBE) [ADM<sup>+</sup>07, GMS12, PT11].

---

E-mail: [subhranil.dutta@unig.ch](mailto:subhranil.dutta@unig.ch) (Subhranil Dutta), [tapas.pal@kit.edu](mailto:tapas.pal@kit.edu) (Tapas Pal), [amitmintu01991@gmail.com](mailto:amitmintu01991@gmail.com) (Amit Kumar Singh), [sourav@maths.iitkgp.ac.in](mailto:sourav@maths.iitkgp.ac.in) (Sourav Mukhopadhyay)



Recently, Do, Phan and Pointcheval [DPP20] bring this feature of traceability into *functional encryption* (FE) [BSW11], a more fine-grained encryption mechanism. The notion is called *traceable functional encryption* (TFE). In particular, they provide a construction of a *traceable inner product functional encryption* (TIPFE) scheme where secret keys are generated for tuples  $(j, \mathbf{u})$  representing user indices and vectors. The ciphertexts are computed for some vectors  $\mathbf{v}$  in such a manner that the decryption reveals nothing about the message  $\mathbf{v}$  except the inner product  $\langle \mathbf{u}, \mathbf{v} \rangle$ . Suppose many secret keys  $\text{sk}_{j,\mathbf{u}}$  for vectors  $\mathbf{u}$  are provided to different users having distinct indices. It may happen that some of these users create a pirate decoder embedding their own secret keys in order to sell it for personal interests. Therefore, anyone from outside who does not have a secret key can learn the inner product value using the pirate decoder. The tracing algorithm of TIPFE is employed to identify such dishonest users in the system.

Following the usual tracing procedures [BSW06, BF99], Do et al. [DPP20] design TIPFE with the following properties:

- *Index tracing.* Their tracing algorithm is designed to find out a set  $T^{\text{index}}$  containing the traitor's indices associated with the secret keys of IPFE. In order to find the actual traitors, the indices in  $T^{\text{index}}$  are *mapped back* to the identities of traitors. Thus, the central authority must maintain a *map* or a look-up table to discover the identities linked to the indices of  $T^{\text{index}}$ . More precisely, the key generation process of [DPP20] encodes the identities as codewords or vectors (having the same length as the IPFE vectors), and the tracing algorithm needs to access the list of these codewords. This makes the key generation inherently *stateful*, which not only dilutes the whole purpose of tracing but is inconvenient for many practical scenarios.
- *Private traceability.* The TIPFE only supports *private* tracing. In other words, the tracing algorithm requires the master secret key of the system, meaning that *only* the central authority can find out the traitors' identities.
- *One-target security.* The TIPFE of [DPP20] is proven secure under the assumption that the adversary queries secret keys  $\text{sk}_{j,\mathbf{u}}$  for a *fixed* target vector  $\mathbf{u}$ , but with multiple indices. The notion is called one-target security and constructing TIPFE with *many-target* security where any polynomial number of target functions can be queried, was left as an open problem.

In this work, we follow the definition of *fully collusion resistant* as proposed by Boneh et al. [BSW06, BW06] for a TT scheme. We say that a TFE scheme is fully collusion resistant if an adversary is allowed to query secret keys corresponding to all indices and many-target functions in the system. In particular, the adversary can query polynomially many secret keys and there is no predefined bound on the number of key queries. According to our knowledge, there does not exist a publicly traceable fully collusion resistant TFE scheme. Nishimaki, Wichs and Zhandry [NWZ16] proposed a framework that publicly traces users' identities from a decoder box and hence eliminates the requirement of the index-identity *map* from the system. However, their TT scheme relies on a heavy cryptographic tool, namely adaptively-secure collusion resistant public key FE scheme for general circuits with compact ciphertexts. To avoid the route of full-fledged FE, Goyal, Koppula, and Waters [GKW19] designed a more efficient identity tracing mechanism and achieved full collusion resistant TT from standard assumptions based on pairings and lattices. However, all existing TT schemes can trace users' identities *only* in plain PKE systems. Recently, Luo et al. [LAWH22] proposed a generic construction of trace and revoked IPFE which traces *only* the indices of traitors. In addition, the encryption algorithm depends on a public directory containing user-specific information that is sampled while generating keys for the users in the system.

Another drawback of the current structure of TIPFE is that the ciphertexts do not contain any information about the source of the message. As a result, the sender can not be recognized during decryption, although it is an essential and desirable feature for any

public key infrastructure. If many broadcasters (or content providers) encrypt their data using the same TIPFE scheme then users having secret keys from one broadcaster can decrypt the content of others. Further, in the context of tracing, this means the tracing must be run across all the users' identities even when the decoder is created to cheat a specific broadcaster. A naive way to resolve this problem is to sample individual TIPFE system for each of these broadcasters. However, this requires maintaining a huge database for storing parameters of all the TIPFE systems, and complications may arise in managing certificates. This shortcoming of TIPFE can be surpassed by introducing identities assigned to the broadcasters and enabling them to encrypt contents under their own individual identities. A secret key can be associated with an additional broadcaster's identity which restricts the user to decrypt only the contents released by that broadcaster. Thus, we ask the following question.

**Open Problem.** *Can we build an efficient broadcaster-identity-based TIPFE under a standard assumption which satisfies (a) identity tracing, (b) public traceability and (c) fully collusion resistant security?*

**Our Contributions.** In this work, we affirmatively answer to the above question. More precisely, our contributions are as follows.

**Embedded identity TIBIPFE.** We formally introduce the notion of *embedded identity traceable identity-based IPFE* (EI-TIBIPFE) where encryption takes place under a broadcaster's identity referred to as group-identity in this work. A secret key of user (or subscriber) corresponds to his/her identity along with a group-identity and decryption is successful whenever the group-identities of key and ciphertext are equal. In other words, users who have subscribed to a specific broadcaster are able to decrypt the content of that broadcaster. There is a dedicated tracing key which is used to trace the dishonest subscribers that are responsible in creation of a decoder box, through a black-box tracing algorithm.

**An intermediate primitive: EIPL-IBIPFE.** To construct fully collusion resistant EI-TIBIPFE, we formalize an intermediate primitive called *embedded identity private linear IBIPFE* (EIPL-IBIPFE). We show a generic transformation to construct EI-TIBIPFE from a EIPL-IBIPFE without assuming existence of any other primitive. Therefore, the answer to the above question boils down to construct the intermediate primitive EIPL-IBIPFE. We build EIPL-IBIPFE under standard group-based assumptions which eventually leads to the following construction:

- *A pairing-based construction.* We propose a *selectively* secure EI-TIBIPFE in a composite-order pairing group based on the standard *decisional 3-party Diffie-Hellman* (D3DH) [BW06, BSW06] and subgroup decision assumptions [GKW19]. We follow a three step approach to build the primitive. In the first step, we construct an *adaptively* secure EI-TIPFE from similar assumption. In second step, we design a *selectively* secure IBIPFE scheme in a prime-order pairing group which is secure under a *target-group-based* assumption. In third step, we upgrade the EI-TIPFE using our IBIPFE. This upgradation is non-trivial, and we devise a non-generic *binding randomization* technique for the final step. The EI-TIBIPFE is publicly traceable, and the size of ciphertext, master public keys grow with  $\sqrt{n}$  and  $\sqrt{k}$ , where  $n$  is the number of users and  $k$  is the length of the embedded user identities.

Our group-based IBIPFE is built upon a target-group-based assumption, namely the *decisional bilinear Diffie-Hellman* (DBDH) assumption in the standard model. All existing group-based IBIPFEs either rely on source group assumptions [ACGU20, AGT21] or the security is proven in the random oracle model [SP19]. It is well-known that target-group-based assumptions are qualitatively weaker than the source group ones [Fre10, DKW21]. Hence, it is worth mentioning that our IBIPFE is the *first* instantiation of a FE scheme

Table 1: Comparison between Traceable FEs

Scheme	Assum.	Size of the component			Tracing	(Func., Secu.)	Identity
		$ \text{mpk} $	$ \text{ct} $	$ \text{sk} $	mode		trace
[DPP20]	DBDH	$m(n + \text{poly}(\lambda))$	$m \cdot \text{poly}(\lambda)$	$\text{poly}(\lambda)$	Private	(IPFE, Sel)	×
[LAWH22] <sup>†</sup>	DDH, DCR, LWE	$m^2 \cdot \text{poly}(\lambda)$	$m^2 \cdot \text{poly}(\lambda)$	$\text{poly}(\lambda)$	Public	(IPFE, Adp)	×
Sec. 7	D3DH	$m \cdot \sqrt{n \cdot k} \cdot \text{poly}(\lambda)$	$m \cdot \sqrt{n \cdot k} \cdot \text{poly}(\lambda)$	$\log n + k + k' + \text{poly}(\lambda)$	Public	(IBIPFE, Sel)	✓

$n$ : number of user;  $m, k, k'$ : dimension of input vector, user identity and group identity respectively; Func., Secu.: functionality and security model; DDH: decisional Diffie–Hellman; DCR: Decisional Composite Residuosity; LWE: Learning with Errors; Sel: selective; Adp: adaptive. [†]: The generic Traceable IPFE of Luo et al. [LAWH22] depends on a unnecessary public directory which we discuss below.

beyond IPFE that is designed from a simple target-group-based assumption in the standard model.

**Many-target tracing security.** We solve the open problem left by Do et al. [DPP20] of building TIPFE that satisfies many-target tracing security. An adversary of our intermediate primitive EIPL-IBIPFE can query secret keys for *many* vectors or functions with multiple associated identities. As a result, the tracing security of EI-TIBIPFE scheme allow the adversary to query secret keys for (polynomially) *many*-target functions which can be linked to an arbitrary number of users in the system. Such functional keys may involve in devising the decoder box. Since all our schemes satisfy the many-target tracing security, we simply ignore the term “many-target” while describing the security of tracing in this paper. In Table 1, we compare our schemes with the TIPFE of [DPP20] with respect to efficiency, functionality and hardness assumptions. Like [BSW06, Fre10, GKRW18, CVW<sup>+</sup>18, NWZ16, GKW19], our TFE schemes are fully collusion resistant meaning that there is no bound on adversary’s secret key queries. On the technical side, we extend the framework of Goyal et al. [GKW19] from tracing the traitors in a normal PKE-system to tracing in an IBIPFE-system. Since IBIPFE provides more finer access control than PKE or IPFE [GKW19, DPP20], one of the main technical contribution is to design a tracing algorithm against more powerful adversaries (with the ability to query more sophisticated keys), which directly traces user identities in a black-box manner. We think our work as a stepping stone towards building efficient TFE schemes with such advanced tracing mechanism for more expressive functionalities beyond IBIPFE. Finally, we summarize our results in the following Theorem.

**Theorem 1** (Informal). *Assuming D3DH assumption, there exist a fully collusion resistant selectively secure EI-TIBIPFE scheme with a public tracing algorithm that can trace user identities from a pirate decoder.*

Recently, Zhandry [Zha21] proposed white-box tracing mechanism for PKE which allows the tracer to inspect the implementation of decoder and prevents some attack scenarios that are inherent to black-box traitor tracing such as availability of the decoder to an outsider. On the other hand, we stress that this work is motivated to design efficient black-box traitor tracing scheme with public/private tracing for specific FEs which are fully collusion resistant.

**Comparison with Luo et al. [LAWH22].** Luo et al. [LAWH22] addressed the many-target security challenge in the IPFE setting but with certain limitations. Notably, the key generation and encryption algorithms can not operate independently. Specifically, these algorithms rely on a public directory which contains vectors that are related to the

indices of secret keys, which are sampled during the key generation procedure. During encryption, the randomness must be sampled based on these vectors, creating an inherent dependency between the encryption and key generation algorithms. This dependency is both unnecessary and undesirable for a traceable IPFE scheme. Furthermore, the one-target secure traceable IPFE proposed by Do et al. [DPP20] does not impose such a restriction. In this work, we not only resolve the many-target security issue of [DPP20] without relying on any such public directory but also extend the inner product functionality to a more natural and practically relevant IBIPFE framework. In terms of efficiency, the ciphertext and master public key sizes of [LAWH22] scale with  $O(m^2)$ , where  $m$  is the size of the input vector. This is unusual in the context of any plain IPFE schemes [ABCP15, ALS16] where the master public key and ciphertext grow linearly with the length of the input vectors. Given these limitations, we believe their protocol is significantly weaker than ours, both from the functionality and efficiency grounds.

**Application.** Although there could be many applications of EI-TIBIPFE, we consider a specific application scenario to understand the importance of the primitive from practical grounds. Suppose the Department of Health (DOH) of a country authorizes certain labs to perform a clinical trial of any drug in order to create a medicine as early as possible. Each lab (playing the role of a broadcaster) encrypts the clinical data of their manufactured medicine under their lab-*id* and scientists from different labs receive secret keys directly from DOH to perform statistical analysis on the encrypted data and learn important characteristics of the medicine produced by their own labs. Such statistical analysis or findings should be kept secret within a lab until the medicine is approved by DOH due to several reasons including financial profits, dignity of the lab. During the trial, it may happen that a scientist of a particular lab  $X$  is compromised and (s)he creates a decoder box by embedding the secret key to sell out sensitive data about the medicine manufactured by lab  $X$ . To prevent this, DOH can employ our EI-TIBIPFE and encode *identification information* of each scientist into their secret keys which is a *tuple* of the form (index number, employ-*id*, lab-*id*/name) in order to facilitate tracing such culprits via a dedicated algorithm.

**Organization.** We discuss our techniques and ideas in Section 2. The notations and complexity assumptions are given Section 3 and our EI-TIBIPFE notion is defined in Section 4. We formally introduce the notions of EIPL-IBIPFE in Section 5 respectively with their syntax and security models. Section 6 presents our generic transformation of EI-TIBIPFE from EIPL-IBIPFE and the security analysis is explicitly discussed in the Appendix B. Next, we present our pairing-based EIPL-IBIPFE in Section 7 with its formal security analysis. Finally, our pairing-based IBIPFE from the DBDH assumption is presented in the Appendix C. We provide additional preliminaries in the Appendix A.

## 2 Technical Overview

We start by reviewing the *embedded identity traitor tracing* (EI-TT) framework of Goyal et al. [GKW19] which is referred to GKW-TT from now on. The tracing algorithm of GKW-TT is designed to trace traitors' identities directly in a PKE-system via an intermediate primitive called *embedded identity private linear broadcast encryption* (EIPL-BE). We extend their framework from PKE to IPFE and define the notion of *embedded identity traceable IPFE* (EI-TIPFE) for tracing identities of traitors in an IPFE system.

We further extend the notion of EI-TIPFE to *embedded identity traceable identity-based IPFE* (EI-TIBIPFE) which allows to trace users that belong to a specific group. We generically construct EI-TIBIPFE from an intermediate primitive *embedded identity private linear IBIPFE* (EIPL-IBIPFE). We *emphasize* that TIPFE is a particular case of TIBIPFE. Thus in this overview, we focus on describing the core techniques behind the realization of TIBIPFE.

**The Framework of GKW-TT.** Goyal et al. [GKW19] designed EI-TT schemes from various standard assumptions. The core idea of [GKW19] was to extend the framework of *private linear broadcast encryption* (PL-BE) [BSW06] and introduce the notion of EIPL-BE. An EIPL-BE has a normal and a special encryption algorithms. The special encryption algorithm of EIPL-BE is associated with the index-position-bit tuple  $(i, \ell, b)$ , whereas in PL-BE [BSW06], it is associated only with the index  $i$ . More specifically, GKW-TT generates secret keys for a tuple  $(j, \text{id})$  and encrypts messages under a tuple  $(i, \ell, b)$  using a tracing key (secret or public) where  $b$  represents the  $\ell$ -th bit of users' identities. The special encryption algorithm is designed in a way to control the ability of decrypting a ciphertext depending on the  $\ell$ -th bit of an identity associated with the secret keys. In particular, a ciphertext computed under the tuple  $(j, \ell, \text{id}_\ell)$  cannot be decrypted by a secret key  $\text{sk}_{j, \text{id}}$  whereas a ciphertext computed under the tuple  $(j, \ell, 1 - \text{id}_\ell)$  is decryptable. This extra access control mechanism enables the tracing algorithm to uncover the dishonest identities in a bit-by-bit manner.

## 2.1 Defining Embedded Identity Traceable IBIPFE

Inspired by the usual TT schemes, Do et al. [DPP20] introduced the notion of *traceable IPFE* (TIPFE) where secret keys of IPFE are additionally associated with an index number mapping to the identity of a user. A tracing algorithm given a black-box access to a decoder box  $\mathcal{D}_u$  designed for a vector  $u$  can be used to extract a set of indices  $T^{\text{index}}$ . The tracing algorithm is said to be correct if  $T^{\text{index}}$  includes only the indices of users who embedded their secret keys into the decoder box.

The main downside of the tracing algorithm of [DPP20] is that the indices in  $T^{\text{index}}$  must be traced back to the actual identities through a central *map*, which becomes problematic for many applications. To overcome this limitation, we extend the notion of TIPFE into *embedded identity TIPFE* (EI-TIPFE) where the users' secret keys are associated with index-identity pair  $(j, \text{id})$  such that  $j \in [n]$  and  $\text{id}$  can be a binary string of length  $k$ . The tracing algorithm now directly extracts a set  $T^{\text{id}}$  containing the identities of traitors. Hence, there is no need of such unnecessary *map* of TIPFE [DPP20].

We note that the encryption of EI-TIPFE is performed independently of the sender's identity, hence receivers are unaware of the source of plaintexts. Moreover, in real applications, it is often the case that a group of users (e.g. employees) possesses a group identity (e.g. the company in which they work). This motivates us to define the notion of *embedded identity traceable IBIPFE* (EI-TIBIPFE) where the secret keys of users are additionally associated with a group identity  $\text{gid} \in \{0, 1\}^{k'}$  and the ciphertexts are computed under a group identity  $\text{gid}'$ . The decryption successfully recovers the inner product if these two group identities are the same, i.e.  $\text{gid} = \text{gid}'$ . It becomes more convenient to trace the traitors since the pirated decoder box  $\mathcal{D}_u$  works with a specific  $\text{gid}$  and one needs to only trace over the set of users that are linked with the  $\text{gid}$  (instead of the set of all users in the system). More formally, our EI-TIBIPFE scheme is defined as follows:

- $\text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) \rightarrow (\text{msk}, \text{mpk}, \text{key})$ : The setup algorithm generates master key pairs and a tracing key.
- $\text{KeyGen}(\text{msk}, i, \text{id}, \text{gid}, u) \rightarrow \text{sk}_u$ : It generates secret keys of a user having index-identity pair  $(i, \text{id})$  and a group identity  $\text{gid}$ .
- $\text{Enc}(\text{mpk}, \text{gid}', v) \rightarrow \text{ct}_v$ : It encrypts a vector  $v$  under a group identity  $\text{gid}'$ .
- $\text{Dec}(\text{sk}_u, \text{ct}_v) \rightarrow \zeta/\perp$ : The decryption recovers the inner product  $\langle u, v \rangle$  if  $\text{gid} = \text{gid}'$ ; otherwise returns  $\perp$ .
- $\text{Trace}^{\mathcal{D}_u}(\text{key}, \text{gid}, u, v^{(0)}, v^{(1)}) \rightarrow T^{\text{id}}$ : It outputs a set  $T^{\text{id}} \subset \{0, 1\}^k$  of the identities of traitors belong to the group identity  $\text{gid}$ .

We say that the tracing is correct if it does not falsely accuse an honest user as traitor and  $T^{\text{id}}$  is a subset of the identities of released secret keys. EI-TIPFE can be viewed as

a particular case of EI-TIBIPFE when  $\text{gid} = \text{gid}'$  always holds. In other words, one can independently define EI-TIPFE from EI-TIBIPFE where  $\text{KeyGen}, \text{Enc}, \text{Trace}^{\mathcal{D}_u}$  do not take a group identity as input and decryption with an honestly generated secret key is always successful.

## 2.2 The framework of EIPL-IBIPFE for tracing identities in IBIPFE

The backbone of constructing EI-TIBIPFE is the notion of EIPL-IBIPFE. The concept of EIPL-IBIPFE is inspired by the primitive of EIPL-BE introduced by [GKW19]. We extend their framework from PKE to FE, more specifically to IPFE or even richer functionality of IBIPFE. Firstly, the EIPL-IBIPFE enables tracing identities directly in an IBIPFE system and secondly the primitive serves the purpose of surpassing the limitation of *one-target tracing* security [DPP20]. In particular, we construct EIPL-IBIPFE where the adversary is allowed to query secret keys for many functions instead of restricting the queries to a single function like [DPP20]. More precisely, EIPL-IBIPFE has the same setup, key generation and encryption algorithms as in EI-TIBIPFE, but there is an additional special encryption algorithm which is core of our advanced tracing mechanism. The ciphertext obtained from the special encryption algorithm is associated with a tuple  $(i, \ell, b)$  where  $b$  is a bit.

- $\text{SplEnc}(\text{key}, \text{gid}', \mathbf{v}, (i, \ell, b)) \rightarrow \text{ct}_{\mathbf{v}}$ : The special encryption algorithm encrypts a message vector  $\mathbf{v}$  for index-position-bit tuple  $(i, \ell, b)$ . If (the special encryption key)  $\text{key}$  is publicly available then it is called *public* EIPL-IBIPFE; otherwise it is known as *private* EIPL-IBIPFE.
- $\text{Dec}(\text{sk}_{\mathbf{u}}, \text{ct}_{\mathbf{v}}) \rightarrow \zeta/\perp$ : If  $\text{ct}_{\mathbf{v}}$  is a ciphertext of normal encryption then the decryption recovers  $\langle \mathbf{u}, \mathbf{v} \rangle$  when  $\text{gid} = \text{gid}'$  holds. On the other hand, if  $\text{ct}_{\mathbf{v}}$  is special-encryption-ciphertext then the decryption algorithm additionally checks whether the index-identity tuple  $(j, \text{id})$  of  $\text{sk}_{\mathbf{u}}$  satisfies the conditions  $(j > i)$  or  $(j = i \wedge \ell = \perp)$  or  $(j = i \wedge \text{id}_{\ell} = 1 - b)$  where  $\text{id}_{\ell}$  is the  $\ell$ -th bit of  $\text{id}$ .

We require our EIPL-IBIPFE to satisfy the following security properties:

- **Normal-hiding.**  $\text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}) \approx_c \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (1, \perp, 0))$ .
- **Index-hiding.**  $\text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \perp, 0)) \approx_c \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^* + 1, \perp, 0))$  if an adversary is not given a key for  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$ .
- **Lower identity-hiding.**  $\text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \perp, 0)) \approx_c \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \ell^*, b^*))$  if an adversary is not given a key for  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = b^*$ .
- **Upper identity-hiding.**  $\text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \ell^*, b^*)) \approx_c \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^* + 1, \perp, 0))$  if an adversary is not given a key for  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = 1 - b^*$ .
- **Message-hiding.**  $\text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(0)}, (i^*, \perp, 0)) \approx_c \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(1)}, (i^*, \perp, 0))$  if all the secret keys associated to  $(i \geq i^*, \text{id}, \text{gid}^*, \mathbf{u})$  satisfy the condition  $\langle \mathbf{v}^{(0)}, \mathbf{u} \rangle = \langle \mathbf{v}^{(1)}, \mathbf{u} \rangle$ .

We call an EIPL-IBIPFE selectively/adaptively secure subject to the selection of the challenge tuple  $(\text{gid}^*, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$  by an adversary before/after the setup and pre-ciphertext key queries. The above security properties of EIPL-IBIPFE facilitates revealing a traitor's identity in a bit-by-bit manner. The role of a special encryption algorithm is similar to that of the indexed-encryption algorithm of [BSW06] except it provides an additional feature that disables the decryption ability of users upon a single bit of the identity. In more detail, the tracing mechanism follows a two-step process:

1. **Index tracing.** The first step is similar to the usual PI-BE or EIPL-BE [BSW06, GKW19], where the indices of dishonest users' are traced. Formally, for each indices  $i \in [n + 1]$ , it finds the probability  $\hat{p}_i^{\text{ind}}$  of the decoder box for successfully decrypting special encryptions to the tuple  $(i, \perp, 0)$ . It outputs  $\text{Index} = \{i \in [n + 1] : \hat{p}_i^{\text{ind}} \text{ and } \hat{p}_{i+1}^{\text{ind}} \text{ are noticeably far}\}$ .
2. **Identity tracing.** The second step is a sub-search technique which is performed to trace the identity for each  $i \in \text{Index}$ . It checks whether the  $\ell$ -th bit in a (possibly)

traitor's identity is zero or one for all index positions  $\ell \in [k]$ . Formally, for each  $i \in \text{Index}$ , and  $\ell \in [k]$ , it finds the probability  $\hat{q}_{i,\ell}^{\text{id}}$  of the decoder box for successfully decrypting special encryptions to the tuple  $(i, \ell, 0)$ .

Finally, for each  $\ell \in [k]$ , it sets  $\text{id}_\ell = 0$  if  $\hat{p}_i^{\text{id}}$  and  $\hat{q}_{i,\ell}^{\text{id}}$  are noticeably far; otherwise 1. The index tracing phase identifies the possible indices of traitors using the index-hiding property. In the second step, the lower identity-hiding and the upper identity-hiding properties ensure that estimated  $\hat{q}_{i,\ell}^{\text{id}}$  is either close to  $\hat{p}_i^{\text{id}}$  or  $\hat{p}_{i+1}^{\text{id}}$ , which indeed enables it to extract the correct bit of  $\text{id}_\ell$  with high probability.

Although the high-level tracing mechanism of EI-TIBIPFE proceeds similar to GKW-TT, it is actually more technically involved than GKW-TT since we need to identify the traitors belonging to a certain group. From the lens of EI-TIBIPFE, it can be seen that GKW-TT has only a *single* group of users (or a *single* broadcaster) in the system whereas we deal with *multiple* group of users (or *multiple* broadcasters) and the adversary is allowed to query secret keys of users belonging to different groups. We formalize the above security properties of EIPL-IBIPFE in order to efficiently trace the traitors from different groups.

### 2.3 EIPL-IBIPFE from pairing

The generic construction discussed above is not a desirable solution as the ciphertext size linearly grows with the number of users in the system. In search of a more efficient solution, we investigate a non-generic group based construction of EIPL-IBIPFE.

Our starting point is the pairing-based EIPL-BE scheme by Goyal et al. [GKW19], which is built upon the PL-BE scheme by Boneh et al. [BSW06]. Let us fix a few notations. For an element  $g$  of the group  $\mathbb{G}$  of order  $N = p \cdot q$  (where  $p, q$  are the primes) and a vector  $\mathbf{a} = (a_1, \dots, a_m)$ , we denote  $(g^{a_1}, \dots, g^{a_m})$  by  $g^{\mathbf{a}}$ . For two vectors  $\mathbf{a}$  and  $\mathbf{b}$ , we denote  $g^{\mathbf{a}} \cdot \mathbf{b} = g^{\langle \mathbf{a}, \mathbf{b} \rangle}$ . Let  $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$  be a bilinear map, and  $\mathbb{G}_p, \mathbb{G}_q$  be the subgroups of  $\mathbb{G}$  of orders  $p$  and  $q$  respectively. Suppose, there are  $n$  parties indexing each by  $i \in [n]$  which is represented as a pair  $(x, y) \in [\sqrt{n}] \times [\sqrt{n}]$ . The components associated to  $x, y$  are called the *row-specific* and *column-specific* components of  $i$  respectively. We say that  $i_1 \equiv (x_1, y_1) > i_2 \equiv (x_2, y_2)$  if either  $x_1 > x_2$  or  $(x_1 = x_2 \wedge y_1 > y_2)$ .

**First step: A simple version without group-identity.** As first step, we consider a simpler situation where the group identity is absent. That is, we first try to build a pairing-based EIPL-IPFE which will eventually lead to an EI-TIPFE scheme. The EIPL-BE of GKW-TT considers two PL-BE-subsystems of Boneh et al. [BSW06] for each bit of the identity, in total, there are  $2k$  such subsystems. While all the subsystems share the same set of row-specific randomness  $\{\alpha_x, r_x\}$  associated to  $x$ , each of it possesses an individual and independently sampled column-specific random value  $\{c_y = \sum_{\ell} c_{y,\ell,b}\}$  so that the key generation algorithm can select an appropriate and different  $c_y$  value for each identity  $\text{id}$ . This prevents mixing terms of different secret keys to create a hybrid key. Inspired from this approach, we carefully upgrade the system of EIPL-BE to make it capable of encrypting vectors  $\mathbf{v} \in \mathbb{Z}_q^m$  instead of a single integer. More specifically, for each component of  $\mathbf{v}$ , we consider  $2k$  PL-BE-subsystems of [BW06], in total, there are  $2km$  such subsystems. In our EIPL-IPFE, all the subsystems share a set of row-specific random vectors  $\{\alpha_x, r_x\}$  and each of it is linked to an independent column-specific random value  $c_y$ . Since our goal is to build a selectively secure EIPL-IBIPFE, in this technical overview for the sake of simplicity, we discuss our EIPL-IPFE with selective security which is based on the selectively secure IPFE of Abdalla et al. [ABCP15]. We present our adaptively secure EIPL-IPFE. The EIPL-IPFE is described as follows:

- The setup samples  $\text{mpk}$  and  $\text{msk}$  using the following procedure where the components are written according to their role in the scheme:

- **General components:** This component is used to blind secret key and linking



together the message-embedding and id-specific components during decryption. Explicitly, we sample  $\beta \leftarrow \mathbb{Z}_N$  and include  $E_q = g_q^\beta$  in mpk.

- **Row-specific components:** To encrypt the message vector at each row position, the row-specific components are used during encryption. Concretely, we sample  $\alpha_x, \mathbf{r}_x \leftarrow \mathbb{Z}_N^m$  and compute  $\mathbf{E}_{q,x} = (g_q^{\beta \mathbf{r}_{x,j}})_{j \in [m]}$ ,  $\mathbf{G}_{q,x} = (e(g_q, g_q)^{\beta \alpha_{x,j}})_{j \in [m]}$ ,  $\mathbf{E}_x = (g^{\mathbf{r}_{x,j}})_{j \in [m]}$ ,  $\mathbf{G}_x = (e(g, g)^{\alpha_{x,j}})_{j \in [m]}$ .
- **Column-specific components:** These components are used to embed id-specific components into the ciphertext. We sample  $c_{y,\ell,b} \leftarrow \mathbb{Z}_N$  and compute  $H_{y,\ell,b} = g^{c_{y,\ell,b}}$ .

Collecting all the components together, we set mpk (= key) and msk as

$$\text{mpk} = \left( \underbrace{E_q}_{\text{general components}}, \underbrace{\left\{ \begin{array}{cc} \mathbf{E}_x, & \mathbf{G}_x, \\ \mathbf{E}_{q,x}, & \mathbf{G}_{q,x} \end{array} \right\}_x}_{\text{row-specific components}}, \underbrace{\{H_{y,\ell,b}\}_{y,\ell,b}}_{\text{column-specific components}} \right), \text{msk} = \left( \begin{array}{c} \{\alpha_x, \mathbf{r}_x\}_x \\ \{c_{y,\ell,b}\}_{y,\ell,b} \end{array} \right).$$

- The user secret key associated to the tuple  $(i, \text{id}, \mathbf{u})$  is set as follows:

$$K_1 = g^{\langle \alpha_x, \mathbf{u} \rangle + \langle \mathbf{r}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} c_{y,\ell, \text{id}_\ell} \text{ where } i \equiv (x, y)$$

- To encrypt the message vector  $\mathbf{v}$  to an index-position-bit tuple  $(i^* \equiv (x^*, y^*), \ell^*, b^*)$  using the tracing key  $\text{key} = \text{mpk}$ , the encryptor samples  $t, s_x \leftarrow \mathbb{Z}_N$  and compute:

- **Row-specific components:** The components are categorized according to  $x > x^*$ ,  $x = x^*$  and  $x < x^*$  as follows:

For  $x > x^*$ :

1. *Linking component:*  $\mathbf{R}_x = \mathbf{E}_{q,x}^{s_x}$ .
2. *Message-embedding components:*  $\mathbf{I}_x = e(g_q, g_q)^{\mathbf{v}} \cdot \mathbf{G}_{q,x}^{s_x t}$ ,  $A_x = E_q^{s_x t}$ .

For  $x = x^*$ :

1. *Linking component:*  $\mathbf{R}_x = \mathbf{E}_x^{s_x}$ .
2. *Message-embedding components:*  $\mathbf{I}_x = e(g_q, g_q)^{\mathbf{v}} \cdot \mathbf{G}_x^{s_x t}$ ,  $A_x = g^{s_x t}$ .

For  $x < x^*$ : The linking components  $A_x$  and message-embedding components  $\mathbf{I}_x, \mathbf{R}_x$  are randomly chosen from  $\mathbb{G}$  and  $\mathbb{G}^m$  respectively.

- **Column-specific components:** The column components are *id-specific components* which are sampled based on a restriction over  $y, \ell, b$  as follows:

For  $(y > y^*)$  or  $((y = y^*) \wedge (\ell, b) \neq (\ell^*, b^*))$ :  $C_{y,\ell,b} = H_{y,\ell,b}^t$

Otherwise:  $C_{y,\ell,b} = H_{y,\ell,b}^t \cdot h_p$  where  $h_p \leftarrow \mathbb{G}_p$ .

Putting all the components together, we finally get the ciphertext

$$\text{ct}_{\mathbf{v}} = (\{\mathbf{R}_x, \mathbf{I}_x, A_x\}_x, \{C_{y,\ell,b}\}_{y,\ell,b})$$

The normal encryption is the same as special encryption when run with  $(i^*, \ell^*, b^*) = (1, \perp, 0)$ .

- Recall that the successful decryption occurs when the relation  $\mathcal{R}$  (say) defined as  $i > i^*$  or  $(i = i^* \wedge \ell^* = \perp)$  or  $(i = i^* \wedge \text{id}_{\ell^*} = 1 - b^*)$  between  $(i, \text{id})$  of secret key and  $(i^*, \ell^*, b^*)$  of the ciphertext holds.

- **Checking the relation  $\mathcal{R}$ :** If  $i > i^*$  then either  $x > x^*$  or  $x = x^* \wedge y > y^*$  holds. To check if the relation  $\mathcal{R}$  holds, we compute the following term:

$$\mathcal{R}_{\text{check}} = \frac{e(\mathbf{R}_x, \prod_{\ell \in [k]} C_{y, \ell, \text{id}_\ell}^{\mathbf{u}})}{e(K_1, A_x)} = \begin{cases} (e(g_q, g_q)^{\beta_{s_x t}(\alpha_x, \mathbf{u})})^{-1} & \text{if } x > x^* \\ (e(g, g)^{s_x t(\alpha_x, \mathbf{u})})^{-1} & \text{if } x = x^* \end{cases}$$

Finally, we compute  $\langle \mathbf{u}, \mathbf{v} \rangle = \log_{e(g_q, g_q)}((\mathbf{I}_x \cdot \mathbf{u}) \cdot \mathcal{R}_{\text{check}})$  where

$$\mathbf{I}_x \cdot \mathbf{u} = \begin{cases} e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g_q, g_q)^{\beta_{s_x t}(\alpha_x, \mathbf{u})} & \text{if } x > x^* \\ e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g, g)^{s_x t(\alpha_x, \mathbf{u})} & \text{if } x = x^* \end{cases}$$

The normal-hiding security directly follows from the construction of the EIPL-IPFE scheme. The main intuition behind the index-hiding security proof is that if an adversary does not have a secret key for the index  $i^* = (x^*, y^*)$  then the factor of  $C_{y^*, \ell, b}$  which belongs to  $\mathbb{G}_p$  can be chosen undetectably, added, and removed. Similar techniques are used while proving the lower identity-hiding and upper identity-hiding security of the scheme. In the message-hiding security, an adversary can not distinguish between the special encryption of the challenge vectors  $\mathbf{v}^{(0)}, \mathbf{v}^{(1)}$  to the index-position-bit tuple  $(i^* = (x^*, y^*), \perp, 0)$  with the restriction that the adversary is allowed to query secret keys for  $i \geq i^*$  satisfying  $\langle \mathbf{v}^{(0)}, \mathbf{u} \rangle = \langle \mathbf{v}^{(1)}, \mathbf{u} \rangle$ . For each index-factor  $x \in [\sqrt{n}]$ , there is an IPFE system encrypting the message vector in the target group  $\mathbb{G}_T$ . Hence, one can hope to prove the message-hiding security using the technique of [ALS16]. Finally, we note that the size of the ciphertext grows linearly with  $\sqrt{n}$  and  $\sqrt{k}$ , similar to previous TT schemes [BSW06, GKW19].

**From EIPL-IPFE to EIPL-IBIPFE.** Next, our aim is to upgrade the system into a more advanced one where a user's secret key is additionally associated with a group identity  $\text{gid}$  and decryption is successful only when the underlying message is encrypted under the same  $\text{gid}$ . In particular, we try to build EIPL-IBIPFE which provides more finer access control than EIPL-IPFE. A natural first attempt is to generically construct EIPL-IBIPFE from EIPL-IPFE and an IBE scheme. However, any such attempt would fail to provide the message-hiding security due to common mix and match attacks [ACGU20]. More specifically, a *mixed* secret key obtained by combining an authorized IBE key with an unauthorized EIPL-IPFE key can be used to decrypt an undesirable ciphertext. Thus, it is advisable to build such a scheme in a non-generic manner.

We see that the above construction of EIPL-IPFE (or EIPL-BE [GKW19]) encrypts the message in the target group. Thus, one needs to have an IBIPFE scheme based on a target group assumption so that it is well fitted into our EIPL-IPFE system. However, all known pairing-based IBIPFE schemes [AGT21, ACGU20] are based on dual system encryption mechanisms and hence naturally depend on various source group assumptions (such as subgroup decision assumptions). Therefore, our next step is to construct an IBIPFE scheme based on the plain DBDH assumption.

**Second Step: A DBDH-based IBIPFE.** Our starting point is the Water's IBE [Wat05] based on the plain DBDH assumption in the standard model. We upgrade their scheme in a natural way to enable encrypting vectors under a given identity. Although the construction is simple, it is an interesting extension of [Wat05] and motivates to build primitives like attribute-based IPFE from target-group-based assumptions in future.

- To compute the master keys, we sample  $g, g_2, u' \leftarrow \mathbb{G}$ ,  $\mathbf{u} = (u_i) \leftarrow \mathbb{G}^{k'}$ ,  $\psi \leftarrow \mathbb{Z}_p^m$  and set  $\mathbf{g}_1 = g^\psi$ ,  $\mathbf{g}_2 = g_2^\psi$  and output  $\text{mpk} = (\mathbf{g}_1, \mathbf{g}_2, u', \mathbf{u}, g)$ ,  $\text{msk} = \mathbf{g}_2$ .
- To generate a secret key corresponding to  $(\text{gid}, \mathbf{y})$ , we first define  $\mathcal{V} = \{i \in [k'] : \text{gid}_i = 1\}$  and set  $\text{H}(\text{gid}) = u' \prod_{i \in \mathcal{V}} u_i \in \mathbb{G}$ . Then, we sample  $r \leftarrow \mathbb{Z}_p$  and output the secret key as  $\text{sk}_{\mathbf{y}} = (d_1 = g_2^{\langle \psi, \mathbf{y} \rangle} \cdot \text{H}(\text{gid})^r, d_2 = g^r)$ .

- To encrypt a message vector  $\mathbf{x}$  with respect to a group-identity  $\text{gid}'$ , we first sample  $t \leftarrow \mathbb{Z}_p$  and then output the ciphertext as  $\text{ct}_{\mathbf{x}} = (C_1 = e(g, g_2)^{\psi t + \mathbf{x}}, C_2 = g^t, C_3 = H(\text{gid}'^t))$ .
- In decryption phase, if  $\text{gid} = \text{gid}'$ , outputs  $\langle \mathbf{x}, \mathbf{y} \rangle = (C_1 \cdot \mathbf{y}) \cdot \frac{e(d_2, C_3)}{e(d_1, C_2)}$ .

At a very high level, for proving the indistinguishability security of the above IBIPFE, we partially rely on the ideas of [AGT21] where they use dual system encryption techniques to prove the security of their attribute-based IPFE scheme based on source group assumptions. However, we aim to prove security based on the plain DBDH assumption, and there is no known way to use dual system encryption methodologies in a prime-order group-based construction. We consider a different approach. Suppose that the challenge message vectors are  $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}$  and the challenge identity is  $\text{gid}^*$ . The reduction begins by sampling a random *orthogonal* (full rank) matrix  $\mathbf{F}$  satisfying the condition  $\mathbf{F} \cdot (\mathbf{x}^{(0)} - \mathbf{x}^{(1)}) = \mathbf{e}_1$  where  $\mathbf{e}_1$  is the first canonical basis vector. Accordingly the master key component  $\alpha$  is switched to  $\mathbf{F}^\top \tilde{\alpha}$ . As the full rank matrix  $\mathbf{F}$  is chosen uniformly at random, this transformation is statistically indistinguishable to the adversary's view. We observe that the challenge vectors are used in this hybrid to generate the master key pairs. Hence, we consider the selective security model where the adversary is restricted to submit the challenge messages before seeing any public parameter of the system. In the next hybrid, we use the DBDH assumption in order to hide the information of the challenge bit. Given a DBDH instance  $(g^a, g^b, g^c, g_T^{abc})$ , we can extend the group elements into vectors as  $g^a, g^b, g^c$  where  $\mathbf{a} = (a, a_2, \dots, a_m), \mathbf{b} = (b, \dots, b), \mathbf{c} = (c, \dots, c)$ . It allows us to embed the DBDH-instance into the master keys. In the secret key query phase, we define identity encoding functions based on  $\text{gid}^*$  (similar to [Wat05]) to correctly simulate the accepting and non-accepting key queries. Finally, to simulate the challenge ciphertext we implicitly set  $g_2 = g^b$  and  $t = c$  so that ciphertext component  $C_1$  transforms to  $e(g, g)^{\mathbf{F}^\top \cdot (\mathbf{w} - \mathbf{b}\mathbf{b}\mathbf{e}_1 + \mathbf{b}\mathbf{F}\mathbf{x}^{(0)})}$  due to the choice of  $\mathbf{F}$ , where  $\mathbf{w}$  represents the component wise multiplication of the vectors  $\mathbf{a}, \mathbf{b}$  and  $\mathbf{c}$ . Now, observe that the challenge bit  $\mathbf{b}$  only occurs in the first entry of  $(\mathbf{w} - \mathbf{b}\mathbf{b}\mathbf{e}_1 + \mathbf{b}\mathbf{F}\mathbf{x}^{(0)})$  and at the same time the DBDH-challenge element  $abc$  is encoded in the first entry of  $\mathbf{w}$ . Hence, the security of our IBIPFE follows from the plain DBDH assumption. In Appendix C, we give the full security analysis with selective identity for simplicity of exposition, however, we are hopeful that using the techniques of [Wat05, BR09] one can prove the security with adaptive identity.

**Third Step: A combination of our EIPL-IPFE and IBIPFE.** The next and final step towards the goal of achieving EIPL-IBIPFE is to combine the techniques of our EIPL-IPFE and IBIPFE. Recall that, in EIPL-IBIPFE, the users are linked with different group identities. Therefore, the secret key of a user is now associated with a tuple of the form  $(i, \text{id}, \text{gid}, \mathbf{u})$ . A *trivial* combination of a secret key of EIPL-IPFE (which corresponds to the tuple  $(i, \text{id}, \mathbf{u})$ ) and a secret key of IBIPFE (which corresponds to the tuple  $(\text{gid}, \mathbf{u})$ ) takes the form

$$\underbrace{K_1 = g^{\langle \alpha_{\mathbf{x}, \mathbf{u}} \rangle + \langle \mathbf{r}_{\mathbf{x}, \mathbf{u}} \rangle \sum_{\ell \in [k]} c_{y, \ell, \text{id}_\ell}}}_{(i, \text{id})\text{-specific key component}}, \quad \underbrace{K_2 = f^{\langle \psi_{\mathbf{x}, \mathbf{u}} \rangle} H(\text{gid})^{\tilde{r}}, K_3 = g^{\tilde{r}}}_{\text{gid-specific key component}}$$

where  $\tilde{r} \leftarrow \mathbb{Z}_p, \psi_{\mathbf{x}} \leftarrow \mathbb{Z}_p^m$ . However, such a *combined* key would lead to a mix-and-match attack. For example, given secret keys  $\text{sk}_{i, \text{id}, \text{gid}} = (K_1, K_2, K_3)$  for  $(i, \text{id}, \text{gid}, \mathbf{u})$  and  $\text{sk}_{i', \text{id}', \text{gid}'} = (K'_1, K'_2, K'_3)$  for  $(i', \text{id}', \text{gid}', \mathbf{u})$ , an adversary can easily construct a new legitimate secret key  $\text{sk}_{i, \text{id}, \text{gid}'} = (K_1, K'_2, K'_3)$  for the tuple  $(i, \text{id}, \text{gid}', \mathbf{u})$  which may lead to an attack to the system. This is because in the *index-hiding* security experiment of EIPL-IBIPFE we allow secret key queries for all tuples but  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$ . Therefore, the adversary can query secret keys  $\text{sk}_{i^*, \text{id}, \text{gid} \neq \text{gid}^*}, \text{sk}_{i \neq i^*, \text{id}, \text{gid}^*}$  and compute a *new* secret key  $\text{sk}_{i^*, \text{id}, \text{gid}^*}$  which can be used against the *index-hiding* security experiment.

To this end, we devise a *binding randomization* mechanism that prevents this mix-and-match attack. More specifically, we sample a random element  $\hat{r}$  from the underlying ring

to *bind* the  $(i, \text{id})$ -specific and  $\text{gid}$ -specific key components as follows:

$$\underbrace{K_1 = g^{\langle \alpha_x, \mathbf{u} \rangle + \widehat{r} \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} c_{y, \ell, \text{id}_\ell}}}_{(i, \text{id})\text{-specific key component}}, \quad \underbrace{K_2 = f^{\langle \psi_x, \mathbf{u} \rangle} \text{H}(\text{gid})^{\widetilde{r} \cdot \widehat{r}}, K_3 = g^{\widetilde{r} \cdot \widehat{r}}}_{\text{gid-specific key component}}$$

Note that, the binding randomness  $\widehat{r}$  also plays the role of binding ciphertext components of EIPL-IPFE and IBIPFE during encryption. Thus, the binding randomness is also required in the generation of master public key of the system.

At the time encryption, the message vector  $\mathbf{v}$  is encrypted under a group identity  $\text{gid}'$ . The encoding  $\text{H}(\text{gid}')$  appears in both the groups  $\mathbb{G}$  and  $\mathbb{G}_q$ . Thus, we can not directly use the encryption mechanism of our IBIPFE for designing the encryption algorithm of EIPL-IBIPFE. To overcome this obstacle, we define a projection  $\text{H}_q$  of the group-identity-encoding function  $\text{H}$  into the subgroup  $\mathbb{G}_q$  of  $\mathbb{G}$  as follows:

$$\begin{aligned} \text{Given group elements: } & \vartheta'_p, \{\vartheta_{p,i}\}_{i \in [k']}, \vartheta'_q, \{\vartheta_{q,i}\}_{i \in [k']} \text{ with } \vartheta' = \vartheta'_p \vartheta'_q, \vartheta_i = \vartheta_{p,i} \vartheta_{q,i} \\ \text{define: } & \text{H}(\text{gid}) = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i, \quad \text{H}_q(\text{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i} \end{aligned}$$

where  $\mathcal{V} = \{i \in [k'] : \text{gid}_i = 1\}$ . Equipped with this ideas, we describe our EIPL-IBIPFE as follows.

- The setup samples  $\text{mpk}$  and  $\text{msk}$  using the following procedure where the components are written according to their role in the scheme:
  - **General components:** This component is used to blind secret key and linking together the message-embedding,  $\text{id}$ -specific and  $\text{gid}$ -specific components during decryption. Explicitly, we sample  $\beta \leftarrow \mathbb{Z}_N$  and include  $E_q = g_q^\beta$  in  $\text{mpk}$ .
  - **Row-specific components:** We sample  $\alpha_x, \psi_x, \mathbf{r}_x \leftarrow \mathbb{Z}_N^m, \widehat{r} \leftarrow \mathbb{Z}_N$  and compute  $\mathbf{E}_{q,x} = (g_q^{\beta \widehat{r} r_{x,j}})_{j \in [m]}$ ,  $\mathbf{G}_{q,x} = (e(g_q, g_q)^{\beta \alpha_{x,j}})_{j \in [m]}$ ,  $\mathbf{W}_{q,x} = (e(f_q, g_q)^{\beta \psi_{x,j}})_{j \in [m]}$ ,  $\mathbf{E}_x = (g^{\widehat{r} r_{x,j}})_{j \in [m]}$ ,  $\mathbf{G}_x = (e(g, g)^{\alpha_{x,j}})_{j \in [m]}$ ,  $\mathbf{W}_x = (e(f, g)^{\psi_{x,j}})_{j \in [m]}$ .
  - **Column-specific components:** We sample  $c_{y,\ell,b} \leftarrow \mathbb{Z}_N$  and compute  $H_{y,\ell,b} = g^{c_{y,\ell,b}}$ .
  - **gid-specific components:** These components are used in the  $\text{gid}$ -encoding functions  $\text{H}$  and  $\text{H}_q$  to embed  $\text{gid}$ -specific components in the ciphertext. We sample  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q, \vartheta_p \leftarrow \mathbb{G}_p^{k'}, \vartheta_q \leftarrow \mathbb{G}_q^{k'}$  and set  $\vartheta' = \vartheta'_p \vartheta'_q, \vartheta = (\vartheta_{p,i} \vartheta_{q,i})_{i \in [k']}$ .

Collecting all the components together, we set  $\text{mpk}$  (= key) and  $\text{msk}$  as

$$\text{mpk} = \left( \underbrace{E_q}_{\text{general components}}, \underbrace{\left\{ \begin{array}{ccc} \mathbf{E}_x, & \mathbf{G}_x, & \mathbf{W}_x \\ \mathbf{E}_{q,x}, & \mathbf{G}_{q,x}, & \mathbf{W}_{q,x} \end{array} \right\}}_{\text{row-specific components}}, \underbrace{\left( \begin{array}{c} \vartheta', \vartheta'_\beta \\ \vartheta, \{\vartheta_{q,i}^\beta\}_i \end{array} \right)}_{\text{gid-specific components}}, \underbrace{\{H_{y,\ell,b}\}_{y,\ell,b}}_{\text{column-specific components}} \right); \text{msk} = \left( \begin{array}{c} \{\alpha_x, \mathbf{r}_x, \psi_x\}_x \\ \widehat{r}, \{c_{y,\ell,b}\}_{y,\ell,b} \end{array} \right).$$

- The user secret key associated to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  is set as

$$\underbrace{K_1 = g^{\langle \alpha_x, \mathbf{u} \rangle + \widehat{r} \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} c_{y, \ell, \text{id}_\ell}}}_{(i, \text{id})\text{-specific key component}}, \quad \underbrace{K_2 = f^{\langle \psi_x, \mathbf{u} \rangle} \text{H}(\text{gid})^r, K_3 = g^r}_{\text{gid-specific key components}}$$

where  $i \equiv (x, y)$ ,  $\widetilde{r} \leftarrow \mathbb{Z}_N$  and  $r = \widehat{r} \cdot \widetilde{r}$ .

- To encrypt the message vector  $\mathbf{v}$  under an index-position-bit tuple  $(i^* \equiv (x^*, y^*), \ell^*, b^*)$  and a group identity  $\text{gid}'$  using the tracing key  $\text{key}$  (=  $\text{mpk}$ ), the encryptor samples  $t, s_x \leftarrow \mathbb{Z}_N$  and compute:

- **Row-specific components:** The components are categorized according to  $x > x^*, x = x^*$  and  $x < x^*$  as follows:  
**For  $x > x^*$ :**

1. *Linking component*:  $\mathbf{R}_x = \mathbf{E}_{q,x}^{s_x}$ .
2. *gid-specific component*:  $B_x = H_q(\text{gid}')^{\beta s_x t}$ .
3. *Message-embedding components*:  $\mathbf{I}_x = e(g_q, g_q)^v \cdot \mathbf{G}_{q,x}^{s_x t} \cdot \mathbf{W}_{q,x}^{s_x t}$ ,  $A_x = E_q^{s_x t}$ .

**For  $x = x^*$ :**

1. *Linking component*:  $\mathbf{R}_x = \mathbf{E}_x^{s_x}$ .
2. *gid-specific component*:  $B_x = H(\text{gid}')^{s_x t}$ .
3. *Message-embedding components*:  $\mathbf{I}_x = e(g_q, g_q)^v \cdot \mathbf{G}_x^{s_x t} \cdot \mathbf{W}_x^{s_x t}$ ,  $A_x = g^{s_x t}$ .

**For  $x < x^*$ :** The linking components  $A_x$ , gid-specific components  $B_x$  and message-embedding components  $\mathbf{I}_x, \mathbf{R}_x$  are randomly chosen from  $\mathbb{G}$  and  $\mathbb{G}^m$  respectively.

- **Column-specific components:** The column components or *id-specific components* are sampled based on a restriction over  $y, \ell, b$  as follows:

**For  $(y > y^*)$  or  $((y = y^*) \wedge (\ell, b) \neq (\ell^*, b^*))$ :**  $C_{y,\ell,b} = H_{y,\ell,b}^t$ .

**Otherwise:**  $C_{y,\ell,b} = H_{y,\ell,b}^t \cdot h_p$  where  $h_p \leftarrow \mathbb{G}_p$ .

Putting all the components together, we finally get the ciphertext

$$\text{ct}_v = (\{\mathbf{R}_x, B_x, \mathbf{I}_x, A_x\}_x, \{C_{y,\ell,b}\}_{y,\ell,b}).$$

The normal encryption is the same as special encryption when run with  $(i^*, \ell^*, b^*) = (1, \perp, 0)$ .

- Recall that, the successful decryption occurs when both the relations  $\mathcal{R}$  (say) defined as  $i > i^*$  or  $(i = i^* \wedge \ell^* = \perp)$  or  $(i = i^* \wedge \text{id}_{\ell^*} = 1 - b^*)$  and  $\mathcal{R}'$  (say) defined as  $\text{gid} = \text{gid}'$  between  $(i, \text{id}, \text{gid})$  of secret key and  $(\text{gid}', (i^*, \ell^*, b^*))$  of the ciphertext hold.
  - **Checking the relation  $\mathcal{R}$ :** To check the relation  $\mathcal{R}$  holds, we compute the following term:

$$\mathcal{R}_{\text{check}} = \frac{e(\mathbf{R}_x, \prod_{\ell \in [k]} C_{y,\ell, \text{id}_\ell}^u)}{e(K_1, A_x)} = \begin{cases} (e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle})^{-1} & \text{if } x > x^* \\ (e(g, g)^{s_x t \langle \alpha_x, \mathbf{u} \rangle})^{-1} & \text{if } x = x^* \end{cases}$$

- **Checking the relation  $\mathcal{R}'$ :** To check the relation  $\mathcal{R}'$  holds, we compute the following term:

$$\mathcal{R}'_{\text{check}} = \frac{e(K_3, B_x)}{e(K_2, A_x)} = \begin{cases} (e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle})^{-1} & \text{if } x > x^* \\ (e(f, g)^{s_x t \langle \psi_x, \mathbf{u} \rangle})^{-1} & \text{if } x = x^* \end{cases}$$

Finally, we compute  $\langle \mathbf{u}, \mathbf{v} \rangle = \log_{e(g_q, g_q)} ((\mathbf{I}_x \cdot \mathbf{u}) \cdot \mathcal{R}_{\text{check}} \cdot \mathcal{R}'_{\text{check}})$  where

$$\mathbf{I}_x \cdot \mathbf{u} = \begin{cases} e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle} & \text{if } x > x^* \\ e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g, g)^{s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(f, g)^{s_x t \langle \psi_x, \mathbf{u} \rangle} & \text{if } x = x^* \end{cases}$$

We consider selective security for our EIPL-IBIPFE where the adversary submits both the challenge message vectors and the challenge group identity before receiving any public parameter of the system. The security analysis is more involved and challenging in EIPL-IBIPFE. For instance, the index-hiding security game of EIPL-IPFE (or EIPL-BE of [GKW19]) does not allow an adversary  $\mathcal{A}$  to query a secret key for the challenge index  $i^*$ , however, it is not the same for EIPL-IBIPFE. In this case,  $\mathcal{A}$  can ask for a secret key associated to the tuple  $(i^*, \text{id}, \text{gid} \neq \text{gid}^*)$  where  $\text{gid}^*$  is the challenge group identity. The *binding* randomness plays an important role in simulating the additional secret keys associated to the tuples of the form  $(i^*, \text{id}, \text{gid} \neq \text{gid}^*)$ . In particular, we introduce a *new* modified version of D3DH assumption and crucially embed some parts of the instance into the *binding* randomness  $\hat{r}$  to simulate the additional keys. We show the generic security of

our modified D3DH assumption in Section 3.1. On the other hand, the message-hiding security is proved utilizing the techniques that we devise while proving the security of IBIPFE. However, we need to rely on the D3DH assumption and extend the proof techniques of IBIPFE from prime-order group to composite-order group setting to make it compatible with the system of EIPL-IBIPFE. Although the top-level idea is inspired from the proof analysis of [GKW19], we can not directly use their techniques because our EIPL-IBIPFE provides more finer access control and the adversary is more powerful in the sense that it is entitled to query secret keys decrypting the challenge ciphertext. The complete security analysis is given in Section 7.2.

### 3 Preliminaries

**Notations.** Let  $\lambda \in \mathbb{N}$  be a security parameter and  $\text{poly}(\lambda)$  be a polynomial in  $\lambda$ . For a prime  $p$ , let  $\mathbb{Z}_p$  denotes the field  $\mathbb{Z}/p\mathbb{Z}$ . For a set  $S$ , we use the notation  $s \leftarrow S$  to indicate the fact that  $s$  is sampled uniformly at random from a finite set  $S$ . We write  $x \leftarrow \mathcal{X}$  to denote that the element  $x$  is sampled at random according to the distribution  $\mathcal{X}$ . For any natural number  $n$ ,  $[n]$  denotes the set  $\{1, 2, \dots, n\}$ . We use a bold lower-case letter e.g.,  $\mathbf{a}$  to denote a vector, and a bold upper-case letter e.g.,  $\mathbf{A}$  denotes a matrix. The  $i$ -th element of the vector  $\mathbf{a}$  is expressed as  $a_i$ , and  $(i, j)$ -th element of a matrix  $\mathbf{A}$  is represented by  $a_{i,j}$ . The transpose of a matrix  $\mathbf{A}$  is denoted by  $\mathbf{A}^\top$ . Let  $\mathbf{u}, \mathbf{v} \in \mathbb{Z}^m$ , then the inner product between the vectors is defined as  $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^m u_i v_i \in \mathbb{Z}$ . A function  $\text{negl} : \mathbb{N} \rightarrow \mathbb{R}$  is said to be *negligible* if for all  $c \in \mathbb{N}$  there exists a  $\lambda_c \in \mathbb{N}$  such that  $\text{negl}(\lambda) \leq \frac{1}{\lambda^c}$  for all  $\lambda > \lambda_c$ . An algorithm  $A$  is said to be a probabilistic polynomial time (PPT) algorithm if it is modeled as a probabilistic Turing machine that runs in time  $\text{poly}(\lambda)$ . If for any PPT adversary  $\mathcal{A}$  such that  $|\Pr[\mathcal{A}(1^\lambda, X) = 1] - \Pr[\mathcal{A}(1^\lambda, Y) = 1]|$  is *negligible* in  $\lambda$ , then we say that the two distributions are indistinguishable, denoted by  $X \approx Y$ .

#### 3.1 Complexity Assumptions

Let  $\mathbb{BG} = (p, q, N, \mathbb{G}, \mathbb{G}_T, g, e(\cdot, \cdot)) \leftarrow \mathcal{G}_{\text{BG.Gen}}(1^\lambda)$  be a bilinear group with composite-order  $N = p \cdot q$  where  $p, q$  be two prime integers. We define a series of source and target group-based assumptions [BSW06, GKW19] that are required for proving security of our pairing-based schemes.

**Assumption 1** (Decisional 3-party Diffie-Hellman). (D3DH)[BSW06]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \mathbb{BG}, g_p, g_p \right) = \mathbf{b} : \begin{array}{l} \mathbb{BG}; g_p \leftarrow \mathbb{G}_p; g_q \leftarrow \mathbb{G}_q; a, b, c, r \leftarrow \mathbb{Z}_q; \\ T_0 = g_p^{abc}; T_1 = g_q^r; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

**Assumption 2** (Modified-1 Decisional 3-party Diffie-Hellman). (modified-1 D3DH). For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \mathbb{BG}, g_p, g_q, g_p^a, g_p^b, \right) = \mathbf{b} : \begin{array}{l} \mathbb{BG}; g_p \leftarrow \mathbb{G}_p; g_q \leftarrow \mathbb{G}_q; a, b, c, r \leftarrow \mathbb{Z}_p; \\ T_0 = g_p^{abc}; T_1 = g_p^r; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

We prove the generic security of this assumption in the following.

**Assumption 3** (Modified-2 Decisional 3-party Diffie-Hellman). (modified-2 D3DH) [BW06, GKW19]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$

such that

$$\Pr \left[ \mathcal{A} \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g_p, g_q, g_p^a, g_p^b, \\ g_p^c, g_p^{b^2}, T_b \end{array} \right) = \mathbf{b} : \begin{array}{c} \mathbb{B}\mathbb{G}; g_p \leftarrow \mathbb{G}_p; g_q \leftarrow \mathbb{G}_q; a, b, c, r \leftarrow \mathbb{Z}_p; \\ T_0 = g_p^{abc}; T_1 = g_p^r; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

This assumption is exactly identical with the modified D3DH assumption of Goyal et al. [GKW19].

**Assumption 4** ((Plain) Decisional Bilinear Diffie-Hellman). (DBDH)[Wat05]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g, g^a, g^b, \\ g^c, T_b \end{array} \right) = \mathbf{b} : \begin{array}{c} \mathbb{B}\mathbb{G}; g \leftarrow \mathbb{G}; a, b, c, r \leftarrow \mathbb{Z}_N; \\ T_0 = e(g, g)^{abc}; T_1 = e(g, g)^r; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

**Assumption 5** (Diffie-Hellman Sub-group Decisional). (DHSD) [GKW19]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g, h, g_p, g_q, g_p^a, h_q^a, \\ g^b, g_p^c, h^b, T_b \end{array} \right) = \mathbf{b} : \begin{array}{c} \mathbb{B}\mathbb{G}; g = g_p g_q; h = h_p h_q; \\ g_p, h_p \leftarrow \mathbb{G}_p; g_q, h_q \leftarrow \mathbb{G}_q; \\ a, b, c \leftarrow \mathbb{Z}_N; \\ T_0 \leftarrow \mathbb{G}_q; T_1 \leftarrow \mathbb{G}; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

**Assumption 6** (Bilinear Sub-group Decisional). (BSD) [BW06, BSW06, GKW19]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g, g_p, g_q, \\ e(T_b, g) \end{array} \right) = \mathbf{b} : \begin{array}{c} \mathbb{B}\mathbb{G}; g = g_p g_q; h = h_p h_q; \\ g_p \leftarrow \mathbb{G}_p; g_q \leftarrow \mathbb{G}_q; g \leftarrow \mathbb{G}; \\ T_0 \leftarrow \mathbb{G}_p; T_1 \leftarrow \mathbb{G}; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

**Assumption 7** (Relaxed 3-party Diffie-Hellman). (R3DH) [GKW19]. For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}(\cdot)$  such that

$$\Pr \left[ \mathcal{A} \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g, g_p, g_q, \\ g_q^a, \tilde{g}_p^a g_q^{a^2}, \tilde{g}_p^{ac}, \tilde{g}_p^c g_q^c, T_b \end{array} \right) = \mathbf{b} : \begin{array}{c} \mathbb{B}\mathbb{G}; g_p \leftarrow \mathbb{G}_p; g_q \leftarrow \mathbb{G}_q; \\ \tilde{a}, \tilde{c} \leftarrow \mathbb{Z}_p; a, c \leftarrow \mathbb{Z}_q; \\ T_0 = g_q^{a^2 c}; T_1 \leftarrow \mathbb{G}_q; \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \leq \frac{1}{2} + \text{negl}(\lambda)$$

**Generic Security of Assumption 2:** We used two D3DH types assumption which we call modified-1 D3DH and modified-2 D3DH. Note that, modified-2 is exactly the same assumption that Goyal et al. [GKW19] used in their bilinear group-based EIPL-BE security whereas our modified-1 D3DH assumption is a slight modification of modified-2 D3DH. It is easy to show that our modified-1 D3DH assumption is secure in the generic group model (GGM) using the techniques of Boneh et al. [BBG05]. We emphasize that all the well-known group-based assumptions (e.g., DDH, DBDH, D3DH etc.) are proven secure in GGM only. We do not find any immediate attack on our assumption against a non-generic adversary. To show the security of modified-2 D3DH in GGM, we use the following Lemma 1 and Definition 1 of [BBG05].

**Definition 1.** [BBG05] Let  $P, Q \in \mathbb{F}_p[X_1, \dots, X_n]^s$  be two  $s$ -tuples of  $n$ -variate polynomials over  $\mathbb{F}_p$ . Consider  $P = (p_1, p_2, \dots, p_s)$  and  $Q = (q_1, q_2, \dots, q_s)$  with  $p_1 = q_1 = 1$ . We

say that a polynomial  $f \in \mathbb{F}_p[X_1, \dots, X_n]$  is dependent on the sets  $(P, Q)$  if there exist  $s^2 + s$  constants  $\{a_{i,j}\}_{i,j=1}^s, \{b_k\}_{k=1}^s$  such that

$$f = \sum_{i,j=1}^s a_{i,j} p_i p_j + \sum_{k=1}^s b_k q_k$$

Note that,  $f$  is independent of  $(P, Q)$  if  $f$  is not dependent on  $(P, Q)$ . For a polynomial  $f \in \mathbb{F}_p[X_1, \dots, X_n]^s$ , let  $d_f$  denotes the total degree of  $f$ . For a set  $P \subseteq \mathbb{F}_p[X_1, \dots, X_n]^s$  consider  $d_P = \max\{d_f : f \in P\}$ .

**Lemma 1.** [BBG05] *Let  $P, Q \in \mathbb{F}_p[X_1, \dots, X_n]^s$  be two  $s$ -tuples of  $n$ -variate polynomials over  $\mathbb{F}_p$  and let  $f \in \mathbb{F}_p[X_1, \dots, X_n]$ . Let  $d = \max(2d_P, d_Q, d_f)$ . If  $f$  is independent of  $(P, Q)$  then any PPT adversary  $\mathcal{A}$  that has advantage  $1/2$  in solving the decision  $(P, Q, f)$ -Diffie-Hellman Problem in a generic bilinear group  $G$  must take time at least  $\Omega(\sqrt{p/d} - s)$ .*

**Theorem 2.** *Using  $n$ -variate polynomials of  $P, Q$  and  $f$  as per Definition 1, our modified-1 D3DH assumption 2 is secure in GGM as per Lemma 1.*

*Proof.* We use Lemma 1 of Boneh et al. [BBG05] to prove that our modified-1 D3DH assumption is secure in GGM. We consider the polynomials  $P, Q$  and  $f$  as:

$$P = (1, a, b, c, b^2, b^3, b^4, b^2c, b^3c) = (p_i)_i; \quad Q = (1); \quad f = (abc)$$

We see that  $P$  contains  $s = 9$  tuples. It is easy to see that there does not exist a set of constants  $\{x_{i,j}, z\} \subset \mathbb{Z}_p$  such that  $f = \sum_{i,j=1}^s x_{i,j} p_i p_j + z$ . Thus,  $f$  is independent of  $(P, Q, f)$  according to the Definition 1. The maximum total degrees of  $P, Q, f$  are  $d_P = 4, d_Q = 0, d_f = 3$  and hence  $d = \max(2d_P, d_Q, d_f) = 8$ . Therefore, by Lemma 1, any generic algorithm breaking the modified-1 D3DH assumption with advantage  $1/2$  must take time at least  $\Omega(\sqrt{p/8} - 9)$ .  $\square$

## 4 Definition: Embedded Identity Traceable IBIPFE

An EI-TIBIPFE for a message vector space  $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ , a predicate vector space  $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ , a user identity space  $\mathcal{ID} = \{\{0, 1\}^k : k \in \mathbb{N}\}$ , a group identity space  $\mathcal{GID} = \{\{0, 1\}^{k'} : k' \in \mathbb{N}\}$  consists of five PPT algorithms EI-TIBIPFE = (Setup, KeyGen, Enc, Dec, Trace). The details about these algorithms are given below.

- **Setup**( $1^\lambda, n, 1^k, 1^{k'}, 1^m$ )  $\rightarrow$  (msk, mpk, key): The trusted authority takes as input the security parameter  $\lambda$ , an index  $n$ , a *user identity space* parameter  $k$ , a *group identity space* parameter  $k'$ , a vector length parameter  $m$  and generates the master secret key **msk**, a master public key **mpk**, and a tracing key **key**.
- **KeyGen**(msk,  $i$ , id, gid,  $\mathbf{u}$ )  $\rightarrow$   $\mathbf{sk}_\mathbf{u}$ : On input the master secret key **msk**, user index  $i \in [n]$ , a user identity  $\text{id} \in \{0, 1\}^k$ , a group identity  $\text{gid} \in \{0, 1\}^{k'}$ , and a vector  $\mathbf{u} \in \mathbb{Z}^m$ , the trusted authority outputs a secret key  $\mathbf{sk}_\mathbf{u}$ .
- **Enc**(mpk, gid',  $\mathbf{v}$ )  $\rightarrow$   $\text{ct}_\mathbf{v}$ : The encryption algorithm takes input the master public key **mpk**, a group identity  $\text{gid}' \in \{0, 1\}^{k'}$ , a message vector  $\mathbf{v} \in \mathbb{Z}^m$ , and produces a ciphertext  $\text{ct}_\mathbf{v}$ .
- **Dec**( $\mathbf{sk}_\mathbf{u}, \text{ct}_\mathbf{v}$ )  $\rightarrow$   $\zeta / \perp$ : The decryption algorithm is run by taking input a secret key  $\mathbf{sk}_\mathbf{u}$  and a ciphertext  $\text{ct}_\mathbf{v}$ . It either outputs a decrypted value  $\zeta$  or a symbol  $\perp$  indicating



decryption failure.

- **Trace** $\mathcal{D}^{\mathbf{u}}$ (key,  $1^{\frac{1}{\epsilon(\lambda)}}$ , gid,  $\mathbf{u}$ ,  $\mathbf{v}^{(0)}$ ,  $\mathbf{v}^{(1)}$ )  $\rightarrow T$ : This algorithm has oracle access to a program  $\mathcal{D}_{\mathbf{u}}$  associated with the vector  $\mathbf{u}$ , it takes as input the tracing key key, a group identity gid, the predicate vector  $\mathbf{u}$ , two message vectors  $\mathbf{v}^{(0)}$ ,  $\mathbf{v}^{(1)}$  and outputs a set of identities  $T \subseteq \{0, 1\}^k$ . We call the tracing as public or private depending on whether key is equal to mpk or it is kept secret.

- **Correctness.** An EI-TIBIPFE = (Setup, KeyGen, Enc, Dec, Trace) scheme is said to be correct if for all  $\lambda, n, k, k', m \in \mathbb{N}$ ,  $i \in [n]$ ,  $\text{id} \in \{0, 1\}^k$ ,  $\text{gid} \in \{0, 1\}^{k'}$  and  $\mathbf{v}, \mathbf{u} \in \mathbb{Z}^m$ , there exists a *negligible* function  $\text{negl}$  satisfying  $\text{gid} = \text{gid}'$  such that the following holds,

$$\Pr \left[ \begin{array}{l} (\text{msk}, \text{mpk}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) \\ \text{Dec}(\text{sk}_{\mathbf{u}}, \text{ct}_{\mathbf{v}}) = \langle \mathbf{u}, \mathbf{v} \rangle : \quad \begin{array}{l} \text{sk}_{\mathbf{u}} \leftarrow \text{KeyGen}(\text{msk}, i, \text{id}, \text{gid}, \mathbf{u}) \\ \text{ct}_{\mathbf{v}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}, \mathbf{v}) \end{array} \end{array} \right] \geq 1 - \text{negl}(\lambda)$$

where the probability is taken over random coins of Setup, KeyGen and Enc of EI-TIBIPFE.

- **Security.** We define security notions of the EI-TIBIPFE as follows.

**Definition 2** (Adaptive Security of EI-TIBIPFE). An EI-TIBIPFE is said to satisfy adaptive indistinguishable-based (Adp-IND-CPA) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}$  such that the following holds:

$$\Pr[\text{Expt}_{\mathcal{A}, \text{Adp-IND-CPA}}^{\text{EI-TIBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{Adp-IND-CPA}}^{\text{EI-TIBIPFE}}(\lambda)$  is defined as follows:

1.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
2.  $(\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \text{gid}^*) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot)}(\text{mpk})$
3.  $\mathfrak{b} \leftarrow \{0, 1\}$
4.  $\text{ct}_{\mathbf{v}^{(\mathfrak{b})}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(\mathfrak{b})})$
5.  $\mathfrak{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot)}(\text{ct}_{\mathbf{v}^{(\mathfrak{b})}})$
6. Output 1 if  $\mathfrak{b} = \mathfrak{b}'$  else 0.

Figure 1:  $\text{Expt}_{\mathcal{A}, \text{Adp-IND-CPA}}^{\text{EI-TIBIPFE}}(\lambda)$

In the experiment, all the key queries of  $\mathcal{A}$  to the KeyGen oracle should be of form  $(i, \text{id}, \text{gid}, \mathbf{u})$  with  $i \in [n]$ ,  $\text{id} \in \{0, 1\}^k$ ,  $\text{gid} \in \{0, 1\}^{k'}$  and if  $\text{gid} = \text{gid}^*$  then  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  holds.

*Remark 1* (Selective Security). We can similarly define the Sel-IND-CPA security of EI-TIBIPFE (alike to Definition 9) where the adversary submits the challenge tuple  $(\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \text{gid}^*)$  before it receives the public parameters.

**Definition 3** (Security of Tracing). For any non-negligible function  $\epsilon(\cdot)$ , polynomial  $p(\cdot)$  and for all PPT adversary  $\mathcal{A}$ , consider the experiment  $\text{Expt}_{\mathcal{A}}^{\text{EI-TIBIPFE}}(\lambda)$  defined in Fig. 2. The tracing security of the scheme EI-TIBIPFE = (Setup, KeyGen, Enc, Dec, Trace) is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\mathcal{D}_u, \mathbf{u}, \mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \text{gid}^*) \leftarrow \mathcal{A}^{O(\cdot)}$
4.  $T \leftarrow \text{Trace}^{\mathcal{D}_u}(\text{key}, 1^{\frac{1}{\epsilon(\lambda)}}, \text{gid}^*, \mathbf{u}, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$

The oracle  $O(\cdot)$  has the  $\text{msk}$  hardwired in it and on query  $(i, \text{id}, \text{gid}, \mathbf{u})$  the oracle runs  $\text{KeyGen}(\text{msk}, i, \text{id}, \text{gid}, \mathbf{u})$  and sends the output iff the pair  $(i, \text{gid})$  was not queried before, otherwise it sends  $\perp$ . Let  $\mathcal{S}_{\mathcal{ID}}^u$  be the set of all users identities (id's) queried by  $\mathcal{A}$  associated with the vector  $\mathbf{u}$ . The above model defines the *adaptive* tracing security. In case of *selective* tracing,  $\mathcal{A}$  selects  $(\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \text{gid}^*)$  before setup and it outputs the decoder  $\mathcal{D}_u$  after it queries some secret keys.

Figure 2:  $\text{Expt}_{\mathcal{A}}^{\text{EI-TIBIPFE}}(\lambda)$

Based on the above experiment in Fig. 2, we define the following events and corresponding probabilities.

- Good-Decoder:  $\Pr[\mathcal{D}_u(\text{ct}_{\mathbf{v}^{(b)}}) = \mathbf{b} : \mathbf{b} \leftarrow \{0, 1\}, \text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(b)})] \geq \frac{1}{2} + \epsilon(\lambda)$ ,  $\Pr\text{-G-D}_{\mathcal{A}, \epsilon, p}(\lambda) = \Pr[\text{Good-Decoder} \wedge p(\lambda) \geq |\mathcal{S}_{\mathcal{ID}}^u|]$ .
- Cor-Tr:  $T \neq \phi \wedge T \subseteq \mathcal{S}_{\mathcal{ID}}^u$ ,  $\Pr\text{-Cor-Tr}_{\mathcal{A}, \epsilon, p}(\lambda) = \Pr[\text{Cor-Tr}]$ .
- Fal-Tr:  $T \not\subseteq \mathcal{S}_{\mathcal{ID}}^u$ ,  $\Pr\text{-Fal-Tr}_{\mathcal{A}, \epsilon, p}(\lambda) = \Pr[\text{Fal-Tr}]$ .

The EI-TIBIPFE is said to satisfy secure tracing if for any PPT adversary  $\mathcal{A}$ , polynomial  $q(\lambda)$  and non-negligible function  $\epsilon(\cdot)$ , there exists negligible functions  $\text{negl}_1, \text{negl}_2$  satisfying  $\epsilon(\lambda) > 1/q(\lambda)$  with the following conditions,

$$\Pr\text{-Fal-Tr}_{\mathcal{A}, \epsilon, p}(\lambda) \leq \text{negl}_1, \quad \Pr\text{-Cor-Tr}_{\mathcal{A}, \epsilon, p}(\lambda) \geq \Pr\text{-G-D}_{\mathcal{A}, \epsilon, p}(\lambda) - \text{negl}_2.$$

Note that the notion of EI-TIPFE is a particular case of EI-TIBIPFE where we simply ignore  $\text{gid}$  used in the syntax of EI-TIBIPFE.

*Remark 2* (Many Target and Fully Collusion Resistance Security). Our many target security allows the adversary to query polynomially many secret keys corresponding to many functions. In particular, the adversary can create a pirate box  $\mathcal{D}$  with many target functions  $\mathbf{u}_i$ . In that case, the tracing algorithm  $\text{Trace}^{\mathcal{D}}(\text{key}, 1^{\frac{1}{\epsilon(\lambda)}}, \text{gid}^*, \mathbf{u}_i, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$  can be run for each  $\mathbf{u}_i$  to identify the traitors associated with many target functions. In contrast, the tracing algorithm of [DPP20] is designed to trace traitors holding secret keys corresponding to a single target function since the adversary can only ask a secret key for a single vector  $\mathbf{u}$ . Moreover, we achieve the same *fully collusion resistance security* for our EI-TIBIPFE as defined by Boneh et al. [BSW06, BW06] for a normal traitor tracing scheme which does not impose any bound on the number of secret key queries submitted by the adversary (although the total number of users is fixed during setup). This is an important property to achieve for EI-TIBIPFE as a secret key for a user index  $i$  and a function  $\mathbf{u}_i$  is additionally associated with a user identity  $\text{id} \in \{0, 1\}^k$  and a group identity  $\text{gid} \in \{0, 1\}^{k'}$ , which means there could be an exponential number of keys corresponding to each user index. Our security model captures a powerful adversary that can query any polynomial number of secret keys corresponding to each indices in the system, ensuring fully collusion resistance security for EI-TIBIPFE.

## 5 Definition: Embedded Identity Private Linear IBIPFE

An EIPL-IBIPFE for a message vector space  $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ , a predicate vector space  $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ , a user identity space  $\mathcal{ID} = \{\{0, 1\}^k : k \in \mathbb{N}\}$  and a group identity space  $\mathcal{GID} = \{\{0, 1\}^{k'} : k' \in \mathbb{N}\}$  consists of five PPT algorithms  $\text{EIPL-IBIPFE} = (\text{Setup}, \text{KeyGen}, \text{Enc}, \text{SplEnc}, \text{Dec})$ . The details about these algorithms are given below.

- $\text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) \rightarrow (\text{msk}, \text{mpk}, \text{key})$ : The trusted authority takes as input the security parameter  $\lambda$ , the index space  $n$ , the user identity space parameter  $k$ , the group identity space parameter  $k'$ , a vector length parameter  $m$ , and outputs a master secret key  $\text{msk}$ , a master public key  $\text{mpk}$  and a key  $\text{key}$ .

- $\text{KeyGen}(\text{msk}, i, \text{id}, \text{gid}, \mathbf{u}) \rightarrow \text{sk}_\mathbf{u}$ : On input the master secret key  $\text{msk}$ , an index  $i \in [n]$ , an user identity  $\text{id} \in \{0, 1\}^k$ , a group identity  $\text{gid} \in \{0, 1\}^{k'}$  and a vector  $\mathbf{u} \in \mathbb{Z}^m$ , the trusted authority outputs a secret key  $\text{sk}_\mathbf{u}$ .

- $\text{Enc}(\text{mpk}, \text{gid}', \mathbf{v}) \rightarrow \text{ct}_\mathbf{v}$ : This algorithm is run by an encryptor by taking input as  $\text{mpk}$ , a group identity  $\text{gid}'$ , a message vector  $\mathbf{v} \in \mathbb{Z}^m$  and generates a ciphertext  $\text{ct}_\mathbf{v}$  associated to the vector  $\mathbf{v}$ .

- $\text{SplEnc}(\text{key}, \text{gid}', \mathbf{v}, (i, \ell, b)) \rightarrow \text{ct}_\mathbf{v}$ : It outputs a ciphertext  $\text{ct}_\mathbf{v}$  by taking input a key  $\text{key}$ , a group identity  $\text{gid}'$ , a message vector  $\mathbf{v} \in \mathbb{Z}^m$  and index-position-bit tuple  $(i, \ell, b) \in [n+1] \times ([k] \cup \{\perp\}) \times \{0, 1\}$ . If  $\text{key} = \text{mpk}$ , then EIPL-IBIPFE is called public key EIPL-IBIPFE, else it is called private key EIPL-IBIPFE.

- $\text{Dec}(\text{sk}_\mathbf{u}, \text{ct}_\mathbf{v}) \rightarrow \zeta/\perp$ : On input the secret key  $\text{sk}_\mathbf{u}$ , the ciphertext  $\text{ct}_\mathbf{v}$  decryptor outputs either a decrypted value  $\zeta$  or a symbol  $\perp$  indicating failure.

- **Correctness.** An EIPL-IBIPFE = (Setup, KeyGen, Enc, SplEnc, Dec) scheme is said to be correct if there exists *negligible* functions  $\text{negl}_1, \text{negl}_2$  such that for all  $\lambda, n, k, k', m \in \mathbb{N}$ ,  $\mathbf{v} \in \mathbb{Z}^m$ ,  $i \in [n+1], j \in [n]$ , user identity  $\text{id} \in \{0, 1\}^k$ , group identity  $\text{gid} \in \{0, 1\}^{k'}$ ,  $\ell \in ([k] \cup \{\perp\}), b \in \{0, 1\}$ , the following holds,

$$\Pr \left[ \begin{array}{l} \text{Dec}(\text{sk}_\mathbf{u}, \text{ct}_\mathbf{v}) = \langle \mathbf{u}, \mathbf{v} \rangle : \\ \text{msk}, \text{mpk}, \text{key} \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) \\ \text{sk}_\mathbf{u} \leftarrow \text{KeyGen}(\text{msk}, j, \text{id}, \text{gid}, \mathbf{u}) \\ \text{ct}_\mathbf{v} \leftarrow \text{Enc}(\text{mpk}, \text{gid}, \mathbf{v}) \end{array} \right] \geq 1 - \text{negl}_1(\lambda).$$

If  $(j \geq i+1) \vee (i, \ell) = (j, \perp) \vee (i, \text{id}_\ell) = (j, 1-b)$  then the following holds,

$$\Pr \left[ \begin{array}{l} \text{Dec}(\text{sk}_\mathbf{u}, \text{ct}_\mathbf{v}) = \langle \mathbf{u}, \mathbf{v} \rangle : \\ \text{msk}, \text{mpk}, \text{key} \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) \\ \text{sk}_\mathbf{u} \leftarrow \text{KeyGen}(\text{msk}, j, \text{id}, \text{gid}, \mathbf{u}) \\ \text{ct}_\mathbf{v} \leftarrow \text{SplEnc}(\text{key}, \text{gid}, \mathbf{v}, (i, \ell, b)) \end{array} \right] \geq 1 - \text{negl}_2(\lambda).$$

**Security.** We now formalize the IND-CPA security notions of EIPL-IBIPFE. Let  $q(\cdot)$  be a fixed polynomial. The security definitions for EIPL-IBIPFE is a generalization from the  $q$ -query security notions of EIPL-BE [GKW19] as given below.

**Definition 4** ( $q$ -query Normal-Hiding Security). The EIPL-IBIPFE scheme is said to satisfy  $q$ -query normal-hiding (NH) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr[\text{Expt}_{\mathcal{A}, \text{NH}}^{\text{EIPL-IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{NH}}^{\text{EIPL-IBIPFE}}(\lambda)$  is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\text{gid}^*, \mathbf{v}) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{mpk})$
4.  $\mathbf{b} \leftarrow \{0, 1\}$ ;
5.  $\text{ct}_v^{(0)} \leftarrow \text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}), \text{ct}_v^{(1)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (1, \perp, 0))$
6.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{ct}_v^{(\mathbf{b})})$
7. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 3:  $\text{Expt}_{\mathcal{A}, \text{NH}}^{\text{EIPL-IBIPFE}}(\lambda)$ 

In this experiment,  $\mathcal{A}$  can make at most  $q(\lambda)$  queries to the  $\text{SplEnc}$  oracle of the form  $(\text{gid}^*, \mathbf{v}, (1, \ell, \gamma))$  and all the secret key queries of  $\mathcal{A}$  to the  $\text{KeyGen}$  oracle should be of distinct indices. That is, if  $\mathcal{A}$  makes the queries  $(i_1, \text{id}_1, \text{gid}_1, \mathbf{u}_1), (i_2, \text{id}_2, \text{gid}_2, \mathbf{u}_2), \dots, (i_\kappa, \text{id}_\kappa, \text{gid}_\kappa, \mathbf{u}_\kappa)$ , then  $i_a \neq i_b$  when  $a \neq b$  for all  $a, b \in [\kappa]$ .

**Definition 5** ( $q$ -query Index-Hiding Security). The EIPL-IBIPFE scheme is said to satisfy  $q$ -query index-hiding (IH) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr[\text{Expt}_{\mathcal{A}, \text{IH}}^{\text{EIPL-IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{IH}}^{\text{EIPL-IBIPFE}}(\lambda)$  is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m, i^*) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\text{gid}^*, \mathbf{v}) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{mpk})$
4.  $\mathbf{b} \leftarrow \{0, 1\}$
5.  $\text{ct}_v^{(0)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \perp, 0)), \text{ct}_v^{(1)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^* + 1, \perp, 0))$
6.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{ct}_v^{(\mathbf{b})})$
7. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 4:  $\text{Expt}_{\mathcal{A}, \text{IH}}^{\text{EIPL-IBIPFE}}(\lambda)$ 

In this experiment,  $\mathcal{A}$  can make at most  $q(\lambda)$  queries to the  $\text{SplEnc}$  oracle of the form  $(\text{gid}^*, \mathbf{v}, (i, \ell, \gamma))$ , where the index  $i$  must be equal to either  $i^*$  or  $i^* + 1$ . All the secret key queries of  $\mathcal{A}$  to the  $\text{KeyGen}$  oracle should be of distinct indices and should not be of the form  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$ . That is, if  $\mathcal{A}$  makes the key queries  $(i_1, \text{id}_1, \text{gid}_1, \mathbf{u}_1), (i_2, \text{id}_2, \text{gid}_2, \mathbf{u}_2), \dots, (i_\kappa, \text{id}_\kappa, \text{gid}_\kappa, \mathbf{u}_\kappa)$ , then  $i_a \neq i_b$  when  $a \neq b$  for every  $a, b \in [\kappa]$  and  $i_a \neq i^*$  when  $\text{gid}_a = \text{gid}^*$  for every  $a \in [\kappa]$ .

**Definition 6** ( $q$ -query Lower Identity-Hiding Security). The EIPL-IBIPFE scheme is said to satisfy  $q$ -query lower identity-hiding ( $\text{LowIdH}$ ) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr[\text{Expt}_{\mathcal{A}, \text{LowIdH}}^{\text{EIPL-IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{LowIdH}}^{\text{EIPL-IBIPFE}}(\lambda)$  is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m, i^*, \ell^*, b^*) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\text{gid}^*, \mathbf{v}) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{mpk})$
4.  $\mathbf{b} \leftarrow \{0, 1\}$
5.  $\text{ct}_v^{(0)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \perp, 0)), \text{ct}_v^{(1)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \ell^*, b^*))$
6.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{ct}_v^{(\mathbf{b})})$
7. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 5:  $\text{Expt}_{\mathcal{A}, \text{LowIdH}}^{\text{EIPL-IBIPFE}}(\lambda)$ 

In this experiment,  $\mathcal{A}$  can make at most  $q(\lambda)$  queries to the  $\text{SplEnc}$  oracle of the form  $(\text{gid}^*, \mathbf{v}, (i, \ell, \gamma))$ , where the index  $i$  must be equal to  $i^*$ . All the secret key queries of  $\mathcal{A}$  to the  $\text{KeyGen}$  oracle should be of distinct indices and should not be of the form  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = b^*$ . That is, if  $\mathcal{A}$  makes the key queries  $(i_1, \text{id}_1, \text{gid}_1, \mathbf{u}_1), (i_2, \text{id}_2, \text{gid}_2, \mathbf{u}_2), \dots, (i_\kappa, \text{id}_\kappa, \text{gid}_\kappa, \mathbf{u}_\kappa)$ , then  $i_a \neq i_b$  when  $a \neq b$  for all  $a, b \in [\kappa]$  and  $i_a \neq i^*$  or  $(\text{id}_a)_{\ell^*} \neq b^*$  when  $\text{gid}_a = \text{gid}^*$  for all  $a \in [\kappa]$ .

**Definition 7** ( $q$ -query Upper Identity-Hiding Security). The EIPL-IBIPFE scheme is said to satisfy  $q$ -query upper identity-hiding ( $\text{UppldH}$ ) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr[\text{Expt}_{\mathcal{A}, \text{UppldH}}^{\text{EIPL-IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{UppldH}}^{\text{EIPL-IBIPFE}}(\lambda)$  is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m, i^*, \ell^*, b^*) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\text{gid}^*, \mathbf{v}) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{mpk})$
4.  $\mathbf{b} \leftarrow \{0, 1\}$
5.  $\text{ct}_v^{(0)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^* + 1, \perp, 0)), \text{ct}_v^{(1)} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}, (i^*, \ell^*, b^*))$
6.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{ct}_v^{(\mathbf{b})})$
7. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 6:  $\text{Expt}_{\mathcal{A}, \text{UppldH}}^{\text{EIPL-IBIPFE}}(\lambda)$ 

In this experiment,  $\mathcal{A}$  can make at most  $q(\lambda)$  queries to the  $\text{SplEnc}$  oracle of the form  $(\text{gid}^*, \mathbf{v}, (i, \ell, \gamma))$ , where the index  $i$  must be equal to either  $i^*$  or  $i^* + 1$ . All the secret key queries of  $\mathcal{A}$  to the  $\text{KeyGen}$  oracle should be of distinct indices and should not be of the form  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = 1 - b^*$ . That is, if  $\mathcal{A}$  makes the key queries  $(i_1, \text{id}_1, \text{gid}_1, \mathbf{u}_1), (i_2, \text{id}_2, \text{gid}_2, \mathbf{u}_2), \dots, (i_\kappa, \text{id}_\kappa, \text{gid}_\kappa, \mathbf{u}_\kappa)$ , then  $i_a \neq i_b$  when  $a \neq b$  for all  $a, b \in [\kappa]$  and  $i_a \neq i^*$  or  $(\text{id}_a)_{\ell^*} \neq 1 - b^*$  when  $\text{gid}_a = \text{gid}^*$  for all  $a \in [\kappa]$ .

**Definition 8** ( $q$ -query Message-Hiding Security). The EIPL-IBIPFE scheme is said to satisfy  $q$ -query message-hiding ( $\text{MH}$ ) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr[\text{Expt}_{\mathcal{A}, \text{MH}}^{\text{EIPL-IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

where the experiment  $\text{Expt}_{\mathcal{A}, \text{MH}}^{\text{EIPL-IBIPFE}}(\lambda)$  is defined as follows:

1.  $(1^n, 1^k, 1^{k'}, 1^m, i^*) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}, \text{key}) \leftarrow \text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$
3.  $(\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \text{gid}^*) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{mpk})$
4.  $\mathbf{b} \leftarrow \{0, 1\}$
5.  $\text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(b)}, (i^*, \perp, 0))$
6.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot, \cdot, \cdot), \text{SplEnc}(\text{key}, \cdot, \cdot, \cdot)}(\text{ct}_{\mathbf{v}^{(b)}})$
7. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 7:  $\text{Expt}_{\mathcal{A}, \text{MH}}^{\text{EIPL-IBIPFE}}(\lambda)$ 

In this experiment,  $\mathcal{A}$  can make at most  $q(\lambda)$  queries to the  $\text{SplEnc}$  oracle of the form  $(\text{gid}^*, \mathbf{v}, (i, \ell, \gamma))$ , where the index  $i$  must be equal to  $i^*$ . All the secret key queries of  $\mathcal{A}$  to the  $\text{KeyGen}$  oracle should be of distinct indices and the form of  $(i, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $i \geq i^*$  satisfying the condition  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  if  $\text{gid} = \text{gid}^*$ . That is, if  $\mathcal{A}$  makes the key queries  $(i_1, \text{id}_1, \text{gid}_1, \mathbf{u}_1), (i_2, \text{id}_2, \text{gid}_2, \mathbf{u}_2), \dots, (i_\kappa, \text{id}_\kappa, \text{gid}_\kappa, \mathbf{u}_\kappa)$ , then  $i_a \neq i_b$  when  $a \neq b$  for every  $a, b \in [\kappa]$ , and if  $i_a \geq i^*$  and  $\text{gid}_a = \text{gid}^*$  then  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  for any  $a \in [\kappa]$ .

*Remark 3* (Selective Security). Note that, the above security notions are described as the  $\text{Adp-IND-CPA}$  security model. In case of  $\text{Sel-IND-CPA}$  security model,  $\mathcal{A}$  is restricted to submit  $\text{gid}^*$  before setup for Definitions 4 - 7 whereas  $\mathcal{A}$  also submits the pair of message vectors  $(\mathbf{v}^{(0)}, \mathbf{v}^{(1)})$  along with  $\text{gid}^*$  before seeing any public parameters in Definition 8.

## 6 EI-TIBIPFE from EIPL-IBIPFE

Consider an EIPL-IBIPFE scheme  $\text{EIPL-IBIPFE} = \text{EIPL-IBIPFE}(\text{Setup}, \text{KeyGen}, \text{Enc}, \text{SplEnc}, \text{Dec})$  for a message vector space  $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ , a predicate vector space  $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ , a user identity space  $\mathcal{ID} = \{\{0, 1\}^k : k \in \mathbb{N}\}$  and a group identity space  $\mathcal{GITD} = \{\{0, 1\}^{k'} : k' \in \mathbb{N}\}$ . In the following, we provide our EI-TIBIPFE scheme with the same message vector space, user identity space, and group identity space. Depending on the special encryption algorithm of the underlying EIPL-IBIPFE scheme, this generic construction of our EI-TIBIPFE is called *public* or *private* EI-TIBIPFE.

- $\text{Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m) = \text{EIPL-IBIPFE.Setup}(1^\lambda, n, 1^k, 1^{k'}, 1^m)$ .
- $\text{KeyGen}(\text{msk}, i, \text{id}, \text{gid}, \mathbf{u}) = \text{EIPL-IBIPFE.KeyGen}(\text{msk}, i, \text{id}, \text{gid}, \mathbf{u})$ .
- $\text{Enc}(\text{mpk}, \text{gid}', \mathbf{v}) = \text{EIPL-IBIPFE.Enc}(\text{mpk}, \text{gid}', \mathbf{v})$ .
- $\text{Dec}(\text{sk}_\mathbf{u}, \text{ct}_\mathbf{v}) = \text{EIPL-IBIPFE.Dec}(\text{sk}_\mathbf{u}, \text{ct}_\mathbf{v})$ .
- $\text{Trace}^{\mathcal{D}_\mathbf{u}}(\text{key}, 1^{\frac{1}{\epsilon(\lambda)}}, \text{gid}, \mathbf{u}, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$ : Consider the two algorithms  $\text{Index-Trace}$  and  $\text{ID-Trace}$  defined in Fig. 8 and Fig. 9. First, the  $\text{Index-Trace}$  algorithm runs for each index  $i \in [n]$  and find a collection of indices set  $T^{\text{index}}$  such that the  $\text{Index-Trace}$  algorithm outputs 1 corresponds to these indices. Next, the  $\text{ID-Trace}$  algorithm runs on the index set  $T^{\text{index}}$ , and uses the decoder box to find the required identity of the particular indexed user. Next the tracing algorithm runs  $\text{ID-Trace}$  algorithm for all indices  $i \in T^{\text{index}}$ , and for each index  $i$  where the  $\text{ID-Trace}$  algorithm does not output  $\perp$ . The tracing algorithm adds the output of  $\text{ID-Trace}$  algorithm to the *identity-set* of traitors  $T$ .

1. Set  $T^{\text{index}} := \emptyset$ . For  $i = 1$  to  $n$ .
  - Compute  $(b, p, q) \leftarrow \text{Index-Trace}(\text{key}, 1^{\frac{1}{\epsilon(\lambda)}}, \text{gid}, \mathbf{u}, \mathbf{v}^{(0)}, \mathbf{v}^{(1)}, i)$ .
  - If  $b = 1$ , set  $T^{\text{index}} := T^{\text{index}} \cup \{(i, p, q)\}$ .
2. Set  $T := \emptyset$ . For  $(i, p, q) \in T^{\text{index}}$ .
  - Compute  $\text{id} \leftarrow \text{ID-Trace}(\text{key}, 1^{\frac{1}{\epsilon(\lambda)}}, \text{gid}, \mathbf{u}, \mathbf{v}^{(0)}, \mathbf{v}^{(1)}, (i, p, q))$ .

- Set  $T := T \cup \{\text{id}\}$ .
3. Return  $T$ .

## 6.1 Correctness

This follows from the correctness of the underlying EIPL-IBIPFE = (Setup, KeyGen, Enc, SplEnc, Dec) scheme. If  $\text{gid} = \text{gid}'$ , the decryptor correctly decrypts the ciphertext using a legitimate secret key; otherwise, it returns  $\perp$ .

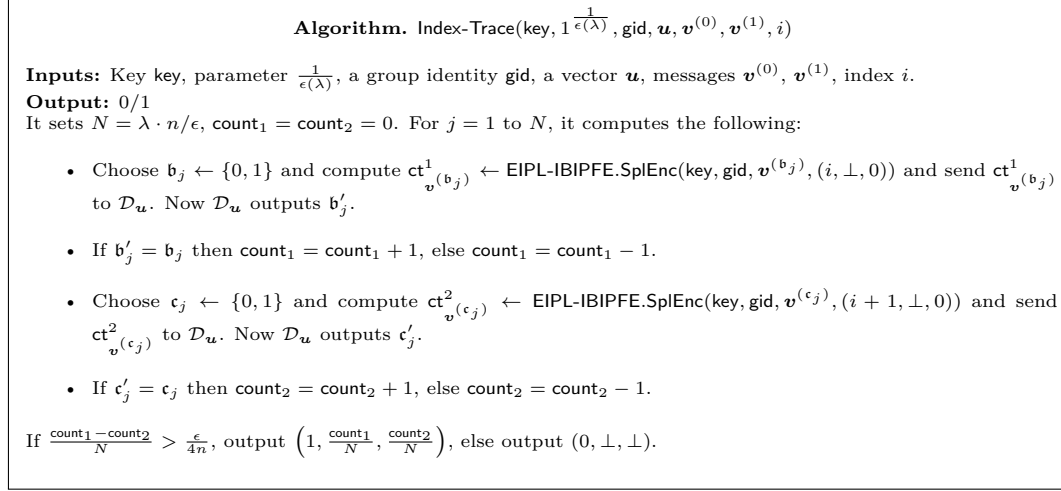


Figure 8: Index-Trace

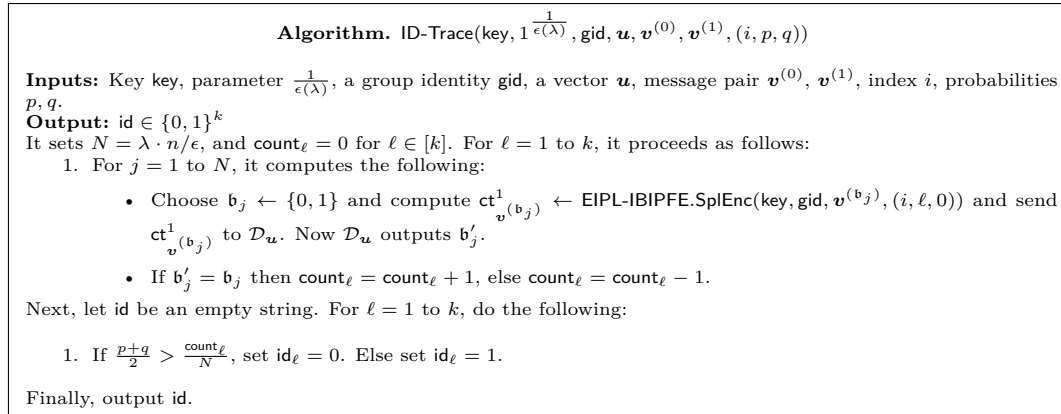


Figure 9: ID-Trace

**Security:** We discuss the security analysis in Appendix B.

## 7 EIPL-IBIPFE from Pairings

Let us assume  $\mathcal{G}_{\text{BG, Gen}}$  be a bilinear group generator of a composite-order group with order  $N = p \cdot q$  where  $p, q$  be two prime integers. Similar to all previous group-based IPFE constructions, we assume that the inner product value  $\langle \mathbf{u}, \mathbf{v} \rangle$  belongs to a polynomial range so that the decryptor can efficiently recover  $\langle \mathbf{u}, \mathbf{v} \rangle$  via a discrete log computation at

the end. In the following, we describe our EIPL-IBIPFE = (Setup, KeyGen, Enc, SplEnc, Dec) scheme using pairings.

- **Setup**( $1^\lambda, n, 1^k, 1^{k'}, 1^m$ ): The setup algorithm works as follows:
  - Set  $\tilde{n} = \lceil \sqrt{\frac{n}{k}} \rceil$  and  $\hat{n} = \lceil \frac{n}{\tilde{n}} \rceil$ .
  - Sample a bilinear group  $\mathbb{BG} = (p, q, N = pq, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot)) \leftarrow \mathcal{G}_{\text{BG.Gen}}(1^\lambda)$ .
  - Choose random generators  $g_p, h_p, f_p \in \mathbb{G}_p$  and  $g_q, h_q, f_q \in \mathbb{G}_q$  and sets  $g = g_p g_q, h = h_p h_q, f = f_p f_q \in \mathbb{G}$ .
  - The authority generates the following components:
    - **General components:** Sample  $\beta \leftarrow \mathbb{Z}_q$  and compute  $E_q = g_q^\beta$ .
    - **Row-specific components:** For all  $x \in [\hat{n}], j \in [m]$  sample  $r_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N, \hat{r} \leftarrow \mathbb{Z}_N$  and compute  $E_{q,x,j} = g_q^{\beta \hat{r} r_{x,j}}, F_{q,x,j} = h_q^{\beta \hat{r} r_{x,j}}, G_{q,x,j} = e(g_q, g_q)^{\beta \alpha_{x,j}}, W_{q,x,j} = e(f_q, g_q)^{\beta \psi_{x,j}}, E_{x,j} = g^{\hat{r} r_{x,j}}, F_{x,j} = h^{\hat{r} r_{x,j}}, G_{x,j} = e(g, g)^{\alpha_{x,j}}, W_{x,j} = e(f, g)^{\psi_{x,j}}$ .
    - **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$  sample  $c_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$  and compute  $H_{y,\ell,b} = g^{c_{y,\ell,b}}, \tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}, V_{\ell,b} = h^{\delta_{\ell,b}}$ .
    - **gid-specific components:** Sample  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  such that  $\vartheta' = \vartheta'_p \vartheta'_q \in \mathbb{G}$  and a  $k'$ -length vector  $\boldsymbol{\vartheta} = (\vartheta_i)_{i \in [k']} = (\vartheta_{p,i} \vartheta_{q,i})_{i \in [k']}$  whose each  $\vartheta_{p,i}, \vartheta_{q,i}$  are chosen at random from the subgroups  $\mathbb{G}_p, \mathbb{G}_q$  respectively. Let **gid** be a  $k'$ -bit string representing a group identity, where  $\text{gid}_i$  denotes the  $i$ -th bit of **gid** and  $\mathcal{V} \subseteq \{1, 2, \dots, k'\}$  be set of all  $i$  for which  $\text{gid}_i = 1$ . Consider two identity encoding functions  $H, H_q$  be defined as  $H(\text{gid}) = (\vartheta'_p \vartheta'_q) \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i}$  and  $H_q(\text{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  for  $\text{gid} \in \mathcal{GID}$ .
- The authority sets and outputs **mpk, msk, key** where

$$\text{mpk} = \text{key} = \left( \begin{array}{c} \mathbb{BG}, h, g, f, \vartheta', \vartheta'_q, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^\beta\}_{i \in [k]}, H, H_q, E_q, \\ \left\{ \begin{array}{c} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j}, \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\hat{n}] \times [m]}, \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}}, \{\tilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right),$$

$$\text{msk} = \left( \begin{array}{c} \mathbb{BG}, g, \hat{r}, \{r_{x,j}, \alpha_{x,j}, \psi_{x,j}\}_{(x,j) \in ([\hat{n}] \times [m])}, \\ \{c_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}} \end{array} \right).$$

- **KeyGen**(**msk, i, id, gid, u**): The key generation algorithm works as follows:
  - Consider  $(x, y) \in [\hat{n}] \times [\tilde{n}]$  be the unique row wise representation of index  $i$  (for any  $i \in [n]$ , the corresponding indices can be defined as  $y = i \bmod \tilde{n}$  and  $x = \lceil \frac{i}{\tilde{n}} \rceil$ ).
  - Choose a random  $\tilde{r} \leftarrow \mathbb{Z}_N$  and set  $r = \tilde{r} \cdot \hat{r} \bmod N$ .
  - Compute  $H(\text{gid}) = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i$  where  $\mathcal{V} = \{i : i\text{-th entry of gid is equals to 1}\}$ .
  - Output the secret key  $\text{sk}_u = (x, y, \text{id}, \text{gid}, K = (K_1, K_2, K_3))$  where

$$K_1 = g^{(\alpha_x, \mathbf{u})} \left( \prod_{\ell \in [k]} H_{y,\ell, \text{id}_\ell} \right)^{\hat{r} \langle r_x, \mathbf{u} \rangle}, K_2 = f^{\langle \psi_x, \mathbf{u} \rangle} \cdot H(\text{gid})^r \text{ and } K_3 = g^r.$$

- **Enc**(**mpk, gid', v**): The encryption algorithm is the same as special encryption algorithm (described below) when run with  $(i^*, \ell^*, b^*) = (1, \perp, 0)$ .
- **SplEnc**(**key, gid', v, (i\*, l\*, b\*)**): The special encryption algorithm executes the following steps:



- Let  $(x^*, y^*) \in [\tilde{n}] \times [\tilde{n}]$  be the unique row-wise representation of the index  $i^*$ .
- Sample the random exponents  $\tau, t \in \mathbb{Z}_N$  and compute:

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$ ,  $s_x, e_x, f_x, d_x \leftarrow \mathbb{Z}_N$  for all  $x \in [\tilde{n}]$  and categorize the components according  $x > x^*, x = x^*, x < x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{s_x}, \tilde{R}_{x,j} = F_{q,x,j}^{s_x \tau}$ .
2. *gid-specific component:*  $B_x = H_q(\text{gid}')^{\beta s_x t}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{q,x,j}^{s_x t} \cdot W_{q,x,j}^{s_x t}, A_x = E_q^{s_x t}$ .

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{x,j}^{s_x}, \tilde{R}_{x,j} = F_{x,j}^{s_x \tau}$ .
2. *gid-specific component:*  $B_x = H(\text{gid}')^{s_x t}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{x,j}^{s_x t} \cdot W_{x,j}^{s_x t}, A_x = g^{s_x t}$ .

**For  $x < x^*$ :**

1. *Linking components:*  $R_{x,j} = g^{s_x \sigma_j}, \tilde{R}_{x,j} = h^{s_x \tau \nu_j}$ .
2. *gid-specific component:*  $B_x = H(\text{gid}')^{d_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}, A_x = g^{e_x}$ .

- **Column-specific components:** Sample  $w_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$  and generate the components as follows:

**For  $(y > y^*) \vee (y = y^* \wedge (\ell, b) \neq (\ell^*, b^*))$ :**  $C_{y,\ell,b} = H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b} \tau}, \tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}}$ .

**For  $(y < y^*) \vee (y, \ell, b) = (y^*, \ell^*, b^*)$ :**  $C_{y,\ell,b} = H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b} \tau} \cdot V_{\ell,b}^{v_{y,\ell,b} \tau}, \tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}} \cdot \tilde{V}_{\ell,b}^{v_{y,\ell,b}}$ .

- Output the ciphertext  $\text{ct}_v$  associated to the vector  $v$  as

$$\text{ct}_v = \left( \left\{ R_{x,j}, \tilde{R}_{x,j}, B_x, I_{x,j}, A_x \right\}_{x \in [\tilde{n}], j \in [m]}, \left\{ C_{y,\ell,b}, \tilde{C}_{y,\ell,b} \right\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}} \right).$$

- $\text{Dec}(\text{sk}_u, \text{ct}_v)$ : The decryptor uses the secret key  $\text{sk}_u$  to decrypt the ciphertext  $\text{ct}_v$ . It either outputs  $\zeta = \log_{e(g_q, g_q)} \eta$  where

$$\eta = \prod_{j \in [m]} \frac{I_{x,j}^{u_j} \cdot e(R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell, \text{id}_\ell}^{u_j})}{e(\tilde{R}_{x,j}, \prod_{\ell \in [k]} \tilde{C}_{y,\ell, \text{id}_\ell}^{u_j})} \cdot \frac{e(K_3, B_x)}{e(K_1, A_x) \cdot e(K_2, A_x)}$$

or outputs  $\perp$ .

## 7.1 Correctness

Consider the secret key  $\text{sk}_u = (x, y, \text{id}, \text{gid}, K)$  corresponding to the index  $i = (x, y)$ , an user identity  $\text{id}$ , a group  $\text{gid}$ , and a predicate vector  $u$ . We know that

$$K = (K_1 = g^{\langle \alpha_x, u \rangle} \cdot \left( \prod_{\ell \in [k]} H_{y,\ell, \text{id}_\ell} \right)^{\widehat{r}(r_x, u)}, K_2 = f^{\langle \psi_x, u \rangle} H(\text{gid})^r, K_3 = g^r)$$

Here, the ciphertext  $\text{ct}_v$ , which is an encryption of a vector  $v$  and index-position-bit tuple  $(i^*, \ell^*, b^*)$ . It consists of  $\{R_{x,j}, \tilde{R}_{x,j}, A_x, B_x, I_{x,j}\}_{x,j}, \{C_{y,\ell,b}, \tilde{C}_{y,\ell,b}\}_{y,\ell,b}$ . Let  $i^* = (x^*, y^*)$ . From the definition of EIPL-IBIPFE, correctness holds or the decryption oracle gives the outputs  $\langle u, v \rangle$  if  $\text{gid} = \text{gid}'$  and  $(i \geq i^* + 1) \vee ((i^*, \ell^*) = (i, \perp)) \vee ((i^*, \text{id}_{\ell^*}) = (i, 1 - b^*))$ . Consider the index position  $i$ , an user identity  $\text{id}$  and the group identity  $\text{gid}$  which satisfies

the above mention constraints then from the representation of  $i$ , we can consider the following cases:

**Case 1:**  $x > x^*$ : In this case, we have all row components for all  $x \in [\tilde{n}], j \in [m]$  as  $R_{x,j} = E_{q,x,j}^{s_x}, \tilde{R}_{x,j} = F_{q,x,j}^{s_x \tau}, A_x = E_q^{s_x t}, B_x = H_q(\mathbf{gid})^{\beta s_x t}, I_{x,j} = e(g_q, g_q)^{v_j} \cdot G_{q,x,j}^{s_x t} \cdot W_{q,x,j}^{s_x t}$ . The decryption does not depend whether  $y > y^*$  or not. First, we consider  $(y > y^*) \vee ((y = y^*) \wedge (\ell, b) \neq (\ell^*, b^*))$  and simplify the following components.

$$\begin{aligned} \prod_{j \in [m]} e \left( R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j} \right) &= \prod_{j \in [m]} e \left( g_q^{\beta \widehat{r} r_{x,j} s_x}, \prod_{\ell \in [k]} g^{c_{y,\ell,\text{id}_\ell} t u_j} h^{w_{y,\ell,\text{id}_\ell} \tau u_j} \right) \\ &= e(g_q, g_q)^{\beta \widehat{r} s_x t \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} c_{y,\ell,\text{id}_\ell}} \cdot e(g_q, h_q)^{\beta \widehat{r} s_x \tau \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} w_{y,\ell,\text{id}_\ell}} \end{aligned} \quad (1)$$

$$\begin{aligned} \prod_{j \in [m]} I_{x,j}^{u_j} &= e(g_q, g_q)^{\sum_{j \in [m]} u_j v_j} \cdot e(g_q, g_q)^{\sum_{j \in [m]} \beta s_x t \alpha_{x,j} u_j} \cdot e(f_q, g_q)^{\sum_{j \in [m]} \beta \psi_{x,j} s_x t u_j} \\ &= e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle} \end{aligned} \quad (2)$$

$$\begin{aligned} \prod_{j \in [m]} e \left( \tilde{R}_{x,j}, \prod_{\ell \in [k]} \tilde{C}_{y,\ell,\text{id}_\ell}^{u_j} \right) &= \prod_{j \in [m]} e \left( h_q^{\beta \widehat{r} r_{x,j} s_x \tau}, \prod_{\ell \in [k]} g^{w_{y,\ell,\text{id}_\ell} u_j} \right) \\ &= e(h_q, g_q)^{\beta \widehat{r} s_x \tau \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} w_{y,\ell,\text{id}_\ell}} \end{aligned} \quad (3)$$

Also, we have

$$e(K_1, A_x) = e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(g_q, g_q)^{\beta \widehat{r} s_x t \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} c_{y,\ell,\text{id}_\ell}} \quad (4)$$

$$e(K_2, A_x) = e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle} \cdot e(H_q(\mathbf{gid}), g_q)^{\beta s_x t r} \quad (5)$$

$$e(K_3, B_x) = e(g^r, H_q(\mathbf{gid}))^{\beta s_x t} = e(g_q, H_q(\mathbf{gid}))^{\beta s_x t r} \quad (6)$$

Using Equations 1, 3 – 6 we compute  $\mathcal{R}_{\text{check}}$  and  $\mathcal{R}'_{\text{check}}$  as follows:

$$\mathcal{R}_{\text{check}} = \prod_{j \in [m]} \frac{e(R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j})}{e(\tilde{R}_{x,j}, \prod_{\ell \in [k]} \tilde{C}_{y,\ell,\text{id}_\ell}^{u_j}) \cdot e(K_1, A_x)} = (e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle})^{-1} \quad (7)$$

$$\mathcal{R}'_{\text{check}} = \frac{e(K_3, B_x)}{e(K_2, A_x)} = (e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle})^{-1} \quad (8)$$

From Equations 2,7,8, we compute

$$\begin{aligned} \eta &= \prod_{j \in [m]} \frac{I_{x,j}^{u_j} \cdot e(R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j})}{e(\tilde{R}_{x,j}, \prod_{\ell \in [k]} \tilde{C}_{y,\ell,\text{id}_\ell}^{u_j})} \cdot \frac{e(K_3, B_x)}{e(K_1, A_x) \cdot e(K_2, A_x)} = \left( \prod_{j \in [m]} I_{x,j}^{u_j} \cdot \mathcal{R}_{\text{check}} \cdot \mathcal{R}'_{\text{check}} \right) \\ &= e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \end{aligned}$$

Next, we consider  $(y < y^*) \vee ((y, \ell, b) = (y^*, \ell^*, b^*))$ . Then, we compute the following components.

$$\begin{aligned}
& \prod_{j \in [m]} e \left( R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j} \right) \\
&= \prod_{j \in [m]} e \left( g_q^{\widehat{r}r_{x,j} s_x}, \prod_{\ell \in [k]} g^{c_{y,\ell,\text{id}_\ell} t u_j} h^{w_{y,\ell,\text{id}_\ell} \tau u_j} h^{\tau \delta_{\ell,\text{id}_\ell} v_{y,\ell,\text{id}_\ell} u_j} \right) \\
&= e(g_q, g_q)^{\widehat{r} s_x t \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} c_{y,\ell,\text{id}_\ell}} \cdot e(g_q, h_q)^{\widehat{r} s_x \tau \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} (w_{y,\ell,\text{id}_\ell} + \delta_{\ell,\text{id}_\ell} v_{y,\ell,\text{id}_\ell})} \quad (9)
\end{aligned}$$

$$\begin{aligned}
& \prod_{j \in [m]} e \left( \widetilde{R}_{x,j}, \prod_{\ell \in [k]} \widetilde{C}_{y,\ell,\text{id}_\ell}^{u_j} \right) \\
&= \prod_{j \in [m]} e \left( h_q^{\widehat{r} r_{x,j} s_x \tau}, \prod_{\ell \in [k]} g^{w_{y,\ell,\text{id}_\ell} u_j} \cdot g^{\delta_{\ell,\text{id}_\ell} v_{y,\ell,\text{id}_\ell} u_j} \cdot g_p^{\gamma_{\ell,\text{id}_\ell} v_{y,\ell,\text{id}_\ell} u_j} \right) \\
&= e(h_q, g_q)^{\widehat{r} s_x \tau \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} (w_{y,\ell,\text{id}_\ell} + \delta_{\ell,\text{id}_\ell} v_{y,\ell,\text{id}_\ell})} \quad (10)
\end{aligned}$$

Also, in this case, we compute  $\mathcal{R}_{\text{check}}$  and  $\mathcal{R}'_{\text{check}}$  as follows:

$$\mathcal{R}_{\text{check}} = \prod_{j \in [m]} \frac{e(R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j})}{e(\widetilde{R}_{x,j}, \prod_{\ell \in [k]} \widetilde{C}_{y,\ell,\text{id}_\ell}^{u_j}) \cdot e(K_1, A_x)} = (e(g_q, g_q)^{\beta s_x t \langle \alpha_x, \mathbf{u} \rangle})^{-1} \quad (11)$$

$$\mathcal{R}'_{\text{check}} = \frac{e(K_3, B_x)}{e(K_2, A_x)} = (e(f_q, g_q)^{\beta s_x t \langle \psi_x, \mathbf{u} \rangle})^{-1} \quad (12)$$

So, correct decryption follows as previous.

**Case 2:** Otherwise: We have  $R_{x,j} = E_{x,j}^{s_x}$ ,  $\widetilde{R}_{x,j} = F_{x,j}^{s_x \tau}$ ,  $A_x = g^{s_x t}$ ,  $B_x = \text{H}(\text{gid})^{s_x t}$  and  $I_{x,j} = e(g_q, g_q)^{v_j} \cdot G_{x,j}^{s_x t} \cdot W_{x,j}^{s_x t}$ , then the correctness holds if  $(i^*, \ell^*) = (i, \perp) \vee (i^*, \text{id}_{\ell^*}) = (i, 1 - b^*)$  or we can write it as  $(x = x^*) \wedge ((y > y^*) \vee (y = y^* \wedge (\ell, b) \neq (\ell^*, b^*)))$ .

$$\begin{aligned}
& \prod_{j \in [m]} e \left( R_{x,j}, \prod_{\ell \in [k]} C_{y,\ell,\text{id}_\ell}^{u_j} \right) = \prod_{j \in [m]} e \left( g^{\widehat{r} r_{x,j} s_x}, \prod_{\ell \in [k]} g^{c_{y,\ell,\text{id}_\ell} t u_j} h^{w_{y,\ell,\text{id}_\ell} \tau u_j} \right) \\
&= e(g, g)^{\widehat{r} s_x t \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} c_{y,\ell,\text{id}_\ell}} \cdot e(g, h)^{\widehat{r} s_x \tau \langle \mathbf{u}, \mathbf{r}_x \rangle \sum_{\ell \in [k]} w_{y,\ell,\text{id}_\ell}} \quad (13)
\end{aligned}$$

$$\begin{aligned}
& \prod_{j \in [m]} e \left( \widetilde{R}_{x,j}, \prod_{\ell \in [k]} \widetilde{C}_{y,\ell,\text{id}_\ell}^{u_j} \right) = \prod_{j \in [m]} e \left( h^{\widehat{r} r_{x,j} s_x \tau}, \prod_{\ell \in [k]} g^{w_{y,\ell,\text{id}_\ell} u_j} \right) \\
&= e(h, g)^{\widehat{r} s_x \tau \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} w_{y,\ell,\text{id}_\ell}} \quad (14)
\end{aligned}$$

$$e(K_1, A_x) = e(g, g)^{s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(g, g)^{\widehat{r} s_x t \langle \mathbf{r}_x, \mathbf{u} \rangle \sum_{\ell \in [k]} c_{y,\ell,\text{id}_\ell}} \quad (15)$$

$$e(K_2, A_x) = e(f^{(\psi_x, \mathbf{u})} \cdot \text{H}(\text{gid})^r, g^{s_x t}) = e(f, g)^{s_x t \langle \psi_x, \mathbf{u} \rangle} \cdot e(\text{H}(\text{gid}), g)^{s_x t r} \quad (16)$$

$$e(K_3, B_x) = e(g^r, \text{H}(\text{gid})^{s_x t}) = e(g, \text{H}(\text{gid}))^{s_x t r} \quad (17)$$

$$\prod_{j \in [m]} I_{x,j}^{u_j} = e(g_q, g_q)^{\langle \mathbf{u}, \mathbf{v} \rangle} \cdot e(g, g)^{s_x t \langle \alpha_x, \mathbf{u} \rangle} \cdot e(f, g)^{s_x t \langle \psi_x, \mathbf{u} \rangle} \quad (18)$$

Using Equations 13 – 18 and following similar computations as in Case 1, the correctness of EIPL-IBIPFE holds.

## 7.2 Security Analysis

We prove the security of our EIPL-IBIPFE scheme below.

**Theorem 3.** *If the assumptions 1,2,3,5,6 and 7 hold over the bilinear group  $\mathbb{BG}$ , then our EIPL-IBIPFE is selectively secure as per Definitions 4 to 8.*

*Proof of Theorem 3.* We prove that our EIPL-IBIPFE satisfies all five security properties discussed in Section 5. We significantly modify the proof technique of [BW06, GKW19] to fit this into our scheme. Before going to the main idea of the proof technique, we would like to focus on the fact that our EIPL-IBIPFE consists of a public key special encryption algorithm. Thus, the adversary does not need to make special encryption queries to the EIPL-IBIPFE challenger. Therefore, the adversary only performs secret key queries to the challenger throughout the security game.

**Lemma 2.** *Our EIPL-IBIPFE satisfies selective normal-hiding security as per the Definition 4.*

*Proof.* Since the ciphertext distribution of normal encryption and special encryption for the index-position-bit tuple  $(1, \perp, 0)$  are the same, thus the definition of normal-hiding security follows from the scheme.  $\square$

**Lemma 3.** *If the assumptions 2,3,5,6 and 7 hold over the bilinear group  $\mathbb{BG}$ , then our EIPL-IBIPFE satisfies selective index-hiding security as per the Definition 5.*

*Proof.* As per the definition of the index-hiding game, we show that the adversary cannot distinguish between the special encryption of the index-position-bit tuple  $(i^*, \perp, 0)$  and  $(i^* + 1, \perp, 0)$ . Note that the adversary is not allowed to query for the secret keys corresponding to the index position  $i^* = (x^*, y^*)$  and the group identity  $\text{gid}^*$  at a time.

If  $y^* = \tilde{n}$ , we have  $i^* + 1 = (x^* + 1, 1)$  otherwise,  $i^* + 1 = (x^*, y^* + 1)$ . Similar to [BW06, GKW19], we consider two cases based on whether  $y = \tilde{n}$  or not. To prove this security, we consider the following two claims 7.2 and 7.2.

**Claim.** *For  $y^* < \tilde{n}$ , the special encryption to the index-position-bit tuple  $((x^*, y^*), \perp, 0)$  and  $((x^*, y^* + 1), \perp, 0)$  are indistinguishable.*

*Proof.* To prove the above claim, we consider  $2k + 1$  sequences of hybrid games as  $H_0$  and  $H_{\tilde{\ell}, \tilde{b}}$  where  $\tilde{\ell} \in [k]$  and  $\tilde{b} \in \{0, 1\}$ . The hybrid  $H_0$  corresponds to the index-hiding security game where the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^*), \perp, 0)$  and  $H_{\tilde{\ell}, \tilde{b}}$  is the same as  $H_0$  except that the column component  $C_{y^*, \ell, b}$  for  $(\ell, b) \in [\tilde{\ell} - 1] \times \{0, 1\}$  and for  $\ell = \tilde{\ell}, b = \tilde{b}$ , we uniformly choose from  $\mathbb{G}_p$ .

Table 2: Computing column components of the ciphertext in Hybrid  $H_{\tilde{\ell}, \tilde{b}}$

	$C_{y, \ell, b}$	$\tilde{C}_{y, \ell, b}$
$(y > y^*) \vee (y = y^* \wedge \ell > \tilde{\ell}) \vee (y = y^* \wedge \ell = \tilde{\ell} \wedge b > \tilde{b})$	$H_{y, \ell, b}^t \cdot h^{w_{y, \ell, b} \tau}$	$g^{w_{y, \ell, b}}$
$(y < y^*) \vee (y = y^* \wedge \ell < \tilde{\ell}) \vee (y = y^* \wedge \ell = \tilde{\ell} \wedge b \leq \tilde{b})$	$H_{y, \ell, b}^t \cdot h^{w_{y, \ell, b} \tau} V_{\ell, b}^{v_{y, \ell, b} \tau}$	$g^{w_{y, \ell, b}} \cdot \tilde{V}_{\ell, b}^{v_{y, \ell, b}}$

Here, the hybrid  $H_{k, 1}$  corresponds to the index-hiding game in which challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* + 1, \perp, 0) = ((x^*, y^* + 1), \perp, 0)$  and it is also required that the hybrid  $H_0$  and  $H_{1, 0}$  are indistinguishable. In the following, we show that the hybrid  $H_{\tilde{\ell}, \tilde{b}}$  and the hybrid  $H_{\tilde{\ell} + \tilde{b} - 1, (\tilde{b} + 1) \bmod 2}$  are indistinguishable. This same proof technique is used to show that all consecutive hybrids are indistinguishable. By combining all indistinguishability of hybrids, the claim 7.2 follows.

$H_{\tilde{\ell}, \tilde{b}} \approx H_{\tilde{\ell} + \tilde{b} - 1, (\tilde{b} + 1) \bmod 2}$ : Suppose on contrary, there exists a PPT adversary  $\mathcal{A}$  that can distinguish between the hybrid  $H_{\tilde{\ell}, \tilde{b}}$  and hybrid  $H_{\tilde{\ell} + \tilde{b} - 1, (\tilde{b} + 1) \bmod 2}$  with non-negligible

advantage  $\epsilon(\cdot)$ . We construct a PPT reduction algorithm  $\mathcal{B}$  which breaks the assumption 2 with the same advantages as follows.

Let the reduction algorithm  $\mathcal{B}$  first receives the modified-1 D3DH assumption 2 challenge instance from the challenger as

$$(\mathbb{B}\mathbb{G}, g_p, g_q, A = g_p^a, B = g_p^b, C = g_p^c, D = g_p^{b^2}, E = g_p^{b^2c}, F = g_p^{b^3}, G = g_p^{b^4}, H = g_p^{b^3c}, T)$$

where  $T$  is either  $g_p^{abc}$  or a random element in the subgroup  $\mathbb{G}_p$  of prime-order  $p$ . Next, it receives the challenge tuple  $(1^\lambda, 1^n, 1^k, 1^{k'}, 1^m, i^* = (x^*, y^*), \mathbf{gid}^*)$  from the adversary  $\mathcal{A}$  where  $y^* < \tilde{n}$ . Now,  $\mathcal{B}$  generates the master public key by using the modified-1 D3DH instance of assumption 2 and sends it to  $\mathcal{A}$ . Next, the adversary makes secret keys query for distinct indices  $i$  and the group identity  $\mathbf{gid}$  except for the tuple  $(i^*, \mathbf{id}, \mathbf{gid}^*, \mathbf{u})$  and sends the challenge message vector  $\mathbf{v}$  to the challenger. In the following, we show that how does  $\mathcal{B}$  generate the master public key and how to answer the queried secret keys and the challenge ciphertext using the challenge instance. Finally,  $\mathcal{A}$  outputs its guess, which  $\mathcal{B}$  uses to break the modified-1 D3DH assumption 2.

Since, this reduction plays over the subgroup  $\mathbb{G}_p$  with its challenger, thus it can choose any required elements from the subgroup  $\mathbb{G}_q$ . We implicitly set the exponents as  $r_{p,x^*,j} = b \cdot \tilde{r}_{p,x^*,j}$  and  $s_{p,x^*} = \tilde{s}_{p,x^*}/b, \hat{r}_p = b^2$  where the exponents  $\tilde{r}_{p,x^*,j}, \tilde{s}_{p,x^*}$  are chosen uniformly random from  $\mathbb{Z}_N$ . Also we set  $h_p = B = g_p^b, f_p = B^{d_1}, t_p = a \cdot b, c_{p,y^*,\tilde{\ell},\tilde{b}} = c \cdot \tilde{c}_{p,y^*,\tilde{\ell},\tilde{b}}$  for some uniformly chosen  $d_1 \leftarrow \mathbb{Z}_N, \tilde{c}_{p,y^*,\tilde{\ell},\tilde{b}} \leftarrow \mathbb{Z}_N$ . With these exponents,  $\mathcal{B}$  correctly simulates the master public key, secret keys and as well as the challenge group elements  $T$  that can be programmed in the challenge ciphertext components  $C_{y,\tilde{\ell},\tilde{b}}$ .

**Public key simulation.** The challenger  $\mathcal{B}$  chooses two random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  such that  $h_q = g_q^d, f_q = g_q^{d'}$  for some random exponents  $d, d' \in \mathbb{Z}_N$  and  $\mathcal{B}$  generates the following components:

- **General component:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = g_q^\beta$ .
- **Row-specific components:** For all  $x \in [\tilde{n}], j \in [m]$ , sample  $\tilde{r}_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N, \hat{r}_q \leftarrow \mathbb{Z}_N$  and compute  $E_{q,x,j} = g_q^{\beta \tilde{r}_{x,j}}, F_{q,x,j} = h_q^{\beta \hat{r}_q \tilde{r}_{x,j}}, G_{q,x,j} = e(g_q, g_q)^{\beta \alpha_{x,j}}, W_{q,x,j} = e(f_q, g_q)^{\beta \psi_{x,j}},$

$$E_{x,j} = \begin{cases} (Dg_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{if } x \neq x^*, \\ (Fg_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{otherwise.} \end{cases}, \quad F_{x,j} = \begin{cases} (Fh_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{if } x \neq x^*, \\ (Gh_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{otherwise.} \end{cases}$$

$$G_{x,j} = e(g, g)^{\alpha_{x,j}}, W_{x,j} = e(B, g_p)^{d_1 \psi_{x,j}} \cdot e(f_q, g_q)^{\psi_{x,j}}.$$

- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $\tilde{c}_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$  and compute  $\tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}, V_{\ell,b} = h^{\delta_{\ell,b}},$

$$H_{y,\ell,b} = \begin{cases} (Cg_q)^{\tilde{c}_{y,\ell,b}} & \text{if } (y, \ell, b) = (y^*, \tilde{\ell}, \tilde{b}) \\ (g_p g_q)^{\tilde{c}_{y,\ell,b}} & \text{otherwise.} \end{cases}$$

- **gid-specific components:** The challenger  $\mathcal{B}$  samples group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  chooses  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $H(\mathbf{gid}) = \vartheta'_p \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i, H_q(\mathbf{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  where  $\vartheta' = \vartheta'_p \vartheta'_q \in \mathbb{G}, \boldsymbol{\vartheta} = (\vartheta_i) \in \mathbb{G}^{k'}$ .

Using the challenge instance of modified-1 D3DH assumption 2,  $\mathcal{B}$  sets the master

public key as

$$\text{mpk} = \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g = g_p g_q, h = B g_q^d, f = B^{d_1} f_q, \vartheta', \vartheta_q^{\beta}, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^{\beta}\}_{i \in [k']}, H, H_q, E_q, \\ \left\{ \begin{array}{c} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\tilde{n}] \times [m]}, \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}}, \{\tilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right),$$

**Secret Key simulation.** To answer these queries, challenger returns the secret key  $\text{sk}_{\mathbf{u}}$  corresponding to the tuple  $(i = (x, y), \text{id}, \text{gid}, \mathbf{u})$  as follows: Note that the adversary is not allowed to secret key queries corresponding to the tuple  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  to the key generation oracle.

If  $\text{gid} = \text{gid}^*$ , then adversary cannot query for the secret key corresponding to the index position  $i^*$ . To generate the secret keys the challenger first computes  $H(\text{gid}^*) = (\vartheta_p' \vartheta_q' \prod_{i \in \mathcal{V}^*} \vartheta_{p,i} \vartheta_{q,i}) = g_p^{d_1^*} g_q^{d_2^*}$  where the random exponents  $d_1^*, d_2^* \leftarrow \mathbb{Z}_N$  and  $\mathcal{V}^*$  is associated with the non-zero indices of the challenge group identity  $\text{gid}^*$ . The challenger  $\mathcal{B}$  chooses a random value  $\tilde{r} \leftarrow \mathbb{Z}_N$  and sets  $r = \tilde{r} \cdot \tilde{r}$ . Then it simulates the secret keys as follows:

$$K_1 = \begin{cases} g^{\langle \alpha_x, \mathbf{u} \rangle} (D g_q^{\tilde{r}_q})^{\langle \tilde{r}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} & \text{if } x \neq x^*, y \neq y^*, \\ g^{\langle \alpha_x, \mathbf{u} \rangle} (F g_q^{\tilde{r}_q})^{\langle \tilde{r}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} & \text{if } x = x^*, (y \neq y^* \vee \text{id}_\ell \neq \tilde{b}), \\ g^{\langle \alpha_x, \mathbf{u} \rangle} (D g_q^{\tilde{r}_q})^{\langle \tilde{r}_x, \mathbf{u} \rangle} \sum_{\ell \neq \tilde{\ell}} \tilde{c}_{y,\ell, \text{id}_\ell} (E g_q^{\tilde{r}_q})^{\langle \tilde{r}_x, \mathbf{u} \rangle} \tilde{c}_{y,\tilde{\ell}, \text{id}_{\tilde{\ell}}} & \text{if } x \neq x^* \wedge (y, \text{id}_{\tilde{\ell}}) = (y^*, \tilde{b}), \end{cases}$$

$$K_2 = (B^{d_1} f_q)^{\langle \psi_x, \mathbf{u} \rangle} (D^{d_1^*} g_q^{d_2^* \tilde{r}_q})^{\tilde{r}}, \quad K_3 = (D g_q^{\tilde{r}_q})^{\tilde{r}}$$

For  $\text{gid} \neq \text{gid}^*$ , the adversary can query for the secret key corresponding to the index  $i^*$ . The challenger  $\mathcal{B}$  generates the secret key components as follows:

$$K_1 = \begin{cases} g^{\langle \alpha_x, \mathbf{u} \rangle} g_q^{\tilde{r}_q} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} (F \sum_{\ell \neq \tilde{\ell}} \tilde{c}_{y,\ell, \text{id}_\ell} H^{\tilde{c}_{y,\tilde{\ell}, \text{id}_{\tilde{\ell}}}})^{\langle \tilde{r}_x, \mathbf{u} \rangle} & \text{if } x = x^*, y = y^*, \text{id}_{\tilde{\ell}} = \tilde{b} \\ g^{\langle \alpha_x, \mathbf{u} \rangle} g_q^{\tilde{r}_q} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} F^{\langle \tilde{r}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} & \text{if } x = x^*, y = y^*, \text{id}_{\tilde{\ell}} \neq \tilde{b} \end{cases}$$

Without loss of generality, we assume that the adversary makes the maximum number of  $Q$  queries with the challenge group identity  $\text{gid}^*$  and challenge index  $i^*$ . Now, the simulator chooses an integer  $k'_1 \leftarrow [k']$ , sets an integer  $s = 10Q$ , a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$  and a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, the simulator also chooses a random value  $w' \leftarrow \mathbb{Z}_N$  and an uniformly random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_N^{k'}$ . All these values are kept secret to the simulator.

Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_\kappa\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  which corresponds to the collection of indices  $\mathcal{V}^*$ . Then set  $\sum_{i \in \mathcal{V}^*} z_i = k'_1 s - z'$  for uniformly chosen  $k'_1 \in [k']$ . Now, we define the function  $K(\text{gid})$  as

$$K(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}} z_i \equiv 0 \pmod{s} \\ 1, & \text{otherwise} \end{cases}$$

From the above definition of the function  $K$ , we can say that  $K(\text{gid}^*) = 0$  and for all  $\text{gid} \neq \text{gid}^*$  it becomes non-zero. Additionally, we set two functions as  $F(\text{gid}) = N - s k'_1 + z' + \sum_{i \in \mathcal{V}} z_i$  and  $J(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $\vartheta' = f^{N - k'_1 s + z'} \cdot g^{w'}$  and  $\vartheta_i = f^{z_i} g^{w_i} = g_p^{d_{p,i}} g_q^{d_{q,i}}$ . Now  $\mathcal{B}$  answers

secret key components  $K_2, K_3$  as follows:

$$\begin{aligned}
K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} \cdot D^{d_1^* \tilde{r}} g_q^{d_2^* \tilde{r} q \tilde{r}} \\
&= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} H(\text{gid})^r \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^r \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r'} \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} H(\text{gid})^{r'} \\
K_3 &= (D^{\tilde{r}} g_q^{\tilde{r} q \tilde{r}}) \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'}
\end{aligned}$$

We implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}$ . So from the construction of K function, we get  $K(\text{gid}) \neq 0$  for any key query corresponding to the group identity  $\text{gid} \neq \text{gid}^*$ . This implies that the function  $F(\text{gid}) \neq 0 \pmod N$  for the group identity (since we assume  $N > sk'_1$  for reasonable values of  $N, s$  and  $k'_1$ . We prove this in Lemma 9).

**Challenge ciphertext simulation.** The challenger  $\mathcal{B}$  samples the exponents  $\tau \in \mathbb{Z}_N, t_q \leftarrow \mathbb{Z}_N$  and for the challenge group identity  $\text{gid}^*$ ,  $\mathcal{B}$  computes  $H(\text{gid}^*) = (\vartheta'_p \vartheta'_q \prod_{i \in \mathcal{V}^*} \vartheta_{p,i} \vartheta_{q,i}) = g_p^{d_1^*} g_q^{d_2^*}$  for some  $d_1^*, d_2^* \in \mathbb{Z}_N$  and  $H_q(\text{gid}^*)^\beta = \vartheta'_q{}^\beta \prod_{i \in \mathcal{V}^*} \vartheta_{q,i}^\beta$ . Now,  $\mathcal{B}$  simulates the challenge ciphertext components as follows.

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x \leftarrow \mathbb{Z}_N, \tilde{s}_x \leftarrow \mathbb{Z}_N$ , for all  $x \in [\tilde{n}]$  and categorize the components according  $x > x^*, x = x^*, x < x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{\tilde{s}_x}, \tilde{R}_{x,j} = F_{q,x,j}^{\tilde{s}_x \tau}$ .
2. *gid-specific component:*  $B_x = H_q(\text{gid}^*)^{\beta \tilde{s}_x t_q}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{q,x,j}^{\tilde{s}_x t_q} \cdot W_{q,x,j}^{\tilde{s}_x t_q}, A_x = E_q^{\tilde{s}_x t_q}$ .

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = D^{\tilde{r}_{x,j} \tilde{s}_x} g_q^{\tilde{r}_{x,j} \tilde{s}_x}, \tilde{R}_{x,j} = F^{\tilde{r}_{x,j} \tilde{s}_x \tau} \cdot g_q^{\tilde{r}_{x,j} \tilde{s}_x \tau}$ .
2. *gid-specific component:*  $B_x = (A^{d_1^*} g_q^{d_2^* t_q})^{\tilde{s}_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot e(g, A g_q^{t_q})^{\alpha_{x,j} \tilde{s}_x} \cdot e(A, B)^{d_1 \psi_{x,j} \tilde{s}_x} \cdot e(f_q, g_q)^{\psi_{x,j} t_q \tilde{s}_x}, A_x = (A g_q^{t_q})^{\tilde{s}_x}$ .

**For  $x < x^*$ :**

1. *Linking components:*  $R_{x,j} = \tilde{g}^{\tilde{s}_x \sigma_j}, \tilde{R}_{x,j} = (B h_q)^{\tilde{s}_x \tau \nu_j}$ .
2. *gid-specific component:*  $B_x = (g_p^{d_1^*} g_q^{d_2^*})^{d_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}, A_x = g^{e_x}$ .

- **Column-specific components:** sample  $\tilde{w}_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$  and generate the components as follows:

**For  $(y > y^*) \vee (y = y^* \wedge \ell > \tilde{\ell}) \vee (y = y^* \wedge \ell = \tilde{\ell} \wedge b > \tilde{b})$ :**  $C_{y,\ell,b} = g_q^{\tilde{c}_{y,\ell,b} t_q} h^{\tilde{w}_{y,\ell,b} \tau}, \tilde{C}_{y,\ell,b} = A^{-\tilde{c}_{y,\ell,b} / \tau} \cdot g^{\tilde{w}_{y,\ell,b}}$ .

**For**  $y = y^* \wedge \ell = \tilde{\ell} \wedge b = \tilde{b}$ :  $C_{y,\ell,b} = g_q^{\tilde{c}_{y,\ell,b} t_q} \cdot h^{\tau \tilde{w}_{y,\ell,b}} \cdot T^{\tilde{c}_{y,\ell,b}}, \tilde{C}_{y,\ell,b} = g^{\tilde{w}_{y,\ell,b}}$

**For**  $(y < y^*) \vee (y = y^* \wedge \ell < \tilde{\ell}) \vee (y = y^* \wedge \ell = \tilde{\ell} \wedge b < \tilde{b})$ :  $C_{y,\ell,b} = g_q^{\tilde{c}_{y,\ell,b} t_q} \cdot h^{\tau \tilde{w}_{y,\ell,b}} \cdot g_p^{v_{y,\ell,b}}, \tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}}$

After generating all the ciphertext components, challenger sends these to the adversary  $\mathcal{A}$ , then  $\mathcal{A}$  guesses a bit  $b'$  and sends it to  $\mathcal{B}$ . It simply forwards it as the guess to the modified-1 D3DH challenger of assumption 2.

**Analysis of simulation.** If  $T = g_p^{abc}$ , then  $\mathcal{B}$  simulates the view of the hybrid is similar as  $H_{\tilde{\ell}, \tilde{b}}$  otherwise if  $T$  is random group elements from  $\mathbb{G}_p$  and the view of the hybrid is similar as  $H_{\tilde{\ell}+b-1, (\tilde{b}+1) \bmod 2}$ . Thus, if  $\mathcal{A}$  wins with the advantages  $\epsilon(\cdot)$  then  $\mathcal{B}$  breaks the modified-1 D3DH assumption 2 with the same advantages.  $\square$

**Claim.** If  $y^* = \tilde{n}$ , then the special encryption to the index-position-bit tuple  $(x^*, y^*, \perp, 0)$  and  $((x^* + 1, 1), \perp, 0)$  are indistinguishable.

*Proof.* To prove the above claim, we consider a sequence of hybrids games. In the following, we discuss about these hybrids.

**Hybrid 1.** This hybrid corresponds to the index-hiding game in which the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^*), \perp, 0)$  for  $y^* = \tilde{n}$ .

**Hybrid 2.** The hybrid is the same as hybrid 1 except that the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^* + 1), \perp, 0)$  for  $y^* = \tilde{n}$ . Note that, the special-encryption algorithm does not generally encrypt to the position  $(x^*, y^* + 1 = \tilde{n} + 1)$ , however the algorithm can be naturally extended to encrypt to such position.

**Hybrid 3.** Hybrid 3 is the same as the previous hybrid 2 except that the row component  $I_{x^*, j}$  as mentioned in the following Table 3.

Table 3: Computing row components for the ciphertext  $x \in [\tilde{n}], j \in [m]$

	$R_{x,j}$	$\tilde{R}_{x,j}$	$A_x$	$B_x$	$I_{x,j}$
$x > x^*$	$E_{q,x,j}^{s_x}$	$F_{q,x,j}^{s_x \tau}$	$E_q^{s_x t}$	$H_q(\text{gid}^*)^{\beta s_x t}$	$e(g_q, g_q)^{v_j} \cdot G_{q,x,j}^{s_x t} \cdot W_{q,x,j}^{s_x t}$
$x = x^*$	$E_{x,j}^{s_x}$	$F_{x,j}^{s_x \tau}$	$g^{s_x t}$	$H(\text{gid}^*)^{s_x t}$	$e(g_q, g_q)^{v_j} \cdot G_{x,j}^{s_x t} \cdot W_{x,j}^{s_x t} \cdot L$
$x < x^*$	$g^{s_x \sigma_j}$	$h^{s_x \tau \nu_j}$	$g^{e_x}$	$H(\text{gid}^*)^{d_x}$	$e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}$

where  $L = e(g_p, g)^z$  and  $z$  is randomly chosen from  $\mathbb{Z}_p$ .

**Hybrid 4.** Hybrid 4 is identical to the hybrid 3 except that the row component of the challenge ciphertext as the Table 3. Here we consider  $L = e(g, g)^z$  with  $z$  is a random exponent from  $\mathbb{Z}_N$ .

**Hybrid 5.** Hybrid 5 is the same as hybrid 4 except that row component in the challenge ciphertext as in Table 4 as mentioned below.

Table 4: Computing row components of the ciphertext for  $x \in [\tilde{n}], j \in [m]$

	$R_{x,j}$	$\tilde{R}_{x,j}$	$A_x$	$B_x$	$I_{x,j}$
$x > x^*$	$E_{q,x,j}^{s_x}$	$F_{q,x,j}^{s_x \tau}$	$E_q^{s_x t}$	$H_q(\text{gid}^*)^{\beta s_x t}$	$e(g_q, g_q)^{v_j} \cdot G_{q,x,j}^{s_x t} \cdot W_{q,x,j}^{s_x t}$
$x \leq x^*$	$g^{s_x \sigma_j}$	$h^{s_x \tau \nu_j}$	$g^{e_x}$	$H(\text{gid}^*)^{d_x}$	$e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}$



**Hybrid 6.** The hybrid 6 is similar to the hybrid 5 except that the column components to the index-position-bit tuple  $((x^*, y^* = 1), \ell^* = \perp, b^* = 0)$  of the challenge ciphertext.

**Hybrid 7.** The hybrid 7 corresponds to the index-hiding security game in which the challenge ciphertext is a special encryption to the index-position-bit tuple  $((x^* + 1, 1), \perp, 0)$  for  $y^* = \tilde{n}$ .

In the following, we prove that the adversary's advantage for all the consecutive hybrids is negligible in the security parameter which completes the proof of claim 7.2.

**Hybrid 1  $\approx$  Hybrid 2:** The indistinguishable proof of the hybrid 1 and hybrid 2 is identical to claim 7.2.

**Hybrid 2  $\approx$  Hybrid 3:** Suppose on the contrary, there exists a PPT adversary  $\mathcal{A}$  that distinguishes between the above two hybrids with the non-negligible advantages  $\epsilon(\lambda)$ . We construct a PPT reduction algorithm that breaks the modified-2 D3DH assumption 3 with the same non-negligible advantages as follows:

The reduction algorithm  $\mathcal{B}$  first receives the modified-2 D3DH assumption 3 challenge instance from its challenger as given below.

$$(\mathbb{B}\mathbb{G}, g_p, g_q, A = g_p^a, B = g_p^b, C = g_p^c, T = e(g_p, g)^z)$$

where  $z$  is either  $abc$  or a random element from  $\mathbb{Z}_N$ . In the setup phase, adversary receives the challenge tuple  $(1^\lambda, 1^n, 1^k, 1^m, i^* = (x^*, y^*), \text{gid}^*)$  from the adversary  $\mathcal{A}$  satisfying the condition  $y^* = \tilde{n}$ . Since the reduction game plays with its challenger in the subgroup  $\mathbb{G}_p$ , thus it can choose any elements from the subgroup  $\mathbb{G}_q$  by itself. Now  $\mathcal{B}$  generates the master public key using the given instance and sends it to the adversary  $\mathcal{A}$ . Then the adversary cannot make secret keys query corresponding the index  $i^*$  and  $\text{gid}^*$  at a time. In the following, we show how does  $\mathcal{B}$  simulate the master public key, secret keys and challenge ciphertext from the given instance. Finally,  $\mathcal{A}$  outputs its guess, which is used to break the modified-2 D3DH assumption 3. For  $x = x^*$ , our approach is to implicitly set the exponents  $r_{p,x^*,j} = b\tau_j$ ,  $\alpha_{p,x^*,j} = abk\hat{r}\tau_j$ ,  $t_p = c$  where  $\hat{r} \leftarrow \mathbb{Z}_N$  and  $\tau_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$ . Additionally, we implicitly set  $c_{y,\ell,b} = \tilde{c}_{p,y,\ell,b} - a$  for all  $(y, \ell, b) \in ([\tilde{n}] \times [k] \times \{0, 1\})$ . According to the exponents as given above, the challenger simulates the master public key, the secret keys and the challenge ciphertext components.

**Public key simulation.** The challenger  $\mathcal{B}$  chooses random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  and  $h_p, f_p \leftarrow \mathbb{G}_p$  such that  $h = h_p h_q$ ,  $f = f_p f_q$  and sets  $h = g^d$ ,  $f = g^{d_1}$  where some exponents  $d, d_1 \in \mathbb{Z}_N$ . Now, the challenger  $\mathcal{B}$  generates the following components:

- **General component:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = g_q^\beta$ .
- **Row-specific components:** For all  $x \in [\tilde{n}], j \in [m]$ , sample  $\tilde{r}_{x,j}, \tilde{\alpha}_{x,j}, \tilde{\psi}_{x,j} \leftarrow \mathbb{Z}_N$  and compute  $E_{q,x,j} = g_q^{\beta\tilde{r}_{x,j}}$ ,  $F_{q,x,j} = h_q^{\beta\tilde{r}_{x,j}}$ ,  $G_{q,x,j} = e(g_q, g_q)^{\beta\tilde{\alpha}_{x,j}}$ ,  $W_{q,x,j} = e(g_q, g_q)^{d_1\tilde{\psi}_{x,j}}$ ,
 
$$E_{x,j} = \begin{cases} (g_p g_q)^{\tilde{r}_{x,j}} & \text{if } x \neq x^* \\ (B^{\tau_j} g_q^{\tilde{r}_{x,j}})^{\hat{r}} & \text{otherwise} \end{cases}, F_{x,j} = \begin{cases} (g_p g_q)^{d\tilde{r}_{x,j}} & \text{if } x \neq x^* \\ (B^{\tau_j} g_q^{d\tilde{r}_{x,j}})^{\hat{r}} & \text{otherwise} \end{cases},$$

$$G_{x,j} = \begin{cases} e(g_p g_q, g_p g_q)^{\tilde{\alpha}_{x,j}} & \text{if } x \neq x^* \\ e(A, B)^{k\hat{r}\tau_j} e(g_q, g_q)^{\tilde{\alpha}_{x,j}} & \text{otherwise} \end{cases}, W_{x,j} = e(g, g)^{d_1\tilde{\psi}_{x,j}}.$$
- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $\tilde{c}_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$  and compute  $H_{y,\ell,b} = A^{-1} g^{c_{y,\ell,b}}$ ,  $\tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}$ ,  $V_{\ell,b} = h^{\delta_{\ell,b}}$ .

- **gid-specific components:** The challenger  $\mathcal{B}$  samples group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  chooses  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\mathbf{H}(\text{gid}) = \vartheta'_p \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i$ ,  $\mathbf{H}_q(\text{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  where  $\vartheta' = \vartheta'_p \vartheta'_q \in \mathbb{G}$ ,  $\boldsymbol{\vartheta} = (\vartheta_i) \in \mathbb{G}^{k'}$ .

Using the challenge instance of modified-2 D3DH assumption 3,  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \mathbb{B}\mathbb{G}, g = g_p g_q, h = g^d, f = g^{d_1}, \vartheta', \vartheta'_q, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^\beta\}_{i \in [k']}, \mathbf{H}, \mathbf{H}_q, E_q, \right. \\ \left. \begin{array}{c} \left\{ E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \right\} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right)_{(x,j) \in [\widehat{n}] \times [m]}, \\ \left\{ H_{y,\ell,b} \right\}_{(y,\ell,b) \in [\widehat{n}] \times [k] \times \{0,1\}}, \left\{ \tilde{V}_{\ell,b}, V_{\ell,b} \right\}_{(\ell,b) \in [k] \times \{0,1\}}$$

**Secret key simulation.** The challenger  $\mathcal{B}$  generates the secret key  $\text{sk}_{\mathbf{u}}$  corresponding to the adversary's query tuple  $(i = (x, y), \text{id}, \text{gid}, \mathbf{u})$  as below.

First consider  $\text{gid} = \text{gid}^*$ , then the adversary can not query for the secret key associated with the index position  $i^*$ .

$$K_1 = \begin{cases} g^{\langle \tilde{\alpha}_x, \mathbf{u} \rangle + \widehat{r} \langle \tilde{\mathbf{r}}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} A^{-\widehat{r} \langle \tilde{\mathbf{r}}_x, \mathbf{u} \rangle} & \text{if } x \neq x^* \\ g_q^{\langle \tilde{\alpha}_x, \mathbf{u} \rangle + \widehat{r} \langle \tilde{\mathbf{r}}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} B^{\widehat{r} \langle \mathbf{r}, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{y,\ell, \text{id}_\ell} & \text{otherwise} \end{cases},$$

$$K_2 = g^{d_1 \langle \tilde{\psi}_x, \mathbf{u} \rangle} \mathbf{H}(\text{gid})^r, \quad K_3 = g^r$$

where  $r = \widehat{r} \cdot \tilde{r}$  and  $\tilde{r}$  is randomly chosen from  $\mathbb{Z}_N$ . Note that, the adversary  $\mathcal{A}$  is not allowed to query for the secret key corresponding to the tuple  $(i^* = (x^*, y^*), \text{id}, \text{gid}^*, \mathbf{u})$ .

If  $\text{gid} \neq \text{gid}^*$ , then the adversary can query for the index position  $i^*$ . Then the challenger generates the corresponding secret keys as below

$$K_1 = g_q^{\langle \tilde{\alpha}_x, \mathbf{u} \rangle + \widehat{r} \langle \tilde{\mathbf{r}}_x, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{n,\ell, \text{id}_\ell} B^{\widehat{r} \langle \mathbf{r}, \mathbf{u} \rangle} \sum_{\ell \in [k]} \tilde{c}_{n,\ell, \text{id}_\ell} \text{ if } x = x^*$$

As the previous case, we assume that the adversary makes the maximum number of  $Q$  queries, the challenge group identity  $\text{gid}^*$  and challenge index  $i^*$ . Now, the simulator chooses an integer  $k'_1 \leftarrow [k']$ , sets an integer  $s = 10Q$ , a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$  and a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, the simulator chooses a random value  $w' \leftarrow \mathbb{Z}_N$  and an uniformly random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_N^{k'}$ . All these values are kept secret to the  $\mathcal{B}$ .

Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_\kappa\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  which correspond to the collection of indices  $\mathcal{V}^*$ . Sets  $\sum_{i \in \mathcal{V}^*} z_i = k'_1 s - z'$  for uniformly chosen  $k'_1 \in [k']$ . Now, we define the function  $\mathbf{K}(\text{gid})$  as

$$\mathbf{K}(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}} z_i \equiv 0 \pmod{s} \\ 1, & \text{otherwise} \end{cases}$$

So, from the above definition of the function  $\mathbf{K}$ , we can say  $\mathbf{K}(\text{gid}^*) = 0$  and for all other  $\text{gid} \neq \text{gid}^*$ , it becomes non-zero. Additionally, we set two functions as  $\mathbf{F}(\text{gid}) = N - sk'_1 + z' + \sum_{i \in \mathcal{V}} z_i$  and  $\mathbf{J}(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $\vartheta' = f^{N - k'_1 s + z'} \cdot g^{w'} = g_p^{d'_p} g_q^{d'_q}$  and  $\vartheta_i = f^{z_i} g^{w_i} = g_p^{d_{p,i}} g_q^{d_{q,i}}$ . Now  $\mathcal{B}$  answers

remaining secret key components as

$$\begin{aligned}
K_2 &= g^{-\langle \tilde{\psi}_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} \mathbf{H}(\text{gid})^r \\
&= f^{\langle \tilde{\psi}_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{-\frac{\langle \tilde{\psi}_x, \mathbf{u} \rangle}{F(\text{gid})}} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^r \\
&= f^{\langle \tilde{\psi}_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r - \frac{\langle \tilde{\psi}_x, \mathbf{u} \rangle}{F(\text{gid})}} \\
&= f^{\langle \tilde{\psi}_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r'} \\
&= f^{\langle \tilde{\psi}_x, \mathbf{u} \rangle} \mathbf{H}(\text{gid})^{r'} \\
K_3 &= g^r \cdot g^{-\frac{\langle \tilde{\psi}_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \tilde{\psi}_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'}
\end{aligned}$$

We implicitly set  $r' = r - \frac{\langle \tilde{\psi}_x, \mathbf{u} \rangle}{F(\text{gid})}$ . So from the construction of  $\mathbf{K}$  function, we get  $\mathbf{K}(\text{gid}) \neq 0$  for any key query corresponding to the group identity  $\text{gid} \neq \text{gid}^*$ . This implies that the function  $F(\text{gid}) \neq 0 \pmod N$  for any such group identity (as we assume  $N > sk'_1$  for reasonable values of  $N, s$  and  $k'_1$ , see Lemma 9).

**Challenge ciphertext simulation.** The challenger  $\mathcal{B}$  can compute all the column components corresponding to  $(y, \ell, b) \in [\tilde{n}] \times [k] \times \{0, 1\}$  on its own, since  $\mathbb{G}_p$  subgroup components are random in  $C_{y, \ell, b}, \tilde{C}_{y, \ell, b}$  terms and for computing remaining terms over the subgroup  $\mathbb{G}_q$ , the required exponents are already known to the challenger  $\mathcal{B}$ . For  $x < x^*$ , all the row components  $R_{x,j}, \tilde{R}_{x,j}, A_x, B_x, I_{x,j}$  are chosen randomly, but for  $x > x^*$ , all the row components are formed over the subgroup  $\mathbb{G}_q$  which it knows. For the challenge group identity  $\text{gid}^*$ , the challenger computes  $\mathbf{H}(\text{gid}^*) = (\vartheta'_p \vartheta'_q) \prod_{j \in \mathcal{V}^*} (\vartheta_{p,j} \vartheta_{q,j}) = g_p^{d_1^*} g_q^{d_2^*}$  for some  $d_1^*, d_2^* \in \mathbb{Z}_N$  and samples  $\tilde{s}_{x^*}, \tau, t_q \leftarrow \mathbb{Z}_N$ . Now, the challenger  $\mathcal{B}$  simulates the challenge ciphertext for  $x = x^*$  as follows:

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = (B^{\tau_j} g_q^{\tilde{r}_{x,j}})^{\tilde{s}_x}, \tilde{R}_{x,j} = (B^{\tau_j} g_q^{\tilde{r}_{x,j}})^{\tilde{d}r_{s_x} \tau}$ .
2. *gid-specific component:*  $B_x = (C^{d_1^*} g_q^{t_q d_2^*})^{\tilde{s}_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{v_j} \cdot e(g_q, g_q)^{\alpha_{x,j} \tilde{s}_x t_q} \cdot T^{\tilde{r} k \tilde{s}_x \tau_j} \cdot e(f, C g_q^{t_q})^{\tilde{\psi}_{x,j} \tilde{s}_x},$   
 $A_x = (C g_q^{t_q})^{\tilde{s}_x}$ .

Finally,  $\mathcal{B}$  gets the guess bit  $b'$  from  $\mathcal{A}$  and it simply forwards it to the modified-2 D3DH assumption 3 challenger.

**Analysis of simulation.** If  $T = e(g_p, g)^{abc}$ , then  $\mathcal{B}$  simulates the view of hybrid 2 else if  $T = e(g_p, g)^z$  for any random  $z$  from  $\mathbb{Z}_N$ , adversary's view same as hybrid 3. Therefore, if  $\mathcal{A}$  wins the game with advantages  $\epsilon(\cdot)$ , then  $\mathcal{B}$  breaks the modified-2 D3DH assumption 3 with the same advantages.

**Hybrid 3  $\approx$  Hybrid 4:** To show the indistinguishability of two hybrids 3 and 4, we use a similar proof technique of [BW06, GKW19]. Here, we discuss the underlying approaches. Let us consider that  $\mathcal{B}$  receives the Bilinear Subgroup Decisional (BSD) assumption 6 challenge instance from the challenger consisting the bilinear group  $\mathbb{B}\mathbb{G}, e(T, g)$  where  $T$  is either a random element from the subgroup  $\mathbb{G}_p$  or a uniform element from the group  $\mathbb{G}$ . Then  $\mathcal{B}$  computes all the components for the master public key  $\text{mpk}$  honestly and forwarded it to the adversary. After seeing  $\text{mpk}$ , adversary can query for the secret key to the key generation oracle. Finally,  $\mathcal{B}$  computes all the challenge ciphertext components honestly except that the value  $I_{x^*,j} = e(g_q, g_q)^{v_j} \cdot G_{x^*,j}^{s_{x^*} t} \cdot e(g, T) \cdot e(f, g)^{\psi_{x^*,j} s_{x^*} t}$  where  $T$  is taken from BSD challenge of assumption 2. If  $T \leftarrow \mathbb{G}_p$ , then simulator's view is the

same as hybrid 3 otherwise, if  $T \leftarrow \mathbb{G}$  then  $\mathcal{B}$  perfectly simulates as hybrid 4. Therefore, if  $\mathcal{A}$  wins with the advantage  $\epsilon(\cdot)$ , then  $\mathcal{B}$  breaks the assumption 6 with the same advantage  $\epsilon(\cdot)$ .

**Hybrid 4  $\approx$  Hybrid 5:** Suppose on the contrary, there exists PPT adversary  $\mathcal{A}$  that distinguish between the hybrid 4 and hybrid 5 with the non-negligible advantage  $\epsilon(\cdot)$ . Then, we construct a PPT reduction algorithm which breaks the R3DH assumption 7 with the same non-negligible advantages.

Let the reduction algorithm  $\mathcal{B}$  first receives the challenges of R3DH assumption 7 from the challenger as

$$(\mathbb{B}\mathbb{G}, \mathfrak{g}_p \in \mathbb{G}_p, \mathfrak{g}_q \in \mathbb{G}_q, A = \mathfrak{g}_q^a, B = \mathfrak{g}_p^{\tilde{a}} \cdot \mathfrak{g}_q^{a^2}, C = \mathfrak{g}_p^{\tilde{c}} \cdot \mathfrak{g}_q^c, D = \mathfrak{g}_p^{\tilde{a}\tilde{c}}, T)$$

where  $T$  is either  $\mathfrak{g}_q^{a^2c}$  or a random element from the subgroup  $\mathbb{G}_q$ . Next, the challenger  $\mathcal{B}$  receives the challenge tuple  $(1^n, 1^k, 1^{k'}, 1^m, i^* = (x^*, y^*), \text{gid}^*)$  from  $\mathcal{A}$ . Then,  $\mathcal{B}$  generates the public keys and sends it to the adversary. After getting the public parameters,  $\mathcal{A}$  can secret key query to the key generation oracle corresponding for the tuple  $(i = (x, y), \text{id}, \text{gid}, \mathbf{u})$  except for the tuple  $(i^* = (x^*, y^*), \text{id}, \text{gid}^*, \mathbf{u})$ . Next the adversary uniformly chooses a challenge message vector  $\mathbf{v}$  and sends it to  $\mathcal{B}$ . To answer the challenge ciphertext,  $\mathcal{B}$  randomly chooses a bit  $\mathbf{b} \in \{0, 1\}$  and generates the challenge ciphertext  $\text{ct}_{\mathbf{v}}^{(\mathbf{b})}$ . In the following, we describe how does the challenger simulate the master public key, the secret key and the challenge ciphertext using the assumption 7 instance. Finally, the adversary  $\mathcal{A}$  outputs a guess bit which breaks the assumption 7. As the reduction plays the game with the challenger in the subgroup  $\mathbb{G}_q$ , so it chooses all the components from the subgroup  $\mathbb{G}_p$  by itself. Although, in the challenge instance of  $B, C$  some parts belong to the subgroup  $\mathbb{G}_p$  but their exponents depends on  $\tilde{a}$  and  $\tilde{c}$  terms. In the following, we implicitly set the exponents as

$$\begin{aligned} g_p = \mathfrak{g}_p, g_q = A, & \quad r_{q,x^*,j} = \tilde{r}_{q,x^*,j}/a, & \quad r_{p,x^*,j} = \tilde{r}_{p,x^*,j}, \\ s_{q,x^*} = c, & \quad s_{p,x^*} = \tilde{c}, & \quad t_q = a, t_p = \tilde{a}, \\ & \quad \text{for all } x \in [\hat{n}] - \{x^*\}, s_x = \tilde{s}_x/a \end{aligned}$$

where  $\tilde{r}_{p,x^*,j}, \tilde{r}_{q,x^*,j} \leftarrow \mathbb{Z}_N$  and for all  $x \in [\hat{n}] - \{x^*\}, \tilde{s}_x \leftarrow \mathbb{Z}_N$ . Additionally, the reduction algorithm samples the exponents uniformly random from  $\mathbb{Z}_N$ . Note that, the reduction algorithm does not know the factorization so at any point, we do not sample these exponents from  $\mathbb{G}_p$  and  $\mathbb{G}_q$  separately, but instead of sample any exponents directly from  $\mathbb{Z}_N$  and make sure that the distributions are not affected.

**Public key simulation.** The challenger  $\mathcal{B}$  chooses random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  and  $h_p, f_p \leftarrow \mathbb{G}_p$  such that  $g = g_p g_q = \mathfrak{g}_p A, h = h_p h_q, f = f_p f_q$  and sets  $h = g^d, f = g^{d_1}$  for some exponents  $d, d_1 \in \mathbb{Z}_N$ . Now the challenger  $\mathcal{B}$  generates the following components:

- **General component:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = A^\beta$ .
- **Row-specific components:** For all  $x \in [\hat{n}], j \in [m]$  sample  $\tilde{r}_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N, \hat{r} \leftarrow \mathbb{Z}_N$  and compute  $F_{q,x,j} = E_q^{\hat{r}\tilde{r}_{x,j}}, G_{q,x,j} = e(A, A)^{\beta\alpha_{x,j}}, W_{q,x,j} = e(A, A)^{\beta d_1 \psi_{x,j}}, F_{x,j} = E_q^{\hat{r}\tilde{r}_{x,j}}, G_{x,j} = e(\mathfrak{g}_p A, \mathfrak{g}_p A)^{\alpha_{x,j}}, W_{x,j} = e(\mathfrak{g}_p A, \mathfrak{g}_p A)^{d_1 \psi_{x,j}}$  with

$$E_{q,x,j} = \begin{cases} A^{\beta\tilde{r}\tilde{r}_{x,j}} & \text{if } x \neq x^*, \\ \mathfrak{g}_q^{\beta\tilde{r}\tilde{r}_{x,j}} & \text{otherwise.} \end{cases}, \quad E_{x,j} = \begin{cases} (\mathfrak{g}_p A)^{\tilde{r}\tilde{r}_{x,j}} & \text{if } x \neq x^*. \\ (\mathfrak{g}_p \mathfrak{g}_q)^{\tilde{r}\tilde{r}_{x,j}} & \text{otherwise.} \end{cases}$$

- **Column-specific components:** For all  $y \in [\hat{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $c_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$ , and compute  $H_{y,\ell,b} = (\mathfrak{g}_p A)^{c_{y,\ell,b}}, \tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}, V_{\ell,b} = h^{\delta_{\ell,b}}$ .

- **gid-specific components:** The challenger  $\mathcal{B}$  samples group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  chooses  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\mathbf{H}(\mathbf{gid}) = \vartheta'_p \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i$ ,  $\mathbf{H}_q(\mathbf{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  where  $\vartheta' = \vartheta'_p \vartheta'_q \in \mathbb{G}$ ,  $\boldsymbol{\vartheta} = (\vartheta_i)_{i \in [k']} \in \mathbb{G}^{k'}$ .

Now, the challenger  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \begin{array}{l} \mathbb{B}\mathbb{G}, g = \mathfrak{g}_p A, h = g^d, f = g^{d_1}, \vartheta', \vartheta_q^{\beta}, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^{\beta}\}_{i \in [k']}, \mathbf{H}, \mathbf{H}_q, E_q, \\ \left\{ \begin{array}{l} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\tilde{n}] \times [m]} \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}}, \{\tilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right)$$

**Secret key simulation.** First,  $\mathcal{B}$  computes  $\mathbf{H}(\mathbf{gid}) = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i = g^{d'} = (\mathfrak{g}_p A)^{d'}$  for some  $d' \in \mathbb{Z}_N$ . In the query phase, the adversary  $\mathcal{A}$  is not allowed to query for the challenge index and group identity  $i^* = (x^*, y^*), \mathbf{gid}^*$  together. If  $\mathbf{gid} = \mathbf{gid}^*$ ,  $\mathcal{B}$  answers the secret key  $\text{sk}_{\mathbf{u}}$  corresponding to the tuple  $(i, \text{id}, \mathbf{gid}, \mathbf{u})$  as given below. Note that, the adversary is not allowed to secret key query for the index position  $i^*$ .

$$K_1 = \begin{cases} g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot (\mathfrak{g}_p A)^{\widehat{r}(\widetilde{r}_x, \mathbf{u}) \sum_{\ell \in [k]} c_{y,\ell, \text{id}_{\ell}}} & \text{if } x \neq x^*, \\ g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot (\mathfrak{g}_p \mathfrak{g}_q)^{\widehat{r}(\widetilde{r}_x, \mathbf{u}) \sum_{\ell \in [k]} c_{y,\ell, \text{id}_{\ell}}} & \text{otherwise.} \end{cases}$$

$$K_2 = (\mathfrak{g}_p A)^{d_1 \langle \psi_x, \mathbf{u} \rangle} \mathbf{H}(\mathbf{gid})^r, \quad K_3 = g^r.$$

If  $\mathbf{gid} \neq \mathbf{gid}^*$ , then the secret keys corresponding to the index  $i^*$  looks as

$$K_1 = g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot (\mathfrak{g}_p \mathfrak{g}_q)^{\widehat{r}(\widetilde{r}_x, \mathbf{u}) \sum_{\ell \in [k]} c_{y,\ell, \text{id}_{\ell}}} \text{ if } x = x^*, y = y^* = \tilde{n}$$

To simulate the secret key components  $K_2$  and  $K_3$ , the challenger constructs the functions F, J and K as mentioned in indistinguishability proof of hybrid 2 and hybrid 3. Also,  $\mathcal{B}$  sets similar public key parameters  $\vartheta'$  and  $\vartheta_i$ 's. It answers the remaining secret key components as follows:

$$\begin{aligned} K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{\mathbf{J}(\mathbf{gid})}{\mathbf{F}(\mathbf{gid})}} \mathbf{H}(\mathbf{gid})^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{\mathbf{F}(\mathbf{gid})} g^{\mathbf{J}(\mathbf{gid})} \right)^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}} \left( f^{\mathbf{F}(\mathbf{gid})} g^{\mathbf{J}(\mathbf{gid})} \right)^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{\mathbf{F}(\mathbf{gid})} g^{\mathbf{J}(\mathbf{gid})} \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{\mathbf{F}(\mathbf{gid})} g^{\mathbf{J}(\mathbf{gid})} \right)^{r'} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \mathbf{H}(\mathbf{gid})^{r'} \\ K_3 &= g^r \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}} = g^{r'} \end{aligned}$$

As in the previous argument, we implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}$ . Therefore, from the construction of K function, we get  $\mathbf{K}(\mathbf{gid}) \neq 0$  for any key query corresponding to the group identity  $\mathbf{gid} \neq \mathbf{gid}^*$ . This implies the function  $\mathbf{F}(\mathbf{gid}) \neq 0 \pmod N$  for any such group identity (since we assume  $N > sk'_1$ , see Lemma 9).

**Challenge ciphertext simulation.** The challenger  $\mathcal{B}$  samples the exponents  $\tau \in \mathbb{Z}_N$  and for the challenge identity  $\mathbf{gid}^*$ ,  $\mathcal{B}$  computes  $\mathbf{H}(\mathbf{gid}^*) = (\vartheta') \prod_{i \in \mathcal{V}^*} \vartheta_i = g_p^{d_1^*} g_q^{d_2^*} = \mathfrak{g}_p^{d_1^*} A^{d_2^*}$  and  $\mathbf{H}_q(\mathbf{gid}^*)^\beta = \vartheta_q^{\beta} \prod_{i \in \mathcal{V}^*} \vartheta_{q,i}^\beta = g_q^{\beta d_2^*} = A^{\beta d_2^*}$  for some exponents  $d_1^*, d_2^* \in \mathbb{Z}_N$ . Now  $\mathcal{B}$  simulates the challenge ciphertext components as follows.

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x \leftarrow \mathbb{Z}_N, \tilde{s}_x \leftarrow \mathbb{Z}_N$ , for all  $x \in [\hat{n}]$  and categorize the components according  $x > x^*, x = x^*, x < x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = \mathfrak{g}_q^{\beta \tilde{r} r_{x,j} \tilde{s}_x}, \tilde{R}_{x,j} = \mathfrak{g}_q^{\beta \tilde{r} \tau d_{x,j} \tilde{s}_x}$ .
2. *gid-specific component:*  $B_x = A^{d_2^* \beta \tilde{s}_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(A, A)^{\nu_j} e(A, A)^{\beta \alpha_{x,j} \tilde{s}_x} e(A, A)^{d_1 \beta \tilde{s}_x \psi_{x,j}}, A_x = A^{\beta \tilde{s}_x}$ .

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = C^{\tilde{r} r_{x,j}}, \tilde{R}_{x,j} = C^{\tilde{r}_{x,j} \tilde{r} \tau d}$ .
2. *gid-specific component:*  $B_x = D^{d_1^*} T^{d_2^*}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{\phi_j f_x} e(g, g)^{d_1^* \phi_j f_x}, A_x = DT$ .

**For  $x < x^*$ :**

1. *Linking components:*  $R_{x,j} = g^{\tilde{s}_x \sigma_j}, \tilde{R}_{x,j} = g^{d_{s_x} \tau \nu_j}$ .
2. *gid-specific component:*  $B_x = g^{d_1 d_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} e(g, g)^{d_1^* \phi_j f_x}, A_x = g^{e_x}$ .

- **Column-specific components:** Sample  $\tilde{w}_{y,\ell,b}, \tilde{v}_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$  and generate the components as follows:

**For all  $y \in [\tilde{n}]$ :**  $C_{y,\ell,b} = B^{c_{y,\ell,b}} h^{\tilde{w}_{y,\ell,b} \tau} \mathfrak{g}_p^{\tilde{v}_{y,\ell,b}}, \tilde{C}_{y,\ell,b} = g^{\tilde{w}_{y,\ell,b}}$ .

**Analysis of simulation.** For  $T = \mathfrak{g}_q^{a^2 c}$ ,  $\mathcal{B}$  simulates the hybrid 4, otherwise if  $T$  is randomly chosen from the group  $\mathbb{G}_q$  then  $\mathcal{B}$  simulates the view of hybrid 5. As the target row ' $x^*$ ' is indistinguishable from the less than row ' $< x^*$ '. Therefore, if  $\mathcal{A}$  wins the game with the advantage  $\epsilon(\cdot)$  then  $\mathcal{B}$  breaks the R3DH assumption with the same advantage.

**Hybrid 5  $\approx$  Hybrid 6:** To prove the hybrid 5 and 6 are indistinguishable, let us consider  $(2\tilde{n}k+1)$  sub-hybrid  $H_0, H_{y,\ell,b}^{\sim\sim}$  for  $(\tilde{y}, \tilde{\ell}, \tilde{b}) \in ([\tilde{n}] \times [k] \times \{0, 1\})$ . In this game, the sub-hybrid  $H_0$  corresponds to the hybrid 5 as described above. Now the sub-hybrid  $H_{y,\ell,b}^{\sim\sim}$  is same as the hybrid  $H_0$  except that the column components in the challenge ciphertext  $C_{y,\ell,b}$  for  $y < \tilde{y}$  and  $(y, \ell, b) \in \{\tilde{y}\} \times [\tilde{\ell} - 1] \times \{0, 1\}$  and for  $y = \tilde{y}, \ell = \tilde{\ell}, b < \tilde{b}$  have a random element in the subgroup  $\mathbb{G}_p$ . The column components are generated as described below in the Table 5.

Table 5: Computing column components of the ciphertext in sub-hybrid  $H_{y,\ell,b}^{\sim\sim}$

	$C_{y,\ell,b}$	$\tilde{C}_{y,\ell,b}$
$(y > \tilde{y}) \vee (y = \tilde{y} \wedge \ell > \tilde{\ell}) \vee (y = \tilde{y} \wedge \ell = \tilde{\ell} \wedge b \geq \tilde{b})$	$H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b} \tau}$	$g^{w_{y,\ell,b}}$
$(y < \tilde{y}) \vee (y = \tilde{y} \wedge \ell < \tilde{\ell}) \vee (y = \tilde{y} \wedge \ell = \tilde{\ell} \wedge b < \tilde{b})$	$H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b} \tau} \cdot V_{\ell,b}^{v_{y,\ell,b} \tau}$	$g^{w_{y,\ell,b}} \cdot \tilde{V}_{\ell,b}^{v_{y,\ell,b}}$

Now we show that sub-hybrid  $H_{y,\ell,b}^{\sim\sim}$  is indistinguishable with sub-hybrid  $H_{y,\ell+b-1,(\tilde{b}+1) \bmod 2}^{\sim\sim}$ . Note that, the sub-hybrid  $H_{1,1,0}$  is identical to the main hybrid 6. It is also required

that the sub-hybrid  $H_0 \approx H_{\tilde{n}, \ell, 1}$  and sub-hybrid  $H_{y, k, 1} \approx H_{y+1, 1, 0}$ . We now show that the sub-hybrid  $H_{y, \ell, b}$  and  $H_{y, \ell+b-1, (\tilde{b}+1) \bmod 2}$  are indistinguishable with the similar techniques that are used to prove the indistinguishability of all the consecutive sub-hybrids for  $(y, \ell, b) \in [\tilde{n}] \times [k] \times \{0, 1\}$ . By combining all claims of sub-hybrids, our required indistinguishability between hybrids 5 and 6 will follow.

*Sub-hybrid  $H_{y, \ell, b} \approx$  Sub-hybrid  $H_{y, y+b-1, (\tilde{b}+1) \bmod 2}$ :* Suppose on the contrary, there exist a PPT adversary  $\mathcal{A}$  that distinguishes sub-hybrid  $H_{y, \ell, b}$  and sub-hybrid  $H_{y, y+b-1, (\tilde{b}+1) \bmod 2}$  with the non-negligible advantage  $\epsilon(\cdot)$ . We construct a PPT reduction algorithm  $\mathcal{B}$  which can break the modified-2 D3DH assumption 3 with the same non-negligible advantage as described above. From the challenger, the reduction algorithm  $\mathcal{B}$  receives the following instance as

$$(\mathbb{B}\mathbb{G}, g_p, g_q, A = g_p^a, B = g_p^b, C = g_p^c, T)$$

where  $T$  is either  $g_p^{abc}$  or a random element in the subgroup  $\mathbb{G}_p$  of prime-order  $p$ . Next, it receives the challenge tuple  $(1^\lambda, 1^n, 1^k, 1^{k'}, 1^m, x^*, y^*, \mathbf{gid}^*)$  from the adversary  $\mathcal{A}$  where  $y^* = \tilde{n}$ . Now,  $\mathcal{B}$  generates the master public key using the modified-2 D3DH instance 3 and sends it to the adversary. Then the adversary can make query for the secret keys corresponding to the distinct indices except that the tuple  $(i^*, \mathbf{id}, \mathbf{gid}^*, \mathbf{u})$  to the key generation oracle. In the following,  $\mathcal{B}$  generates the master public key, secret keys and the challenge ciphertext. Finally,  $\mathcal{A}$  outputs its guess, which  $\mathcal{B}$  uses to break the given modified-2 D3DH assumption 3. Since the reduction plays with the challenger over the subgroup  $\mathbb{G}_p$ , thus it can choose any elements from the subgroup  $\mathbb{G}_q$  by itself. Now implicitly sets  $t_p = a \cdot b$ ,  $h_p = B = g_p^b$  and  $c_{p, y, \ell, b} = c \cdot \tilde{c}_{p, y, \ell, b}$  where the exponent  $\tilde{c}_{p, y, \ell, b}$  is chosen uniformly random from  $\mathbb{Z}_N$ . By using these exponents,  $\mathcal{B}$  simulates the master public key, secret keys and the group elements  $T$  can be programmed in the challenge ciphertext components  $C_{y, \ell, b}$ .

**Public key simulation.** The challenger  $\mathcal{B}$  chooses random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  and  $f_p \leftarrow \mathbb{G}_p$  such that  $h = h_p h_q, f = f_p f_q$  and sets  $h_q = g_q^d, h = B h_q, f = g_p^{d_1} g_q^{d_2}$  for some random exponents  $d, d_1, d_2 \in \mathbb{Z}_N$ . Now, the challenger  $\mathcal{B}$  generates the following components:

- **General components:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = g_q^\beta$ .
- **Row-specific components:** For all  $x \in [\tilde{n}], j \in [m]$  sample  $r_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N$ ,  $\hat{r} \leftarrow \mathbb{Z}_N$  and compute  $E_{q,x,j} = g_q^{\hat{r} r_{x,j}}, F_{q,x,j} = h_q^{\hat{r} r_{x,j}}, G_{q,x,j} = e(g_q, g_q)^{\beta \alpha_{x,j}}, W_{q,x,j} = e(g_q, g_q)^{\beta d_2 \psi_{x,j}}, E_{x,j} = g^{r r_{x,j}}, F_{x,j} = h^{r r_{x,j}}, G_{x,j} = e(g, g)^{\alpha_{x,j}}, W_{x,j} = e(g_p, g_p)^{d_1 \psi_{x,j}} \cdot e(g_q, g_q)^{d_2 \psi_{x,j}}$ .
- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $\tilde{c}_{y, \ell, b}, \delta_{\ell, b} \leftarrow \mathbb{Z}_N, \gamma_{\ell, b} \leftarrow \mathbb{Z}_p$  and compute  $\tilde{V}_{\ell, b} = g^{\delta_{\ell, b}} g_p^{\gamma_{\ell, b}}, V_{\ell, b} = h^{\delta_{\ell, b}}$ ,

$$H_{y, \ell, b} = \begin{cases} (C g_q)^{\tilde{c}_{y, \ell, b}} & \text{if } (y, \ell, b) = (\tilde{y}, \tilde{\ell}, \tilde{b}) \\ (g_p g_q)^{\tilde{c}_{y, \ell, b}} & \text{otherwise.} \end{cases}$$

- **gid-specific components:** The challenger  $\mathcal{B}$  samples group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  chooses  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\mathbf{H}(\mathbf{gid}) = \vartheta'_p \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i, \mathbf{H}_q(\mathbf{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  where  $\vartheta' = \vartheta'_p \vartheta'_q \in \mathbb{G}, \vartheta = (\vartheta_i)_{i \in [k']} = (\vartheta_{p,i} \vartheta_{q,i})_{i \in [k']} \in \mathbb{G}^{k'}$ .

Using the instance of modified-2 D3DH assumption 3, the challenger  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \begin{array}{l} \mathbb{B}\mathbb{G}, g = g_p g_q, h = Bh_q, f = g_p^{d_1} g_q^{d_2}, \vartheta', \vartheta'_q{}^\beta, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^\beta\}_{i \in [k']}, H, H_q, E_q, \\ \left\{ \begin{array}{l} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\widehat{n}] \times [m]}, \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\widehat{n}] \times [k] \times \{0,1\}}, \{\widetilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right)$$

All the components are generated from the challenge D3DH instance.

**Secret key simulation.** In the query phase, the adversary  $\mathcal{A}$  is not allowed to query for the challenge index  $i^*$  and the group identity  $\text{gid}^*$  together. If  $\text{gid} = \text{gid}^*$ ,  $\mathcal{B}$  replies the secret key to  $\mathcal{A}$  corresponding to the index  $i (\neq i^*)$  as

$$K_1 = \begin{cases} g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot g^{\widehat{r} \langle r_x, \mathbf{u} \rangle \sum_{\ell \neq \ell} \widetilde{c}_{y,\ell, \text{id}_\ell} (Cg_q)^{\widehat{r} \langle r_x, \mathbf{u} \rangle \widetilde{c}_{y,\ell,b}^{\sim}}} & \text{if } (y, \text{id}_{\widetilde{y}}) = (\widetilde{y}, \widetilde{b}), \\ g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot g^{\widehat{r} \langle r_x, \mathbf{u} \rangle \sum_{\ell \in [k]} \widetilde{c}_{y,\ell, \text{id}_\ell}} & \text{otherwise.} \end{cases},$$

$$K_2 = g^{d \langle \psi_x, \mathbf{u} \rangle} H(\text{gid})^r, \quad K_3 = g^r$$

where  $\widetilde{r} \leftarrow \mathbb{Z}_N$  and set  $r = \widetilde{r} \cdot \widehat{r}$ . If  $\text{gid} \neq \text{gid}^*$ , then  $\mathcal{A}$  can query for the secret for the index position  $i^*$ . Then the secret keys are generated as

$$K_1 = \begin{cases} g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot g^{\widehat{r} \langle r_x, \mathbf{u} \rangle \sum_{\ell \neq \ell} \widetilde{c}_{y,\ell, \text{id}_\ell} (Cg_q)^{\widehat{r} \langle r_x, \mathbf{u} \rangle \widetilde{c}_{y,\ell,b}^{\sim}}} & \text{if } y = \widetilde{y} = \widetilde{n} \wedge \text{id}_{\widetilde{y}} = \widetilde{b}, \\ g^{\langle \alpha_x, \mathbf{u} \rangle} \cdot g^{\widehat{r} \langle r_x, \mathbf{u} \rangle \sum_{\ell \in [k]} \widetilde{c}_{y,\ell, \text{id}_\ell}} & \text{if } y = \widetilde{y} = \widetilde{n} \wedge \text{id}_{\widetilde{y}} \neq \widetilde{b}, \end{cases}$$

To simulate the other keys components, the challenger similarly construct the functions F, J and K as mentioned above indistinguishability proof of hybrid 2 and hybrid 3. Also,  $\mathcal{B}$  sets similar public key parameters  $\vartheta'$  and  $\vartheta_i$ 's. It answers the remaining secret key components as follows:

$$\begin{aligned} K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} H(\text{gid})^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^F(\text{gid}) g^J(\text{gid}) \right)^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \left( f^F(\text{gid}) g^J(\text{gid}) \right)^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^F(\text{gid}) g^J(\text{gid}) \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^F(\text{gid}) g^J(\text{gid}) \right)^{r'} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} H(\text{gid})^{r'} \\ K_3 &= g^r \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'} \end{aligned}$$

As in the previous argument, we implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}$ . Therefore, from the construction of K function, it implies that  $K(\text{gid}) \neq 0$  for any key query corresponding to the group identity  $\text{gid} \neq \text{gid}^*$ . This implies the function  $F(\text{gid}) \neq 0 \pmod N$  for any such group identities (since we assume  $N > sk'_1$  for reasonable values of  $N, s$  and  $k'_1$ , see Lemma 9).

**Challenge ciphertext simulation.** The challenger  $\mathcal{B}$  samples the random exponents  $\tau \in \mathbb{Z}_N, t_q \leftarrow \mathbb{Z}_N$  and for the challenge group identity  $\text{gid}^*$ ,  $\mathcal{B}$  computes  $H(\text{gid}^*) = (\vartheta'_p \vartheta'_q) \prod_{i \in \mathcal{V}^*} \vartheta_{p,i} \vartheta_{q,i} = g_p^{d_1^*} g_q^{d_2^*}, H_q(\text{gid}^*)^\beta = \vartheta'_q{}^\beta \prod_{i \in \mathcal{V}^*} \vartheta_{q,i}^\beta = g_q^{\beta d_2^*}$  for some  $d_1^*, d_2^* \in \mathbb{Z}_N$ . Now,  $\mathcal{B}$  simulates the challenge ciphertext components as follows.



- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x, s_x \leftarrow \mathbb{Z}_N$  for all  $x \in [\hat{n}]$  and categorize the components according  $x > x^*, x \leq x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{s_x}, \tilde{R}_{x,j} = F_{q,x,j}^{s_x \tau}$ .
2. *gid-specific component:*  $B_x = H_q(\text{gid}^*)^{\beta s_x t_q}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{q,x,j}^{s_x t_q} \cdot W_{q,x,j}^{s_x t_q}, A_x = E_q^{s_x t_q}$ .

**For  $x \leq x^*$ :**

1. *Linking components:*  $R_{x,j} = g^{s_x \sigma_j}, \tilde{R}_{x,j} = h^{s_x \tau \nu_j}$ .
2. *gid-specific component:*  $B_x = H(\text{gid}^*)^{d_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}, A_x = g^{e_x}$ .

- **Column-specific components:** Sample  $\tilde{w}_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\hat{n}], \ell \in [k], b \in \{0, 1\}$  and generates the components as follows:

**For  $(y > \tilde{y}) \vee (y = \tilde{y} \wedge \ell > \tilde{\ell}) \vee (y = \tilde{y} \wedge \ell = \tilde{\ell} \wedge b > \tilde{b})$ :**  $C_{y,\ell,b} = g_q^{c_{y,\ell,b} t_q} h^{\tilde{w}_{y,\ell,b} \tau}, \tilde{C}_{y,\ell,b} = A^{-c_{y,\ell,b} / \tau} \cdot g^{\tilde{w}_{y,\ell,b}}$ .

**For  $y = \tilde{y} \wedge \ell = \tilde{\ell} \wedge b = \tilde{b}$ :**  $C_{y,\ell,b} = g_q^{c_{y,\ell,b} t_q} h^{\tau \tilde{w}_{y,\ell,b}} T^{c_{y,\ell,b}}, \tilde{C}_{y,\ell,b} = g^{\tilde{w}_{y,\ell,b}}$ .

**For  $(y < \tilde{y}) \vee (y = \tilde{y} \wedge \ell < \tilde{\ell}) \vee (y = \tilde{y} \wedge \ell = \tilde{\ell} \wedge b < \tilde{b})$ :**  $C_{y,\ell,b} = g_q^{c_{y,\ell,b} t_q} h^{\tau \tilde{w}_{y,\ell,b}} g_p^{v_{y,\ell,b}}, \tilde{C}_{y,\ell,b} = g^{\tilde{w}_{y,\ell,b}}$ .

**Analysis of simulation.** For  $T = g_p^{abc}$ , then  $\mathcal{A}$  gets the view of the challenge ciphertext as the sub-hybrid  $H_{y,\tilde{y}+b-1,(\tilde{b}+1) \bmod 2}$ , otherwise for any random group element from the subgroup  $\mathbb{G}_p$ , the adversary  $\mathcal{A}$  gets the view of the sub-hybrid  $H_{y,\ell,b}$ . Therefore, if  $\mathcal{A}$  wins the with the advantage  $\epsilon(\cdot)$  then  $\mathcal{B}$  breaks the modified-2 D3DH assumption 3 with the same advantage  $\epsilon(\cdot)$ .

**Hybrid 6  $\approx$  Hybrid 7:** Suppose on the contrary, there exists a PPT adversary  $\mathcal{A}$  that can distinguish between the hybrid 6 and hybrid 7 with non-negligible advantage  $\epsilon(\cdot)$ . Now, we construct a PPT reduction algorithm  $\mathcal{B}$  that breaks the DHSD assumption 5 with the same advantage.

The reduction algorithm  $\mathcal{B}$  first receives the DHSD challenge instance of assumption 5 from its challenger as

$$(\mathbb{B}\mathbb{G}, g = g_p g_q, h = h_p h_q, A = g_q^a, B = h_q^a, C = g^b g_p^c, D = h^b, T)$$

where  $T$  is either sampled as  $T = g_q^d$  or  $T = g^d$  where  $d$  is a random exponent sampled as  $d \leftarrow \mathbb{Z}_N$ . Next,  $\mathcal{B}$  receives the challenge tuple  $(1^\lambda, 1^n, 1^k, 1^{k'}, 1^m, x^*, y^*, \text{gid}^*)$  from the adversary for  $y^* = \tilde{n}$ . Now,  $\mathcal{B}$  generates the master public key from the instance of assumption 5 and sends it to the adversary. The adversary is not allowed to make secret key queries for the tuple  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$ . Now  $\mathcal{B}$  simulates the secret keys and the challenge ciphertext using the instance of DHSD assumption 5. Finally  $\mathcal{A}$  outputs its guess which  $\mathcal{B}$  uses to break the assumption 5. In this proof, the reduction plays with its challenger in the subgroup  $\mathbb{G}_p$  thus it chooses any components from the subgroup  $\mathbb{G}_q$  by itself. Let us consider,  $\mathcal{B}$  first implicitly sets the random exponents as  $\beta = a, s_{x^*+1} = d \cdot \tilde{s}_{x^*+1}, \gamma_{\ell,b} = c \cdot \tilde{\gamma}_{\ell,b}$  and  $\delta_{\ell,b} = b \cdot \tilde{\gamma}_{\ell,b} + \tilde{\delta}_{\ell,b}$  where the exponents  $\tilde{\gamma}_{\ell,b}, \tilde{\delta}_{\ell,b}$  are chosen uniformly random from  $\mathbb{Z}_N$ . Also  $\mathcal{B}$  implicitly sets  $h^\tau = g^{\tilde{\tau}}$  where  $\tilde{\tau}$  be any random exponent from  $\mathbb{Z}_N$ . In this simulation, the challenge group element  $T$  can be programmed in the challenge ciphertext to compute the row components for  $x = x^* + 1$ .

**Public key simulation.** The challenger  $\mathcal{B}$  chooses random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  and  $h_p, f_p \leftarrow \mathbb{G}_p$  such that  $h = h_p h_q, f = f_p f_q$ . Sets  $f_q = g_q^{d'}$  for some exponent  $d' \in \mathbb{Z}_N$ . Now, the challenger  $\mathcal{B}$  generates the following components:

- **General components:** Compute  $E_q = A$ .
- **Row-specific components:** For all  $x \in [\hat{n}], j \in [m]$  sample  $r_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N$ ,  $\hat{r} \leftarrow \mathbb{Z}_N$  and computes  $E_{q,x,j} = A^{\hat{r} r_{x,j}}, F_{q,x,j} = B^{\hat{r} r_{x,j}}, G_{q,x,j} = e(A, g_q)^{\alpha_{x,j}}, W_{q,x,j} = e(f_q, A)^{\psi_{x,j}}, E_{x,j} = g^{\hat{r} r_{x,j}}, F_{x,j} = h^{\hat{r} r_{x,j}}, G_{x,j} = e(g, g)^{\alpha_{x,j}}, W_{x,j} = e(f, g)^{\psi_{x,j}}$ .
- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $c_{y,\ell,b}, \tilde{\delta}_{\ell,b} \leftarrow \mathbb{Z}_N, \tilde{\gamma}_{\ell,b} \leftarrow \mathbb{Z}_p$  and computes  $H_{y,\ell,b} = g^{c_{y,\ell,b}}, \tilde{V}_{\ell,b} = g^{\tilde{\delta}_{\ell,b}} C^{\tilde{\gamma}_{\ell,b}}, V_{\ell,b} = h^{\tilde{\delta}_{\ell,b}} D^{\tilde{\gamma}_{\ell,b}}$ .
- **gid-specific components:** The challenger  $\mathcal{B}$  samples random group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  samples  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\vartheta^p = \vartheta'_p \vartheta'_q, \boldsymbol{\vartheta} = (\vartheta_{p,i} \vartheta_{q,i})_i$  and sets  $\mathsf{H}(\mathsf{gid}) = (\vartheta'_p \vartheta'_q) \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta' \prod_{i \in \mathcal{V}} \vartheta_i$  and  $\mathsf{H}_q(\mathsf{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  corresponding to any group identity  $\mathsf{gid}$ . We write  $\vartheta'_q = g_q^{\alpha} = A^{\alpha}, \vartheta_{q,i} = g_q^{\alpha_i} = A^{\alpha_i}$  where  $\alpha, \alpha_i \leftarrow \mathbb{Z}_q$ .

Using the challenge instance of DHSD assumption 5, the challenger  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g = g_p g_q, h = h_p h_q, f = f_p f_q, \vartheta', \vartheta_q^{\beta}, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^{\beta}\}_{i \in [k']}, \mathsf{H}, \mathsf{H}_q, E_q, \\ \left\{ \begin{array}{c} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\hat{n}] \times [m]}, \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}}, \{\tilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right)$$

**Secret key simulation.** The challenger  $\mathcal{B}$  answers the adversary's queried secret keys corresponding to the tuple  $(i, \text{id}, \mathsf{gid}, \mathbf{u})$ .

For  $\mathsf{gid} = \mathsf{gid}^*$ , the adversary cannot make any secret key queries for the index position  $i^*$ . Therefore, the secret keys corresponding to the index  $i (\neq i^*)$  are generated as follows:

$$K_1 = g^{(\alpha_x, \mathbf{u})} \cdot \left( \prod_{\ell \in [k]} H_{y,\ell, \text{id}_{\ell}} \right)^{\hat{r} \langle r_x, \mathbf{u} \rangle}, \quad K_2 = f^{\langle \psi_x, \mathbf{u} \rangle} \mathsf{H}(\mathsf{gid})^r, \quad K_3 = g^r$$

where  $r = \tilde{r} \cdot \hat{r}$  and  $\tilde{r}$  is randomly sampled from  $\mathbb{Z}_N$ . If  $\mathsf{gid} \neq \mathsf{gid}^*$ , the adversary can make secret key query for the index position  $i^*$ . In that case,  $\mathcal{B}$  answers the secret key  $\text{sk}_{\mathbf{u}}$  as follows

$$K_1 = g^{(\alpha_x, \mathbf{u})} \cdot \left( \prod_{\ell \in [k]} H_{y,\ell, \text{id}_{\ell}} \right)^{\hat{r} \langle r_x, \mathbf{u} \rangle} \quad \text{if } x = x^*, y = y^*$$

The other secret keys are similarly generated as the previous hybrid. For  $\mathsf{gid} \neq \mathsf{gid}^*$ , challenger construct the identity encoding functions F, J and K similarly as the indistinguishability proof of hybrid 2 and hybrid 3. Now,  $\mathcal{B}$  answers the remaining secret keys

components  $K_2, K_3$  that looks as follows:

$$\begin{aligned}
K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} \mathbf{H}(\text{gid})^r \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^r \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r'} \\
&= f^{\langle \psi_x, \mathbf{u} \rangle} \mathbf{H}(\text{gid})^{r'} \\
K_3 &= g^r \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'}
\end{aligned}$$

As the similar argument, we can implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}$ .

**Challenge ciphertext simulation.** The challenger  $\mathcal{B}$  samples the random exponents  $\tilde{\tau} \in \mathbb{Z}_N, t \leftarrow \mathbb{Z}_N$  and for the challenge identity  $\text{gid}^*$ , we have  $\mathbf{H}(\text{gid}^*) = (\vartheta'_p \vartheta'_q) \prod_{i \in \mathcal{V}^*} \vartheta_{p,i} \vartheta_{q,i} = g_p^{d_1^*} g_q^{d_2^*} = g^{d^*}$  and  $\mathbf{H}_q(\text{gid}^*) = \vartheta'_q \prod_{i \in \mathcal{V}^*} \vartheta_{q,i} = g_q^{d_2^*}$  for some random exponents  $d^*, d_1^*, d_2^* \in \mathbb{Z}_N$ . So,  $\mathbf{H}_q(\text{gid}^*)^\beta = A^{d_2^*}$ . Now,  $\mathcal{B}$  simulates the challenge ciphertext components as follows.

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x, \tilde{s}_x \leftarrow \mathbb{Z}_N$  for all  $x \in [\hat{n}]$  and categorize the components according  $x > x^* + 1, x = x^* + 1, x < x^* + 1$  as follows:

**For  $x > x^* + 1$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{\tilde{s}_x}, \tilde{R}_{x,j} = F_{q,x,j}^{\tilde{s}_x \tilde{\tau}}$ .
2. *gid-specific component:*  $B_x = A^{d_2^* \tilde{s}_x t}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{q,x,j}^{\tilde{s}_x t} \cdot e(f_q, A)^{\tilde{s}_x t \psi_{x,j}}, A_x = E_q^{\tilde{s}_x t}$ .

**For  $x = x^* + 1$ :**

1. *Linking components:*  $R_{x,j} = T^{\tilde{s}_x r_{x,j}}, \tilde{R}_{x,j} = T^{\tilde{s}_x r_{x,j} \tilde{\tau}}$ .
2. *gid-specific component:*  $B_x = T^{d^* \tilde{s}_x t}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot e(T, g)^{\tilde{s}_x \alpha_{x,j} t} \cdot e(f, T)^{\psi_{x,j} \tilde{s}_x t}, A_x = T^{\tilde{s}_x t}$ .

**For  $x < x^* + 1$ :**

1. *Linking components:*  $R_{x,j} = g^{\tilde{s}_x \sigma_j}, \tilde{R}_{x,j} = h^{\tilde{s}_x \tau \nu_j}$ .
2. *gid-specific component:*  $B_x = \mathbf{H}(\text{gid}^*)^{d_x}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}, A_x = g^{e_x}$ .

- **Column-specific components:** Sample  $w_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$  and generates the components as follows:

**For all  $(y, \ell, b) \in [\tilde{n}] \times [k] \times \{0, 1\}$ :**  $C_{y,\ell,b} = H_{y,\ell,b}^t \cdot g^{w_{y,\ell,b} \tilde{\tau}}, \tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}}$ .

After seeing the challenge ciphertext,  $\mathcal{B}$  receives a guess bit  $\mathbf{b}'$  from  $\mathcal{A}$  and it simply forwards that as its guess bit to the challenger of DHSD assumption 5.

**Analysis of simulation.** Finally, if  $T = g_q^d$ , then  $\mathcal{B}$  simulates the view of the hybrid 6, otherwise, if  $T$  is randomly chosen element from the group  $\mathbb{G}$  then  $\mathcal{B}$  simulates the view same as hybrid 7. Therefore, if  $\mathcal{A}$  wins with the advantages  $\epsilon(\cdot)$  then  $\mathcal{B}$  breaks the DHSD assumption 5 with the same advantage. Hence proof of the Claim 7.2 is complete.  $\square$

This concludes the proof of index-hiding security.  $\square$

**Lemma 4.** *If the modified-1 D3DH assumption 2 holds over the bilinear group  $\mathbb{BG}$ , then our EIPL-IBIPFE satisfies selective lower identity-hiding security as per the Definition 6.*

*Proof.* We recall that in the lower identity-hiding security, it is required that no PPT adversary can distinguish between the special encryption to the index-position-bit tuple  $(i^* = (x^*, y^*), \ell^*, b^*)$  and  $(i^* = (x^*, y^*), \perp, 0)$  with non-negligible advantages. In the secret key query phase, the adversary is not allowed to secret key query for the tuple  $(i^* = (x^*, y^*), \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = b^*$ . This proof technique is nearly identical to that of Claim 7.2. Here, we just exclude the intermediate hybrids as mentioned in the previous proof. Let  $(i^* = (x^*, y^*), \ell^*, b^*)$  be the challenge tuple provided by the adversary  $\mathcal{A}$ . Then the hybrid  $H_{\ell^*, b^*}$  corresponds to the exactly same as the lower identity-hiding game in which the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^*), \ell^*, b^*)$  and similarly the hybrid  $H_{0,1}$  is same as the lower identity-hiding game for the index-position-bit tuple  $(i^* = (x^*, y^*), \perp, 0)$ . So the indistinguishability proof to the tuple  $(i^* = (x^*, y^*), \perp, 0)$  and  $(i^* = (x^*, y^*), \ell^*, b^*)$  are similar to the Claim 7.2.

In the following, we discuss the secret key simulation where the reduction algorithm  $\mathcal{B}$  answers all permissible secret keys corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  as per the lower identity-hiding security game. From the security restriction of this game, the adversary cannot query for the secret key corresponding to the tuple  $(i^* = (x^*, y^*), \text{id}_{\ell^*} = b^*, \text{gid}^*, \mathbf{u})$ . So all the key queries are of the form either  $i \neq i^*$  or  $\text{id}_{\ell^*} \neq b^*$  or  $\text{gid} \neq \text{gid}^*$ .

**Secret key simulation.** To answer the secret key corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$ ,  $\mathcal{B}$  samples a random value  $\tilde{r} \leftarrow \mathbb{Z}_N$  and sets  $r = \tilde{r} \cdot \hat{r}$  and computes  $\text{H}(\text{gid}^*) = g_p^{d_1^*} g_q^{d_2^*}$  for some exponents  $d_1^*, d_2^* \in \mathbb{Z}_N$ . Note that, the adversary is not allowed to query for the secret key corresponding to the tuple  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = b^*$ .

If  $\text{gid} = \text{gid}^*$ ,  $\mathcal{A}$  cannot query for the secret key corresponding to the index  $i^*$  such that  $\text{id}^* = b^*$ . In Table 6, we show how  $\mathcal{B}$  generates all possible keys corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$ . For  $\text{gid} \neq \text{gid}^*$ ,  $\mathcal{B}$  generates the  $K_2, K_3$  secret key components as follows:

We assume that the adversary makes the maximum number of  $Q$  secret key queries corresponding to the challenge group identity  $\text{gid}^*$  and challenge index  $i^*$ . Now, the simulator chooses an integer  $k'_1 \leftarrow [k']$  and sets an integer  $s = 10Q$ , a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$ , a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, the simulator also chooses a random value  $w' \leftarrow \mathbb{Z}_N$  and a uniformly random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_N^{k'}$ . All these values are kept secret to the simulator. Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_\kappa\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  which correspond to the collection of indices  $\mathcal{V}^*$ . Then set  $\sum_{i \in \mathcal{V}^*} z_i = k'_1 s - z'$  for uniformly chosen  $k'_1 \in [k']$ . Now, we define the function  $\text{K}(\text{gid})$  as

$$\text{K}(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}^*} z_i \equiv 0 \pmod{s} \\ 1, & \text{otherwise.} \end{cases}$$

From the above definition of the function  $\text{K}$ , we can say that  $\text{K}(\text{gid}^*) = 0$  and for all



$\text{gid} \neq \text{gid}^*$  it becomes non-zero. Additionally, we set two functions as  $F(\text{gid}) = N - sk'_1 + z' + \sum_{i \in \mathcal{V}} z_i$  and  $J(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $\vartheta' = f^{N-k'_1s+z'} \cdot g^{w'} = g_p^{d'_p} g_q^{d'_q}$  and  $\vartheta_i = f^{z_i} g^{w_i} = g_p^{d_{p,i}} g_q^{d_{q,i}}$ . Now  $\mathcal{B}$  answers remaining secret key components as

$$\begin{aligned} K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} \cdot D^{d_1^* \tilde{r}} g_q^{d_2^* \tilde{r} \tilde{r}} \\ &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} H(\text{gid})^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r'} = f^{\langle \psi_x, \mathbf{u} \rangle} H(\text{gid})^{r'} \end{aligned}$$

$$K_3 = (D^{\tilde{r}} g_q^{\tilde{r} \tilde{r}}) \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'}$$

We implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}$ . So from the construction of K function, we get  $K(\text{gid}) \neq 0$  for any key query corresponding to the group identity  $\text{gid} \neq \text{gid}^*$ . This implies that the function  $F(\text{gid}) \neq 0 \pmod N$  for any such group identity (since we assume  $N > sk'_1$  for any reasonable value of  $N, s$  and  $k'_1$ , see Lemma 9).

Thus the above shows that the reduction algorithm can perfectly simulate the lower identity-hiding game. This implies that the scheme satisfy lower identity-hiding security, assuming that the modified-1 D3DH assumption 2 holds.  $\square$

**Lemma 5.** *If the assumptions 2,5,6 and 7 hold over the bilinear group  $\mathbb{B}\mathbb{G}$ , then our EIPL-IBIPFE satisfies selective upper identity-hiding security as per the Definition 7.*

*Proof.* The upper identity-hiding security requires that no PPT adversary can distinguish between the special encryption to the index-position-bit tuple  $(i^*, \ell^*, b^*)$  and  $(i^* + 1, \perp, 0)$  with a non-negligible advantage. In the security experiment, the adversary makes only one secret key query for some index position and it is not allowed to make only key query for the tuple  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} = 1 - b^*$ . To prove the Lemma 5, we consider a sequence of hybrid games as discuss below.

**Hyb<sub>1</sub>** : This hybrid corresponding to the upper identity-hiding game in which the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^*), \ell^*, b^*)$ .

**Hyb<sub>2</sub>** : This hybrid is similar to the Hyb<sub>1</sub> except that the column components as in the table below.

Table 7: Computing column components for  $(y, \ell, b) \in [\tilde{n}] \times [k] \times \{0, 1\}$

	$C_{y,\ell,b}$	$\tilde{C}_{y,\ell,b}$
$(y > y^*) \vee (y = y^* \wedge \ell \neq \ell^*)$	$H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b}\tau}$	$g^{w_{y,\ell,b}}$
$(y < y^*) \vee ((y, \ell) = (y^*, \ell^*))$	$H_{y,\ell,b}^t \cdot h^{w_{y,\ell,b}\tau} \cdot V_{\ell,b}^{\tau v_{y,\ell,b}}$	$g^{w_{y,\ell,b}} \cdot \tilde{V}_{\ell,b}^{v_{y,\ell,b}}$

In words, we can say that the ciphertext component  $C_{y^*, \ell^*, 1-b^*}$  also includes random elements from the subgroup  $\mathbb{G}_p$  whereas in Hyb<sub>1</sub> only  $C_{y^*, \ell^*, b^*}$  for the index position  $i^*$  include a random element from the subgroup  $\mathbb{G}_p$ .

**Hyb<sub>3, \tilde{\ell}, \tilde{b}</sub>** where  $(\tilde{\ell}, \tilde{b}) \in [k] \times \{0, 1\}$ : This hybrid is the same as Hyb<sub>2</sub> except that the column components in the challenge ciphertext are computed as in Table 8. In words, we can say that the challenge ciphertext components  $C_{y^*, \ell, b}$  for  $\ell < \tilde{\ell}$ , or  $\ell = \tilde{\ell}$  and  $b \leq \tilde{b}$  include a random element from  $\mathbb{G}_p$ .

**Hyb<sub>4</sub>** : This hybrid is similar to the previous sub-hybrid Hyb<sub>3, k, 1</sub> except that the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* = (x^*, y^* + 1), \perp, 0)$ . Here  $y^* = \tilde{n}$  could be equal to  $\tilde{n}$ . In that case, the special-encryption algorithm can be

Table 8: Computing column components for the sub-hybrid  $\text{Hyb}_{3,\ell,b}^{\sim}$ 

	$C_{y,\ell,b}$	$\tilde{C}_{y,\ell,b}$
$(y > y^*) \vee$ $(y = y^* \wedge \ell \notin [\tilde{\ell}] \cup \{\ell^*\}) \vee$ $(y = y^* \wedge \ell = \tilde{\ell} \wedge b > \tilde{b})$	$H_{y,\ell,b}^t h^{w_{y,\ell,b}\tau}$	$g^{w_{y,\ell,b}}$
$(y < y^*) \vee$ $(y = y^* \wedge \ell \in [\tilde{\ell} - 1] \cup \{\ell^*\}) \vee$ $(y = y^* \wedge \ell = \tilde{\ell} \wedge b \leq \tilde{b})$	$H_{y,\ell,b}^t h^{w_{y,\ell,b}\tau} V_{\ell,b}^{\tau v_{y,\ell,b}}$	$g^{w_{y,\ell,b}}$

directly extended to the encryption of such position.

**Hyb<sub>5</sub>**. This hybrid corresponds to the upper identity-hiding game in which the challenge ciphertext is a special encryption to the index-position-bit tuple  $(i^* + 1, \perp, 0)$ . Note that if  $y^* \neq \tilde{n}$ , then the hybrids 4 and 5 are already identical.

Next, we discuss about the indistinguishability of the above hybrids.

**Hyb<sub>1</sub>  $\approx$  Hyb<sub>2</sub>**: The indistinguishability proof of the hybrids Hyb<sub>1</sub> and Hyb<sub>2</sub> is identical to Claim 7.2 and Lemma 4.

**Hyb<sub>3,\ell,b}^{\sim}  $\approx$  Hyb<sub>3,\ell+b-1,(\tilde{b}+1) \bmod 2}^{\sim}</sub></sub>**: If the modified-1 D3DH assumption 2 holds then there does not exist any PPT adversary that can distinguish between the sub-hybrid  $\text{Hyb}_{3,\ell,b}^{\sim}$  and the sub-hybrid  $\text{Hyb}_{3,\ell+b-1,(\tilde{b}+1) \bmod 2}^{\sim}$  with non-negligible advantage.

If  $\tilde{\ell} = \ell^*$ , sub-hybrid  $\text{Hyb}_{3,\ell,b}^{\sim}$  is identical to sub-hybrid  $\text{Hyb}_{3,\ell+b-1,(\tilde{b}+1) \bmod 2}^{\sim}$ , otherwise there have two cases depending on the secret key query phase.

**Case 1.** Adversary makes a key query for index tuple  $(j, \text{id}, \text{gid}, \mathbf{u})$  such that  $j = i^* \wedge \text{id}_{\tilde{\ell}} = \tilde{b} \wedge \text{gid} = \text{gid}^*$ .

Suppose on contrary, there exists a PPT adversary  $\mathcal{A}$  which can distinguish between the sub-hybrids  $\text{Hyb}_{3,\ell,b}^{\sim}$  and  $\text{Hyb}_{3,\ell+b-1,(\tilde{b}+1) \bmod 2}^{\sim}$  with non-negligible advantage  $\epsilon(\cdot)$ . We construct a PPT reduction algorithm  $\mathcal{B}$  which breaks the modified-1 D3DH assumption 2 with the same non-negligible advantage as

$$(\mathbb{B}\mathbb{G}, g_p, g_q, A = g_p^a, B = g_p^b, C = g_p^c, D = g_p^{b^2}, E = g_p^{b^2c}, F = g_p^{b^3}, G = g_p^{b^4}, H = g_p^{b^3c}, T)$$

where  $T$  is either  $g_p^{abc}$  or a uniformly random element from the sub-group  $\mathbb{G}_p$ . Next,  $\mathcal{B}$  receives a challenge tuple  $(1^\lambda, 1^n, 1^k, 1^{k'}, 1^m, (i^*, \ell^*, b^*), \text{gid}^*)$  from  $\mathcal{A}$ . Then,  $\mathcal{B}$  generates the master public key and sends it to  $\mathcal{A}$ . After seeing  $\text{mpk}$ , the adversary makes polynomial numbers of secret keys for the distinct index positions  $i$  under some admissible conditions. Then  $\mathcal{B}$  simulates the public keys, secret keys and the challenge ciphertext and sends it to  $\mathcal{A}$ . Finally, the adversary outputs a bit  $b'$  as guess which  $\mathcal{B}$  uses to breaks the assumption 2. As the reduction plays the game with its challenger in the subgroup  $\mathbb{G}_p$  so everything it can choose from the subgroup  $\mathbb{G}_q$  by itself. Let us implicitly set the exponents as below

$$t_p = ab; \quad r_{p,x^*,j} = b \cdot \tilde{r}_{p,x^*,j}; \quad c_{p,y^*,\ell^*,b^*} = c + \tilde{c}_{p,y^*,\ell^*,b^*}; \quad \hat{r}_p = b^2, \\ s_{p,x^*} = \tilde{s}_{p,x^*}/b; \quad c_{p,y^*,\tilde{\ell},\tilde{b}} = -c + \tilde{c}_{p,y^*,\tilde{\ell},\tilde{b}}$$

where  $\tilde{r}_{p,x^*,j}, \tilde{s}_{p,x^*}, \tilde{c}_{p,y^*,\tilde{\ell},\tilde{b}}$  and  $\tilde{c}_{p,y^*,\ell^*,b^*}$  are the random exponents. Also, we implicitly set  $h_p = B = g_p^b$  and  $f_p = B^{d_1}$  for  $d_1$  uniformly chosen from  $\mathbb{Z}_N$ . Setting the exponents allows to simulate the public key, secret key exactly as well as the challenge group elements  $T$ , that can be programmed in the challenge ciphertext components,  $C_{y^*,\ell,b}^{\sim}$ .

**Public key simulation.** The challenger  $\mathcal{B}$  chooses random generators  $h_q, f_q \leftarrow \mathbb{G}_q$  such that  $h = h_p h_q = B h_q, f = f_p f_q = B^{d_1} f_q$ . Sets  $h_q = g_q^d, f_q = g_q^{d'}$  for some exponents  $d, d' \in \mathbb{Z}_N$ . Now, the challenger  $\mathcal{B}$  generates the following components:

- **General component:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = g_q^\beta$ .
- **Row-specific components:** For all  $x \in [\tilde{n}], j \in [m]$  sample  $\tilde{r}_{x,j}, \alpha_{x,j}, \psi_{x,j} \leftarrow \mathbb{Z}_N, \hat{r}_q \leftarrow \mathbb{Z}_N$  and computes  $E_{q,x,j} = g_q^{\beta \hat{r}_q \tilde{r}_{x,j}}, F_{q,x,j} = h_q^{\beta \hat{r}_q \tilde{r}_{x,j}}, G_{q,x,j} = e(g_q, g_q)^{\beta \alpha_{x,j}}, W_{q,x,j} = e(f_q, g_q)^{\beta \psi_{x,j}},$

$$E_{x,j} = \begin{cases} (Dg_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{if } x \neq x^*, \\ (Fg_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{otherwise.} \end{cases}, F_{x,j} = \begin{cases} (Fh_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{if } x \neq x^* \\ (Gh_q^{\hat{r}_q})^{\tilde{r}_{x,j}} & \text{otherwise.} \end{cases},$$

$$W_{x,j} = e(B, g_p)^{d_1 \psi_{x,j}} e(f_q, g_q)^{\psi_{x,j}} \quad \forall x$$

$$G_{x,j} = e(g, g)^{\alpha_{x,j}} \quad \forall x$$

- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $\tilde{c}_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$  and computes  $\tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}, V_{\ell,b} = h^{\delta_{\ell,b}},$

$$H_{y,\ell,b} = \begin{cases} Cg^{\tilde{c}_{y,\ell,b}} & \text{if } (y, \ell, b) = (y^*, \ell^*, b^*) \\ C^{-1}g^{\tilde{c}_{y,\ell,b}} & \text{if } (y, \ell, b) = (y^*, \tilde{\ell}, \tilde{b}) \\ g^{\tilde{c}_{y,\ell,b}} & \text{otherwise.} \end{cases}$$

- **gid-specific components:** The challenger  $\mathcal{B}$  samples random group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  samples  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\vartheta'_p = \vartheta'_p \vartheta'_q, \boldsymbol{\vartheta} = (\vartheta_{p,i} \vartheta_{q,i})_{i \in [k']}$ . The challenger  $\mathcal{B}$  sets  $H(\text{gid}) = (\vartheta'_p \vartheta'_q) \prod_{i \in \mathcal{V}} \vartheta_{p,i} \vartheta_{q,i} = \vartheta'_p \prod_{i \in \mathcal{V}} \vartheta_i$  and  $H_q(\text{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  corresponding to any group identity  $\text{gid}$ .

Using the modified-1 D3DH instance, the challenger  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \begin{array}{c} \mathbb{B}\mathbb{G}, g = g_p g_q, h = B h_q, f = B^{d_1} f_q, \vartheta', \vartheta_q^{\beta}, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^{\beta}\}_{i \in [k']}, H, H_q, E_q, \\ \left\{ \begin{array}{c} E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right\}_{(x,j) \in [\tilde{n}] \times [m]}, \\ \{H_{y,\ell,b}\}_{(y,\ell,b) \in [\tilde{n}] \times [k] \times \{0,1\}}, \{\tilde{V}_{\ell,b}, V_{\ell,b}\}_{(\ell,b) \in [k] \times \{0,1\}} \end{array} \right),$$

**Secret key simulation.** To answer the secret key  $\text{sk}_{\mathbf{u}}$  corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$ ,  $\mathcal{B}$  samples a random value  $\tilde{r} \leftarrow \mathbb{Z}_N$  and sets  $r = \tilde{r} \cdot \hat{r}$ . For the challenge group identity  $\text{gid}^*$ , it computes  $H(\text{gid}^*) = g_p^{d_1^*} g_q^{d_2^*}$  where  $d_1^*, d_2^* \leftarrow \mathbb{Z}_N$ . In the secret key query phase,  $\mathcal{A}$  is not allowed to secret key query corresponding to the tuple  $(i^*, \text{id}, \text{gid}^*, \mathbf{u})$  such that  $\text{id}_{\ell^*} \neq b^*$ . Now,  $\mathcal{B}$  simulates the secret key corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  as the Table 9.

To simulate the secret key components  $K_2$  and  $K_3$  for the case  $\text{gid} \neq \text{gid}^*$ , we assume that the adversary makes the maximum number of  $Q$  secret key queries corresponding to the the challenge group identity  $\text{gid}^*$  and challenge index  $i^*$ . Now, the simulator chooses an integer  $k'_1 \leftarrow [k']$ , sets an integer  $s = 10Q$ , a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$  and a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, the simulator also chooses a random value  $w' \leftarrow \mathbb{Z}_N$  and a uniformly random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_N^{k'}$ . All these values are kept secret to the simulator.





Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_\kappa\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  which correspond to the collection of indices  $\mathcal{V}^*$ . Then set  $\sum_{i \in \mathcal{V}^*} z_i = k'_1 s - z'$  for uniformly chosen  $k'_1 \in [k']$ . Now, we define the function  $K(\text{gid})$  as

$$K(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}} z_i \equiv 0 \pmod{s} \\ 1, & \text{otherwise} \end{cases}$$

So, from the above definition of the function  $K$ , we can say that  $K(\text{gid}^*) = 0$  and for all  $\text{gid} \neq \text{gid}^*$  it becomes non-zero. Additionally, we set two functions as  $F(\text{gid}) = N - sk'_1 + z' + \sum_{i \in \mathcal{V}} z_i$  and  $J(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $\vartheta' = f^{N-k'_1 s + z'} \cdot g^{w'}$  and  $\vartheta_i = f^{z_i} g^{w_i} = g_p^{d_p, i} g_q^{d_q, i}$ . Now  $\mathcal{B}$  answers remaining secret key components as

$$\begin{aligned} K_2 &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} \cdot D^{d_1^* r} g_q^{d_2^* r} \tilde{r} \\ &= g^{-\langle \psi_x, \mathbf{u} \rangle \frac{J(\text{gid})}{F(\text{gid})}} H(\text{gid})^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^r \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} \left( f^{F(\text{gid})} g^{J(\text{gid})} \right)^{r'} \\ &= f^{\langle \psi_x, \mathbf{u} \rangle} H(\text{gid})^{r'} \\ K_3 &= (D^{\tilde{r}} g_q^{\tilde{r}}) \cdot g^{-\frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}} = g^{r'} \end{aligned}$$

We implicitly set  $r' = r - \frac{\langle \psi_x, \mathbf{u} \rangle}{F(\text{gid})}$ . So from the construction of  $K$  function, we get  $K(\text{gid}) \neq 0$  for any key query corresponding to the group identity  $\text{gid} \neq \text{gid}^*$ . This implies that the function  $F(\text{gid}) \neq 0 \pmod{N}$  for any such group identity (since we assume  $N > sk'_1$  for reasonable values of  $N, s$  and  $k'_1$ . We prove this in Lemma 9).

There have some restrictions over the key queries to the key generation oracle i.e.,

- Adversary  $\mathcal{A}$  can not query for the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  such that  $i = i^* \wedge \text{id}_{\ell^*} \neq b^* \wedge \text{gid} = \text{gid}^*$ .
- In Case 1, the adversary can make a query for the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  where  $i = i^* \wedge \text{id}_{\tilde{\ell}} = \tilde{b} \wedge \text{gid} = \text{gid}^*$ . If the adversary makes a key query for the challenge index position  $i^*$  with  $\text{id}_{\tilde{\ell}} \neq \tilde{b}$  and  $\text{gid} = \text{gid}^*$ , then the challenger  $\mathcal{B}$  aborts.

**Ciphertext simulation.** The challenger  $\mathcal{B}$  samples the random exponents  $\tau \in \mathbb{Z}_N, t_q \leftarrow \mathbb{Z}_N$  and for the challenge group identity  $\text{gid}^*$ ,  $\mathcal{B}$  computes  $H(\text{gid}^*) = \vartheta' \prod_{i \in \mathcal{V}^*} \vartheta_i = g_p^{d_p^*} g_q^{d_q^*}$  and  $H_q(\text{gid}^*)^\beta = \vartheta_q'^\beta \prod_{i \in \mathcal{V}^*} \vartheta_{q,i}^\beta = g_q^{d_q^*}$  for some random exponents  $d_q^*, d_p^* \in \mathbb{Z}_N$ . Now,  $\mathcal{B}$  simulates the challenge ciphertext components as follows.

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x, \tilde{s}_x \leftarrow \mathbb{Z}_N$  for all  $x \in [\hat{n}]$  and categorize the components according  $x > x^*, x = x^*, x < x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{\tilde{s}_x}, \tilde{R}_{x,j} = F_{q,x,j}^{\tilde{s}_x \tau}$ .
2. *gid-specific component:*  $B_x = H_q(\text{gid}^*)^{\beta \tilde{s}_x t_q}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{\nu_j} \cdot G_{q,x,j}^{\tilde{s}_x t_q} \cdot W_{q,x,j}^{\tilde{s}_x t_q}, A_x = E_q^{\tilde{s}_x t_q}$ .

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = (Dg_q^{\widehat{r}_q})^{\widetilde{s}_x \widetilde{r}_{x,j}}, \widetilde{R}_{x,j} = (Fh_q^{\widehat{r}_q})^{\widetilde{s}_x \widetilde{r}_{x,j} \tau}$ .
2. *gid-specific component:*  $B_x = A^{d_p^* \widetilde{s}_x} g_q^{d_q^* \widetilde{s}_x t_q}$ .
3. *Message-embedding components:*  $I_{x,j} = e(g_q, g_q)^{v_j} e(g, Ag_q^{t_q})^{\widetilde{s}_x \alpha_{x,j}} \cdot e(B, A)^{d_1^2 \psi_{x,j} \widetilde{s}_x} e(f_q, g_q)^{\psi_{x,j} \widetilde{s}_x t_q}, A_x = (Ag_q^{t_q})^{\widetilde{s}_x}$ .

**For  $x < x^*$ :**

1. *Linking components:*  $R_{x,j} = \widetilde{g}^{\widetilde{s}_x \sigma_j}, \widetilde{R}_{x,j} = \widetilde{h}^{\widetilde{s}_x \tau \nu_j}$ .
  2. *gid-specific component:*  $B_x = H(\text{gid}^*)^{d_x}$ .
  3. *Message-embedding components:*  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}, A_x = g^{e_x}$ .
- **Column-specific components:** Sample  $\widetilde{w}_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\widetilde{n}], \ell \in [k], b \in \{0, 1\}$  and generates the components as follows:
 

**For  $(y > y^*) \vee (y = y^* \wedge \ell > \widetilde{\ell}) \vee (y = y^* \wedge \ell \notin [\widetilde{\ell}] \cup \{\ell^*\}) \vee (y = y^* \wedge \ell = \widetilde{\ell} \wedge b > \widetilde{b})$ :**

$$C_{y,\ell,b} = g_q^{\widetilde{c}_{y,\ell,b} t_q} \cdot h^{\widetilde{w}_{y,\ell,b} \tau}, \widetilde{C}_{y,\ell,b} = A^{-\widetilde{c}_{y,\ell,b} / \tau} g^{\widetilde{w}_{y,\ell,b}}$$

**For  $y = y^* \wedge \ell = \widetilde{\ell} \wedge b = \widetilde{b}$ :**  $C_{y,\ell,b} = g_q^{\widetilde{c}_{y,\ell,b} t_q} \cdot h^{\tau \widetilde{w}_{y,\ell,b}} \cdot T^{-1}, \widetilde{C}_{y,\ell,b} = A^{-\widetilde{c}_{y,\ell,b} / \tau} g^{\widetilde{w}_{y,\ell,b}}$

**For  $(y < y^*) \vee (y = y^* \wedge \ell < \widetilde{\ell}) \vee (y = y^* \wedge \ell \in ([\widetilde{\ell} - 1] \cup \{\ell^*\}) \wedge b < \widetilde{b})$ :**  $C_{y,\ell,b} = g_q^{\widetilde{c}_{y,\ell,b} t_q} h^{\tau \widetilde{w}_{y,\ell,b}} g_p^{v_{y,\ell,b}}, \widetilde{C}_{y,\ell,b} = g^{\widetilde{w}_{y,\ell,b}}$

After seeing the challenge ciphertext, the adversary guesses a bit  $b'$  and it forwards to the modified-1 D3DH challenger of assumption 2.

**Analysis of simulation.** If  $T = g_p^{abc}$  then  $\mathcal{B}$  simulates the view of  $\text{Hyb}_{3, \widetilde{\ell} + b - 1, \widetilde{b} + 1 \bmod 2}$  otherwise if  $T$  is a random group element from the subgroup  $\mathbb{G}_p$  then  $\mathcal{B}$  simulates the view of sub-hybrid  $\text{Hyb}_{3, \widetilde{\ell}, \widetilde{b}}$ . Therefore, if the adversary  $\mathcal{A}$  wins the game with an advantage  $\epsilon(\cdot)$  then  $\mathcal{B}$  breaks the assumption 2 with same  $\epsilon(\cdot)$  advantage.

**Case 2.** (otherwise) The proof technique is similar to the proof of Claim 7.2 and Lemma 4.

In this case, we use the same proof strategy as used in Claim 7.2 and Lemma 4, where the reduction algorithm does not need to know the value of the group element  $g^{\widehat{r}_{x^*, \mathbf{u}} c_{y^*, \widetilde{\ell}, \widetilde{b}}}$  for answering the key queries.

**Hyb<sub>3</sub>  $\approx$  Hyb<sub>4</sub>:** The proof of the above indistinguishability of hybrids Hyb<sub>3</sub> and Hyb<sub>4</sub> are identical. No adversary can not distinguish between these hybrids with non-negligible advantage.

**Hyb<sub>4</sub>  $\approx$  Hyb<sub>5</sub>:** The indistinguishable of the hybrids Hyb<sub>4</sub> and Hyb<sub>5</sub> can be classified into two cases.

**Case 1.** If  $y^* \neq \widetilde{n}$ , then both the hybrids Hyb<sub>4</sub> and Hyb<sub>5</sub> are identical.

**Case 2.** For  $y^* = \widetilde{n}$  then the indistinguishability follows from the sequence of hybrid games which is similar to the claim 7.2. In particular, Hyb<sub>4</sub> as described above is similar to the hybrid 2 and Hyb<sub>5</sub> is identical with the hybrid 7 as described in the claim 7.2. Thus this indistinguishable follows from the claim 7.2.

This concludes the upper identity-hiding security game.  $\square$

**Lemma 6.** *If the D3DH assumption 1 holds over the bilinear group  $\mathbb{BG}$ , then our EIPL-IBIPFE satisfies selective message-hiding security as per Definition 8.*

*Proof.* Suppose the adversary  $\mathcal{A}$  is a PPT adversary against the message-hiding security of the our EIPL-IBIPFE scheme. We construct an algorithm  $\mathcal{B}$  for breaking the D3DH assumption 1 that uses  $\mathcal{A}$  as a subroutine. To prove the message-hiding security, we consider two hybrid games. The first hybrid is the same as the original message-hiding security experiment as in definition 8. In the next hybrid, we change the distribution of the master public key, secret key and the challenge ciphertext, where we first sample a random vector  $\tilde{\psi}_x$  and set  $\psi_x = \mathbf{F}^\top \tilde{\psi}_x$  for some  $x \in [\hat{n}]$ . The matrix  $\mathbf{F}$  is a full rank matrix chosen such that  $\mathbf{F}(\mathbf{v}^{(0)} - \mathbf{v}^{(1)}) = \mathbf{e}_1$  where  $\mathbf{v}^{(0)}, \mathbf{v}^{(1)}$  are the challenge message vectors submitted by the  $\mathcal{A}$  and  $\mathbf{e}_1$  denotes a  $m$  length vector as  $(1, 0, \dots, 0)^\top$ . Assuming the D3DH assumption holds over the bilinear group  $\mathbb{B}\mathbb{G}$ , we show that the adversary can distinguish between the challenge ciphertexts with negligible probability.

**Hybrid 0:** This hybrid is the same as the message-hiding security experiment as per Definition 8.

**Hybrid 1:** The hybrid is identical to the previous hybrid except for each identity  $\text{gid}$ , the challenger samples the master secret key  $\text{msk}$  as follows:

- Samples uniformly random vector  $\tilde{\psi}_x = (\tilde{\psi}_{x,1}, \tilde{\psi}_{x,2}, \dots, \tilde{\psi}_{x,m})$  for all  $\tilde{\psi}_{x,j} \in \mathbb{Z}_N$  and  $x \in [\hat{n}]$ .
- Samples a uniformly chosen full rank matrix  $\mathbf{F} \in \mathbb{Z}_N^{m \times m}$  satisfying the relation  $\mathbf{F}(\mathbf{v}^{(0)} - \mathbf{v}^{(1)}) = \mathbf{e}_1$  where  $\mathbf{v}^{(0)}, \mathbf{v}^{(1)}$  are the challenge messages vectors of length  $m$ .
- Sets  $\psi_x = \mathbf{F}^\top \tilde{\psi}_x$  instead of sampling uniformly random as in Hybrid 0.

In the adversary's view, the master public key  $\text{mpk}$ , the secret key associated with the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  where  $i \geq i^*$  and the challenge ciphertext  $\text{ct}_{\mathbf{v}^{(b)}}$  is simulated as below.

**Public key:** All the components of  $\text{mpk}$  are generated similarly as Hybrid 0 except  $W_{q,x,j} = (e(f_q, g_q)^{\beta \mathbf{F} \psi_x})_j$  and  $W_{x,j} = (e(f, g)^{\mathbf{F} \psi_x})_j$  where  $(\mathbf{g})_j$  represents the  $j$ -th group element from the vector  $\mathbf{g} = (g_1, g_2, \dots, g_m)$ .

**Secret key:** We consider  $\mathcal{V} \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the  $i$ -th component of queried group identity is non-zero i.e.,  $\text{gid}_i = 1$ . The secret key  $\text{sk}_{\mathbf{u}}$  components corresponding to the tuple  $(i, \text{id}, \text{gid}, \mathbf{u})$  are set as

$$K_1 = g^{(\tilde{\alpha}_x, \mathbf{u})} \left( \prod_{\ell \in [k]} H_{y, \ell, \text{id}_\ell} \right)^{\hat{r}(r_x, \mathbf{u})}, \quad K_2 = f^{(\tilde{\psi}_x, \mathbf{F} \mathbf{u})} \text{H}(\text{gid})^r, \quad K_3 = g^r$$

where  $r = \tilde{r} \cdot \hat{r}$  with  $\tilde{r} \leftarrow \mathbb{Z}_N$ .

**Challenge ciphertext:** For the challenge group identity  $\text{gid}^*$ , consider  $\mathcal{V}^* \subseteq \{1, \dots, k'\}$  be the set of all  $i$  such that  $\text{gid}_i^* = 1$ . All the components of the ciphertext except  $A_x, B_x, I_{x,j}$  for  $x > x^*$  and  $x = x^*$  are similarly generated as Hybrid 0.

Table 10: Computing row components of the ciphertext for  $x \in [\hat{n}]$

	$R_{x,j}$	$\tilde{R}_{x,j}$	$A_x$	$B_x$	$I_{x,j}$
$x > x^*$	$E_{q,x,j}^{s_x}$	$F_{q,x,j}^{s_x \tau}$	$g^{\beta t s_x}$	$\text{H}(\text{gid}^*)^{t s_x \beta}$	$(e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F} \mathbf{v}^{(0)} - \mathbf{v}^{(0)} - \mathbf{v}^{(1)})}) \cdot e(g_q, g_q)^{\beta s_x t \alpha_x} \cdot e(g_q, f_q)^{\beta t s_x \mathbf{F}^\top \tilde{\psi}_x})_j$
$x = x^*$	$E_{x,j}^{s_x}$	$F_{x,j}^{s_x \tau}$	$g^{t s_x}$	$\text{H}(\text{gid}^*)^{t s_x}$	$(e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F} \mathbf{v}^{(0)} - \mathbf{v}^{(0)} - \mathbf{v}^{(1)})}) \cdot e(g, g)^{s_x t \alpha_x} \cdot e(g, f)^{t s_x \mathbf{F}^\top \tilde{\psi}_x})_j$
$x < x^*$	$g^{s_x \sigma_j}$	$h^{s_x \tau \nu_j}$	$g^{e_x}$	$\text{H}(\text{gid}^*)^{d_x}$	$e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}$

Since,  $\mathbf{F} \in \mathbb{Z}_N^{m \times m}$  is an orthogonal matrix, then the following two distributions are equivalent.

$$\{\psi_x : \psi_x \leftarrow \mathbb{Z}_N^m, x \in [\hat{n}]\} \equiv \{\mathbf{F}^\top \tilde{\psi}_x : \tilde{\psi}_x \leftarrow \mathbb{Z}_N^m, x \in [\hat{n}]\}$$

Suppose on the contrary, there exist a PPT adversary  $\mathcal{A}$  that can distinguish between the Hybrid 0 and Hybrid 1 with non-negligible advantage  $\epsilon(\cdot)$ . Then we construct a PPT adversary  $\mathcal{B}$  that breaks the D3DH assumption 1 with the same non-negligible advantage as follow. The reduction algorithm  $\mathcal{B}$  first receives the bilinear challenge from the challenger as

$$(\mathbb{B}\mathbb{G}, A_q = g_q^a, D_q = g_q^d, C_q = g_q^c, T)$$

of the D3DH assumption 1, where  $\mathbb{B}\mathbb{G} = (p, q, N, \mathbb{G}, \mathbb{G}_T, g, e(\cdot, \cdot))$  is a group description. The elements  $a, d, c \leftarrow \mathbb{Z}_q$  are random integers and the element  $T = g_q^T$  is either  $g_q^{adc}$  or random group element from the subgroup  $\mathbb{G}_q$ . The algorithm  $\mathcal{B}$  works as follows:

The adversary  $\mathcal{B}$  implicitly sets the following vector of length  $m$  as

$$\mathbf{a}_x = (a, a_{x,2}, \dots, a_{x,m}); \mathbf{d} = (d, \dots, d); \mathbf{c} = (c, \dots, c)$$

where it random samples  $a_{x,2}, \dots, a_{x,m} \leftarrow \mathbb{Z}_N$ . We define the notion  $\mathbf{u} \odot \mathbf{v}$  by component wise multiplication of the vectors  $\mathbf{u}$  and  $\mathbf{v}$ . In this case,  $\mathbf{a} \odot \mathbf{d} = (ad, a_{x,2}d, \dots, a_{x,m}d) = \mathbf{d}\mathbf{a}$ . To generate the public key, we implicitly set  $\tilde{\psi}_{q,x} = \mathbf{a}_x, f_q = D_q$  and  $t_q = c$ .

**Public key simulation:** Without loss of generality, we assume that the adversary makes the maximum number of  $Q$  queries and the challenge group identity  $\text{gid}^*$  and challenge index  $i^*$ . Now, the simulator chooses an integer  $k'_1 \leftarrow [k']$ , sets an integer  $s = 10Q$ , a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$  and a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, the simulator also chooses a random value  $w' \leftarrow \mathbb{Z}_N$  and a uniformly random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_N^{k'}$ . All these values are kept secret to the simulator.

Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_\kappa\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  which correspond to the index set  $\mathcal{V}^*$  and then set  $\sum_{i \in \mathcal{V}^*} z_i = k'_1 s - z'$  for uniformly chosen  $k'_1 \in [k']$ . Now, we define the function  $K(\text{gid})$  as

$$K(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}} z_i \equiv 0 \pmod{s} \\ 1, & \text{elsewhere} \end{cases}$$

So, from the above definition of the function  $K$ , we can say that  $K(\text{gid}^*) = 0$  and for all  $\text{gid} \neq \text{gid}^*$  it becomes non-zero. Additionally, we set two functions as  $F(\text{gid}) = N - sk'_1 + z' + \sum_{i \in \mathcal{V}} z_i$  and  $J(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $\vartheta' = f^{N-k'_1 s + z'} \cdot g^{w'} = g_p^{d'_p} g_q^{d'_q}$  and  $\vartheta_i = f^{z_i} g^{w_i} = g_p^{d_{p,i}} g_q^{d_{q,i}}$ . From the adversarial perspective, the distribution of the public parameters are identical to the real construction.

The challenger  $\mathcal{B}$  chooses random generators  $h_q \leftarrow \mathbb{G}_q$  and  $h_p, f_p \leftarrow \mathbb{G}_p$  such that  $h = h_p h_q, f = f_p f_q$ . As mentioned previously, we implicitly set  $f_q = D_q$ . Now, the challenger  $\mathcal{B}$  generates the following components:

- **General components:** Sample  $\beta \leftarrow \mathbb{Z}_N$  and compute  $E_q = g_q^\beta$ .
- **Row-specific components:** For all  $x \in [\tilde{n}], j \in [m]$  sample  $\tilde{r}_{x,j}, \alpha_{x,j}, \tilde{\psi}_{x,j} \leftarrow \mathbb{Z}_N$ ,  $\hat{r} \leftarrow \mathbb{Z}_N$  and computes  $E_{q,x,j} = g_q^{\beta \tilde{r}_{x,j}}, F_{q,x,j} = h_q^{\beta \tilde{r}_{x,j}}, G_{q,x,j} = e(g_q, g_q)^{\beta \alpha_{x,j}}, W_{q,x,j} = (e(D_q, g_q^{\beta \mathbf{F}^\top \mathbf{a}_x}))_j, E_{x,j} = g^{\tilde{r}_{x,j}}, F_{x,j} = h^{\tilde{r}_{x,j}}, G_{x,j} = e(g, g)^{\alpha_{x,j}}, W_{x,j} = (e(f_p D_q, g_q^{\mathbf{F}^\top \mathbf{a}_x} g_p^{\mathbf{F}^\top \tilde{\psi}_{p,x}}))_j$ .
- **Column-specific components:** For all  $y \in [\tilde{n}], \ell \in [k], b \in \{0, 1\}$ , sample  $c_{y,\ell,b}, \delta_{\ell,b} \leftarrow \mathbb{Z}_N, \gamma_{\ell,b} \leftarrow \mathbb{Z}_p$  and computes  $H_{y,\ell,b} = g^{c_{y,\ell,b}}, \tilde{V}_{\ell,b} = g^{\delta_{\ell,b}} g_p^{\gamma_{\ell,b}}, V_{\ell,b} = h^{\delta_{\ell,b}}$ .

- **gid-specific components:** The challenger  $\mathcal{B}$  samples random group elements  $\vartheta'_p \leftarrow \mathbb{G}_p, \vartheta'_q \leftarrow \mathbb{G}_q$  and for all  $i \in [k']$  samples  $\vartheta_{p,i} \leftarrow \mathbb{G}_p, \vartheta_{q,i} \leftarrow \mathbb{G}_q$  such that  $\vartheta^i = \vartheta'_p \vartheta'_{q,i}$ ,  $\boldsymbol{\vartheta} = (\vartheta_{p,i} \vartheta_{q,i})_{i \in [k']}$ . The challenger  $\mathcal{B}$  sets  $\mathbf{H}(\mathbf{gid}) = (\vartheta'_p \vartheta'_{q,i})_{i \in \mathcal{V}}$ ,  $\mathbf{H}_q(\mathbf{gid}) = \vartheta'_q \prod_{i \in \mathcal{V}} \vartheta_{q,i}$  corresponding to any group identity  $\mathbf{gid}$ .

Using the D3DH instance, the challenger  $\mathcal{B}$  sets the master public key as

$$\text{mpk} = \left( \mathbb{B}\mathbb{G}, g = g_p g_q, h = h_p h_q, f = f_p D_q, \vartheta', \vartheta'^{\beta}, \boldsymbol{\vartheta}, \{\vartheta_{q,i}^{\beta}\}_{i \in [k']}, \mathbf{H}, \mathbf{H}_q, E_q, \right. \\ \left. \begin{array}{c} \left\{ E_{q,x,j}, F_{q,x,j}, G_{q,x,j}, W_{q,x,j} \right\} \\ E_{x,j}, F_{x,j}, G_{x,j}, W_{x,j} \end{array} \right)_{(x,j) \in [\widehat{n}] \times [m]}, \\ \left. \left\{ H_{y,\ell,b} \right\}_{(y,\ell,b) \in [\widehat{n}] \times [k] \times \{0,1\}}, \left\{ \tilde{V}_{\ell,b}, V_{\ell,b} \right\}_{(\ell,b) \in [k] \times \{0,1\}} \right)$$

Here, the exponent  $g_q^{\alpha_x}$  is computed as

$$g_q^{\alpha_x} = (g_q^a, g_q^{a_{x,2}}, \dots, g_q^{a_{x,m}}) = g_q^{\tilde{\psi}_{q,x}}$$

**Secret key simulation:**  $\mathcal{B}$  answers the secret key  $\text{sk}_{\mathbf{u}}$  associated to the tuple  $(i, \text{id}, \mathbf{gid}, \mathbf{u})$  as describe below. We consider two cases. Before going further, we consider  $\mathcal{V} \subseteq \{1, \dots, k'\}$  be the set of all  $j$  such that  $\mathbf{gid}_j = 1$ . Let  $i = (x, y)$  be the row wise representation with  $i \geq i^*$ .

**Case 1.** If the group identity  $\mathbf{gid} \neq \mathbf{gid}^*$ ,  $\mathcal{B}$  simulates the secret key as follows:

$$K_1 = g^{\langle \alpha_x, \mathbf{u} \rangle} \left( \prod_{\ell \in [k]} H_{y,\ell, \text{id}_{\ell}} \right)^{\widehat{r} \langle \mathbf{r}_x, \mathbf{u} \rangle}$$

$$K_2 = f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} \left( f_p^{\mathbf{F}(\mathbf{gid})} g_p^{\mathbf{J}(\mathbf{gid})} \right)^{r'_p} g_q^{-\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle \frac{\mathbf{J}(\mathbf{gid})}{\mathbf{F}(\mathbf{gid})}} \mathbf{H}_q(\mathbf{gid})^r$$

$$= f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} \left( f_p^{\mathbf{F}(\mathbf{gid})} g_p^{\mathbf{J}(\mathbf{gid})} \right)^{r'_p} f_q^{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle} \left( f_q^{\mathbf{F}(\mathbf{gid})} g_q^{\mathbf{J}(\mathbf{gid})} \right)^r \left( f_q^{\mathbf{F}(\mathbf{gid})} g_q^{\mathbf{J}(\mathbf{gid})} \right)^{-\frac{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}}$$

$$= f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} \left( f_p^{\mathbf{F}(\mathbf{gid})} g_p^{\mathbf{J}(\mathbf{gid})} \right)^{r'_p} f_q^{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle} \left( f_q^{\mathbf{F}(\mathbf{gid})} g_q^{\mathbf{J}(\mathbf{gid})} \right)^{r - \frac{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}}$$

$$= f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} \left( f_p^{\mathbf{F}(\mathbf{gid})} g_p^{\mathbf{J}(\mathbf{gid})} \right)^{r'_p} f_q^{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle} \left( f_q^{\mathbf{F}(\mathbf{gid})} g_q^{\mathbf{J}(\mathbf{gid})} \right)^{r'_q}$$

$$= f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} f_q^{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle} \left( f_p^{\mathbf{F}(\mathbf{gid})} g_p^{\mathbf{J}(\mathbf{gid})} \right)^{r'_p} \left( f_q^{\mathbf{F}(\mathbf{gid})} g_q^{\mathbf{J}(\mathbf{gid})} \right)^{r'_q}$$

$$= f_p^{\langle \tilde{\psi}_{x,p}, \mathbf{F}\mathbf{u} \rangle} f_q^{\langle \alpha_x, \mathbf{F}\mathbf{u} \rangle} \mathbf{H}(\mathbf{gid})^{r'} = f^{\langle \tilde{\psi}_x, \mathbf{F}\mathbf{u} \rangle} \mathbf{H}(\mathbf{gid})^{r'}$$

$$K_3 = g_p^{r'_p} g_q^{r - \langle (a, a_{x,2}, \dots, a_{x,m}), \mathbf{F}\mathbf{u} \rangle \frac{1}{\mathbf{F}(\mathbf{gid})}}$$

$$= g_p^{r'_p} g_q^{r'_q} = g^{r'}$$

We implicitly set  $r'_q = r - \frac{\langle \tilde{\psi}_{x,q}, \mathbf{F}\mathbf{u} \rangle}{\mathbf{F}(\mathbf{gid})}$  and  $r'_p$  is randomly chosen. So from the construction of K function, it can conclude that  $\mathbf{K}(\mathbf{gid}) \neq 0$  for any key query corresponding to the group identity  $\mathbf{gid} \neq \mathbf{gid}^*$ . This implies that the function  $\mathbf{F}(\mathbf{gid}) \neq 0 \pmod N$  for any such group identities (as we assume  $N > sk'_1$  for reasonable values of  $N, s$  and  $k'_1$ , see Lemma 9.

**Case 2.** If  $\text{gid} = \text{gid}^*$ ,  $\mathcal{B}$  responds the secret keys as follows:

$$\begin{aligned}
K_1 &= g^{\langle \alpha_x, \mathbf{u} \rangle} \left( \prod_{\ell \in [k]} H_{y, \ell, \text{id}_\ell} \right)^{\widehat{r}(\mathbf{r}_x, \mathbf{u})} \\
K_2 &= g_q^{d\mu} f_p^{\langle \widetilde{\psi}_{p,x}, \mathbf{F}\mathbf{u} \rangle} \left( (f^{\mathbf{F}(\text{gid}^*)} g^{\mathbf{J}(\text{gid}^*)})^r \right) \\
&= g_q^{\langle \mathbf{a}_x \odot \mathbf{d}, \mathbf{F}\mathbf{u} \rangle} f_p^{\langle \widetilde{\psi}_{p,x}, \mathbf{F}\mathbf{u} \rangle} \left( (f^{\mathbf{F}(\text{gid}^*)} g^{\mathbf{J}(\text{gid}^*)})^r \right) \\
&= f_q^{\langle (a, a_{x,2}, \dots, a_{x,m}), \mathbf{F}\mathbf{u} \rangle} f_p^{\langle \widetilde{\psi}_{p,x}, \mathbf{F}\mathbf{u} \rangle} \left( (f^{\mathbf{F}(\text{gid}^*)} g^{\mathbf{J}(\text{gid}^*)})^r \right) \\
&= f_q^{\langle \mathbf{a}_x, \mathbf{F}\mathbf{u} \rangle} f_p^{\langle \widetilde{\psi}_{p,x}, \mathbf{F}\mathbf{u} \rangle} \left( (f^{\mathbf{F}(\text{gid}^*)} g^{\mathbf{J}(\text{gid}^*)})^r \right) \\
&= f^{\langle \widetilde{\psi}_x, \mathbf{F}\mathbf{u} \rangle} \text{H}(\text{gid}^*)^r \\
K_3 &= g^r
\end{aligned}$$

where the second equality follows from the fact that  $\langle \mathbf{a}_x \odot \mathbf{d}, \mathbf{F}\mathbf{u} \rangle = d\mu$  with  $\mu \in \mathbb{Z}_N$  is known to the challenger  $\mathcal{B}$ . So from the formation of the matrix  $\mathbf{F}$ , we have  $\mathbf{F}(\mathbf{v}^{(0)} - \mathbf{v}^{(1)}) = \mathbf{e}_1$  and for  $\text{gid} = \text{gid}^*$ , the queried secret key associated with the vector  $\mathbf{u}$  satisfies the condition  $\langle \mathbf{v}^{(0)} - \mathbf{v}^{(1)}, \mathbf{u} \rangle = 0$ . Therefore, we have  $\langle \mathbf{e}_1, \mathbf{F}\mathbf{u} \rangle = 0$  which implies that  $\langle \mathbf{a}_x \odot \mathbf{d}, \mathbf{F}\mathbf{u} \rangle = \langle (ad, a_{x,2}d, a_{x,3}d, \dots, a_{x,m}d), \mathbf{F}\mathbf{u} \rangle = d\mu$  for some  $\mu \in \mathbb{Z}_N$ .

**Challenge ciphertext simulation:** The challenger  $\mathcal{B}$  samples the random exponents  $\tau \in \mathbb{Z}_N, t_p \leftarrow \mathbb{Z}_N$  and simulates the challenge ciphertext components as follows.

- **Row-specific components:** Sample  $\sigma_j, \nu_j, \phi_j \leftarrow \mathbb{Z}_N$  for all  $j \in [m]$  and  $e_x, f_x, d_x, s_x \leftarrow \mathbb{Z}_N$  for all  $x \in [\widehat{n}]$  and categorize the components according  $x > x^*, x = x^*, x < x^*$  as follows:

**For  $x > x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{q,x,j}^{s_x}, \widetilde{R}_{x,j} = F_{q,x,j}^{s_x \tau}$ .
2. *gid-specific component:*  $B_x = C_q^{\beta s_x \mathbf{J}_q(\text{gid}^*)}$ . Since,  $N - k'_1 s + z' + \sum_{i \in \mathcal{V}^*} z_i = N = pq$ , this implies  $f_q^N = 1$ . Therefore  $B_x = \text{H}_q(\text{gid}^*)^{\beta s_x t} = C_q^{\beta s_x \mathbf{J}_q(\text{gid}^*)}$ .
3. *Message-embedding components:*  $A_x = C_q^{\beta s_x}$ ,

$$\begin{aligned}
I_{x,j} &= I_{x,j}^1 = \left( e(g_q, g_q)^{\mathbf{F}^\top(\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{\beta s_x c \alpha_x} e(g_q, g_q^{cd \mathbf{F}^\top(a, a_{x,2}, \dots, a_{x,m})})^{\beta s_x} \right)_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top(\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{\beta s_x c \alpha_x} e(g_q, g_q)^{\beta s_x cd \mathbf{F}^\top(a, a_{x,2}, \dots, a_{x,m})} \right)_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top(\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{\beta s_x c \alpha_x} e(g_q, f_q)^{\beta s_x c \mathbf{F}^\top(a, a_{x,2}, \dots, a_{x,m})} \right)_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top(\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{\beta s_x t \alpha_x} e(g_q, f_q)^{\beta s_x t \mathbf{F}^\top \widetilde{\psi}_x} \right)_j
\end{aligned}$$

**For  $x = x^*$ :**

1. *Linking components:*  $R_{x,j} = E_{x,j}^{s_x}, \widetilde{R}_{x,j} = F_{x,j}^{s_x \tau}$ .
2. *gid-specific component:*  $B_x = (C_q g_p^{t_p})^{s_x \mathbf{J}(\text{gid}^*)}$ .

3. *Message-embedding components*:  $A_x = (C_q g_p^{t_p})^{s_x}$ ,

$$\begin{aligned}
I_{x,j} &= I_{x,j}^2 = \left( e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{s_x c \alpha_x} e(g_p, g_p)^{s_x t_p \alpha_x} \right. \\
&\quad \left. e(g_q, g_q)^{cd \mathbf{F}^\top (a, a_x, 2, \dots, a_x, m)} \right)^{s_x} e(g_p, f_p)^{s_x t_p \mathbf{F}^\top \tilde{\psi}_{p,x}} \Big|_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{s_x c \alpha_x} \right. \\
&\quad \left. e(g_p, g_p)^{s_x t_p \alpha_x} e(g_q, g_q)^{s_x c d \mathbf{F}^\top (a, a_x, 2, \dots, a_x, m)} e(g_p, f_p)^{s_x t_p \mathbf{F}^\top \tilde{\psi}_{p,x}} \right)_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{s_x c \alpha_x} \right. \\
&\quad \left. e(g_p, g_p)^{s_x t_p \alpha_x} e(g_q, f_q)^{s_x c \mathbf{F}^\top (a, a_x, 2, \dots, a_x, m)} e(g_p, f_p)^{s_x t_p \mathbf{F}^\top \tilde{\psi}_{p,x}} \right)_j \\
&= \left( e(g_q, g_q)^{\mathbf{F}^\top (\mathbf{F}\mathbf{v}^{(0)} - \mathbf{b}\mathbf{e}_1)} e(g_q, g_q)^{s_x t_q \alpha_x} e(g_p, g_p)^{s_x t_p \alpha_x} \right. \\
&\quad \left. e(g_q, f_q)^{s_x t_q \mathbf{F}^\top \tilde{\psi}_{x,q}} e(g_p, f_p)^{s_x t_p \mathbf{F}^\top \tilde{\psi}_{p,x}} \right)_j
\end{aligned}$$

**For  $x < x^*$ :**

1. *Linking components*:  $R_{x,j} = g^{s_x \sigma_j}$ ,  $\tilde{R}_{x,j} = h^{s_x \tau \nu_j}$ .
2. *gid-specific component*:  $B_x = \mathbf{H}(\text{gid}^*)^{d_x}$ .
3. *Message-embedding components*:  $I_{x,j} = e(g, g)^{f_x \phi_j} \cdot e(f, f)^{f_x \phi_j}$ ,  $A_x = g^{e_x}$ .

- **Column-specific components**: Sample  $w_{y,\ell,b}, v_{y,\ell,b} \leftarrow \mathbb{Z}_N$  for all  $y \in [\tilde{n}]$ ,  $\ell \in [k]$ ,  $b \in \{0, 1\}$  and generates the components as follows:

**For  $(y > y^*) \vee (y = y^* \wedge (\ell, b) \neq (\ell^*, b^*))$ :**  $C_{y,\ell,b} = (C_q g_p^{t_p})^{c_{y,\ell,b}} \cdot h^{w_{y,\ell,b} \tau}$ ,  $\tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}}$ .

**For  $(y < y^*) \vee ((y, \ell, b) = (y^*, \ell^*, b^*))$ :**  $C_{y,\ell,b} = (C_q g_p^{t_p})^{c_{y,\ell,b}} \cdot h^{w_{y,\ell,b} \tau} \cdot V_{\ell,b}^{v_{y,\ell,b} \tau}$ ,  $\tilde{C}_{y,\ell,b} = g^{w_{y,\ell,b}} \cdot \tilde{V}_{\ell,b}^{v_{y,\ell,b}}$ .

**Guess.** If  $\mathcal{A}$  guesses the challenge bit  $\mathbf{b} \leftarrow \{0, 1\}$  correctly, then  $\mathcal{B}$  returns 1 otherwise it outputs 0. We consider  $\mathbf{w}_x = dc(a, a_x, 2, \dots, a_x, m) = (\tau, dca_x, 2, \dots, dca_x, m)$  where  $g_q^\tau$  are the challenge elements. If  $\tau = adc$ , then all the secret key and the challenge ciphertext is properly distributed. In particular, the challenge ciphertext is an encryption of message vector  $\mathbf{v}^{(\mathbf{b})}$ . Therefore, in this case,  $\mathcal{A}$  outputs  $\mathbf{b} = \mathbf{b}'$  with advantages  $\frac{1}{2} + \text{negl}(\lambda)$  where  $\text{negl}(\lambda)$  is the advantage of  $\mathcal{A}$  in the message-hiding security game of the EIPL-IBIPFE. Otherwise, if  $\tau$  is randomly generated from  $\mathbb{Z}_q$  then the challenge ciphertext components  $I_{x,j}$  uniform element from the target group  $\mathbb{G}_T$ . So the  $\mathcal{A}$  can not get any information about the challenge bit  $\mathbf{b}$  from this component. So,  $\mathcal{A}$  wins the game with the probability  $\frac{1}{2}$ . Hence, from the hardness of assumption 1, it can conclude that  $\mathcal{A}$  has a non-negligible advantage against the proposed EIPL-IBIPFE scheme achieves the selective security. This completes the message-hiding security.  $\square$

Therefore, it concludes the security of our EIPL-IBIPFE.  $\square$

**EI-TIBIPFE from Pairing.** Combining Theorem 4 and 3, we obtain the following corollary.

**Corollary 1.** *If the assumptions 1, 2, 3, 5, 6 and 7 hold over a bilinear group  $\mathbb{BG}$ , there exists a selectively secure EI-TIBIPFE with the following properties:*

- *The size of the master public key is  $m \cdot \sqrt{n \cdot k} \cdot \text{poly}(\lambda)$*
- *The size of the secret key is  $\log n + k + k' + \text{poly}(\lambda)$*
- *The size of the ciphertext is  $m \cdot \sqrt{n \cdot k} \cdot \text{poly}(\lambda)$*

where  $n$  denotes the number of users in the system,  $m$  is the length of input vector and  $k, k'$  are the parameters of user identity and group identity spaces respectively.



## References

- [ABCP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, volume 9020 of *LNCS*, pages 733–751. Springer, 2015. URL: [https://doi.org/10.1007/978-3-662-46447-2\\_33](https://doi.org/10.1007/978-3-662-46447-2_33).
- [ABP<sup>+</sup>17] Shweta Agrawal, Sanjay Bhattacharjee, Duong Hieu Phan, Damien Stehlé, and Shota Yamada. Efficient public trace and revoke from standard assumptions: Extended abstract. In Bhavani Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 2277–2293. ACM, 2017. URL: <https://doi.org/10.1145/3133956.3134041>.
- [ACGU20] Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, 2020. URL: [https://doi.org/10.1007/978-3-030-64840-4\\_16](https://doi.org/10.1007/978-3-030-64840-4_16).
- [ADM<sup>+</sup>07] Michel Abdalla, Alexander W. Dent, John Malone-Lee, Gregory Neven, Duong Hieu Phan, and Nigel P. Smart. Identity-based traitor tracing. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings*, volume 4450 of *LNCS*, pages 361–376. Springer, 2007. URL: [https://doi.org/10.1007/978-3-540-71677-8\\_24](https://doi.org/10.1007/978-3-540-71677-8_24).
- [AGT21] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II*, volume 13043 of *LNCS*, pages 224–255. Springer, 2021. URL: [https://doi.org/10.1007/978-3-030-90453-1\\_8](https://doi.org/10.1007/978-3-030-90453-1_8).
- [ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, 2016. URL: [https://doi.org/10.1007/978-3-662-53015-3\\_12](https://doi.org/10.1007/978-3-662-53015-3_12).
- [BB04] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004. URL: [https://doi.org/10.1007/978-3-540-24676-3\\_14](https://doi.org/10.1007/978-3-540-24676-3_14).
- [BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances*

- in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *LNCS*, pages 440–456. Springer, 2005. URL: [https://doi.org/10.1007/11426639\\_26](https://doi.org/10.1007/11426639_26).
- [BF99] Dan Boneh and Matthew K. Franklin. An efficient public key traitor tracing scheme. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *LNCS*, pages 338–353. Springer, 1999. URL: [https://doi.org/10.1007/3-540-48405-1\\_22](https://doi.org/10.1007/3-540-48405-1_22).
- [BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters' IBE scheme. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *LNCS*, pages 407–424. Springer, 2009. URL: [https://doi.org/10.1007/978-3-642-01001-9\\_24](https://doi.org/10.1007/978-3-642-01001-9_24).
- [BSW06] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *LNCS*, pages 573–592. Springer, 2006. URL: [https://doi.org/10.1007/11761679\\_34](https://doi.org/10.1007/11761679_34).
- [BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *LNCS*, pages 253–273. Springer, 2011. URL: [https://doi.org/10.1007/978-3-642-19571-6\\_16](https://doi.org/10.1007/978-3-642-19571-6_16).
- [BW06] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, pages 211–220. ACM, 2006. URL: <https://doi.org/10.1145/1180405.1180432>.
- [BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *LNCS*, pages 480–499. Springer, 2014. URL: [https://doi.org/10.1007/978-3-662-44371-2\\_27](https://doi.org/10.1007/978-3-662-44371-2_27).
- [CFN94] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *LNCS*, pages 257–270. Springer, 1994. URL: [https://doi.org/10.1007/3-540-48658-5\\_25](https://doi.org/10.1007/3-540-48658-5_25).
- [CPP05] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public traceability in traitor tracing schemes. In Ronald Cramer, editor, *Advances in Cryptology*

- *EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *LNCS*, pages 542–558. Springer, 2005. URL: [https://doi.org/10.1007/11426639\\_32](https://doi.org/10.1007/11426639_32).
- [CVW<sup>+</sup>18] Yilei Chen, Vinod Vaikuntanathan, Brent Waters, Hoeteck Wee, and Daniel Wichs. Traitor-tracing from LWE made simple and attribute-based. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *LNCS*, pages 341–369. Springer, 2018. URL: [https://doi.org/10.1007/978-3-030-03810-6\\_13](https://doi.org/10.1007/978-3-030-03810-6_13).
- [DKW21] Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority ABE for dnfs from LWE. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *LNCS*, pages 177–209. Springer, 2021. URL: [https://doi.org/10.1007/978-3-030-77870-5\\_7](https://doi.org/10.1007/978-3-030-77870-5_7).
- [DPP20] Xuan Thanh Do, Duong Hieu Phan, and David Pointcheval. Traceable inner product functional encryption. In Stanislaw Jarecki, editor, *Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24-28, 2020, Proceedings*, volume 12006 of *LNCS*, pages 564–585. Springer, 2020. URL: [https://doi.org/10.1007/978-3-030-40186-3\\_24](https://doi.org/10.1007/978-3-030-40186-3_24).
- [Fre10] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *LNCS*, pages 44–61. Springer, 2010. URL: [https://doi.org/10.1007/978-3-642-13190-5\\_3](https://doi.org/10.1007/978-3-642-13190-5_3).
- [FT01] Amos Fiat and Tamir Tassa. Dynamic traitor tracing. *J. Cryptol.*, 14(3):211–223, 2001. URL: <https://doi.org/10.1007/s00145-001-0006-7>.
- [GKRW18] Rishab Goyal, Venkata Koppula, Andrew Russell, and Brent Waters. Risky traitor tracing and new differential privacy negative results. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *LNCS*, pages 467–497. Springer, 2018. URL: [https://doi.org/10.1007/978-3-319-96884-1\\_16](https://doi.org/10.1007/978-3-319-96884-1_16).
- [GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 660–670. ACM, 2018. URL: <https://doi.org/10.1145/3188745.3188844>.
- [GKW19] Rishab Goyal, Venkata Koppula, and Brent Waters. New approaches to traitor tracing with embedded identities. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019*,

- Nuremberg, Germany, December 1-5, 2019, *Proceedings, Part II*, volume 11892 of *LNCS*, pages 149–179. Springer, 2019. URL: [https://doi.org/10.1007/978-3-030-36033-7\\_6](https://doi.org/10.1007/978-3-030-36033-7_6).
- [GMS12] Fuchun Guo, Yi Mu, and Willy Susilo. Identity-based traitor tracing with short private key and short ciphertext. In Sara Foresti, Moti Yung, and Fabio Martinelli, editors, *Computer Security - ESORICS 2012 - 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012. Proceedings*, volume 7459 of *LNCS*, pages 609–626. Springer, 2012. URL: [https://doi.org/10.1007/978-3-642-33167-1\\_35](https://doi.org/10.1007/978-3-642-33167-1_35).
- [KD98] Kaoru Kurosawa and Yvo Desmedt. Optimum traitor tracing and asymmetric schemes. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *LNCS*, pages 145–157. Springer, 1998. URL: <https://doi.org/10.1007/BFb0054123>.
- [KY02a] Aggelos Kiayias and Moti Yung. Traitor tracing with constant transmission rate. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *LNCS*, pages 450–465. Springer, 2002. URL: [https://doi.org/10.1007/3-540-46035-7\\_30](https://doi.org/10.1007/3-540-46035-7_30).
- [KY02b] Aggelos Kiayias and Moti Yung. Traitor tracing with constant transmission rate. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *LNCS*, pages 450–465. Springer, 2002. URL: [https://doi.org/10.1007/3-540-46035-7\\_30](https://doi.org/10.1007/3-540-46035-7_30).
- [KY02c] Kaoru Kurosawa and Takuya Yoshida. Linear code implies public-key traitor tracing. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*, volume 2274 of *LNCS*, pages 172–187. Springer, 2002. URL: [https://doi.org/10.1007/3-540-45664-3\\_12](https://doi.org/10.1007/3-540-45664-3_12).
- [LAWH22] Fucui Luo, Saif M. Al-Kuwari, Haiyan Wang, and Weihong Han. Generic construction of trace-and-revoke inner product functional encryption. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *Computer Security - ESORICS 2022 - 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26-30, 2022, Proceedings, Part I*, volume 13554 of *LNCS*, pages 259–282. Springer, 2022. URL: [https://doi.org/10.1007/978-3-031-17140-6\\_13](https://doi.org/10.1007/978-3-031-17140-6_13).
- [LPSS14] San Ling, Duong Hieu Phan, Damien Stehlé, and Ron Steinfeld. Hardness of k-lwe and applications in traitor tracing. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *LNCS*, pages 315–334. Springer, 2014. URL: [https://doi.org/10.1007/978-3-662-44371-2\\_18](https://doi.org/10.1007/978-3-662-44371-2_18).

- [NWZ16] Ryo Nishimaki, Daniel Wichs, and Mark Zhandry. Anonymous traitor tracing: How to embed arbitrary information in a key. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *LNCS*, pages 388–419. Springer, 2016. URL: [https://doi.org/10.1007/978-3-662-49896-5\\_14](https://doi.org/10.1007/978-3-662-49896-5_14).
- [PT11] Duong Hieu Phan and Viet Cuong Trinh. Identity-based trace and revoke schemes. In Xavier Boyen and Xiaofeng Chen, editors, *Provable Security - 5th International Conference, ProvSec 2011, Xi'an, China, October 16-18, 2011. Proceedings*, volume 6980 of *LNCS*, pages 204–221. Springer, 2011. URL: [https://doi.org/10.1007/978-3-642-24316-5\\_15](https://doi.org/10.1007/978-3-642-24316-5_15).
- [SP19] Edouard Dufour Sans and David Pointcheval. Unbounded inner-product functional encryption with succinct keys. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *Applied Cryptography and Network Security - 17th International Conference, ACNS 2019, Bogota, Colombia, June 5-7, 2019, Proceedings*, volume 11464 of *LNCS*, pages 426–441. Springer, 2019. URL: [https://doi.org/10.1007/978-3-030-21568-2\\_21](https://doi.org/10.1007/978-3-030-21568-2_21).
- [SSW01] Jessica Staddon, Douglas R. Stinson, and Ruizhong Wei. Combinatorial properties of frameproof and traceability codes. *IEEE Trans. Inf. Theory*, 47(3):1042–1049, 2001. URL: <https://doi.org/10.1109/18.915661>.
- [SW98] Douglas R. Stinson and Ruizhong Wei. Combinatorial properties and constructions of traceability schemes and frameproof codes. *SIAM J. Discret. Math.*, 11(1):41–53, 1998. URL: <https://doi.org/10.1137/S0895480196304246>.
- [TT01] Wen-Guey Tzeng and Zhi-Jia Tzeng. A public-key traitor tracing scheme with revocation using dynamic shares. In Kwangjo Kim, editor, *Public Key Cryptography, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001, Cheju Island, Korea, February 13-15, 2001, Proceedings*, volume 1992 of *LNCS*, pages 207–224. Springer, 2001. URL: [https://doi.org/10.1007/3-540-44586-2\\_16](https://doi.org/10.1007/3-540-44586-2_16).
- [Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *LNCS*, pages 114–127. Springer, 2005. URL: [https://doi.org/10.1007/11426639\\_7](https://doi.org/10.1007/11426639_7).
- [Zha21] Mark Zhandry. White box traitor tracing. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *LNCS*, pages 303–333. Springer, 2021. URL: [https://doi.org/10.1007/978-3-030-84259-8\\_11](https://doi.org/10.1007/978-3-030-84259-8_11).

## A Additional Preliminaries

### A.1 Bilinear Groups

A bilinear group  $\mathbb{B}\mathbb{G} = (p, q, N = p \cdot q, \mathbb{G}, \mathbb{G}_T, g, e(\cdot, \cdot))$  consists of the two primes  $p, q$ , two multiplicative (source and target) groups  $\mathbb{G}, \mathbb{G}_T$  (respectively) with the order  $|\mathbb{G}| = |\mathbb{G}_T| = N$ ,  $g$  as the generator of the group  $\mathbb{G}$  and a bilinear map  $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ . It satisfies the following:

- *bilinearity*:  $e(g^a, g^b) = e(g, g)^{ab}$  for all  $g \in \mathbb{G}$ ,  $a, b \in \mathbb{Z}_N$  and
- *non-degeneracy*:  $e(g, g)$  is a generator of  $\mathbb{G}_T$ .

A bilinear group generator  $\mathcal{G}_{\text{BG.Gen}}(1^\lambda)$  takes input the security parameter  $\lambda$  and outputs a bilinear group  $\mathbb{B}\mathbb{G} = (p, q, N, \mathbb{G}, \mathbb{G}_T, g, e(\cdot, \cdot))$  with a  $\lambda$ -bit composite integer  $N = p \cdot q$ . We consider  $\mathbb{G}_p$  and  $\mathbb{G}_q$  as the subgroups of  $\mathbb{G}$  and their orders  $p$  and  $q$  respectively.

### A.2 Identity-Based Inner Product Functional Encryption

An IBIPFE scheme for a identity space<sup>1</sup>  $\mathcal{GID} = \{\{0, 1\}^{k'} : k' \in \mathbb{N}\}$  and a message/key space  $\mathbb{Z}^m$  consists of four PPT algorithms  $\text{IBIPFE} = (\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Dec})$ . The details about these algorithms are given below.

- **Setup**( $1^\lambda, 1^{k'}, 1^m$ )  $\rightarrow$  (msk, mpk): On input the security parameter  $\lambda$ , a length  $k'$  of identities and a vector length  $m$  (as unary), the trusted authority generates a master secret key msk and a master public key mpk.
- **KeyGen**(msk, gid,  $\mathbf{u}$ )  $\rightarrow$   $\text{sk}_{\mathbf{u}}$ : The trusted authority takes as input the master secret key msk, an identity  $\text{gid} \in \{0, 1\}^{k'}$ , a vector  $\mathbf{u} \in \mathbb{Z}^m$ , and outputs a secret key  $\text{sk}_{\mathbf{u}}$ .
- **Enc**(mpk, gid',  $\mathbf{v}$ )  $\rightarrow$   $\text{ct}_{\mathbf{v}}$ : The encryption algorithm takes as input the master public key mpk, an identity  $\text{gid}' \in \{0, 1\}^{k'}$  and a message vector  $\mathbf{v} \in \mathbb{Z}^m$ . It outputs a ciphertext  $\text{ct}_{\mathbf{v}}$ .
- **Dec**( $\text{sk}_{\mathbf{u}}, \text{ct}_{\mathbf{v}}$ )  $\rightarrow$   $\zeta/\perp$ : The decryption algorithm uses a secret key  $\text{sk}_{\mathbf{u}}$  to decrypt the ciphertext  $\text{ct}_{\mathbf{v}}$ . It either outputs a decrypted value  $\zeta$  on successful decryption or a symbol  $\perp$  indicating decryption failure.
- **Correctness**. An IBIPFE = (Setup, KeyGen, Enc, Dec) scheme is said to be correct if for all  $\lambda, k, m \in \mathbb{N}$ ,  $\mathbf{u}, \mathbf{v} \in \mathbb{Z}^m$ , identity  $\text{gid} \in \{0, 1\}^{k'}$ , there exists a *negligible* function  $\text{negl}$  such that the following holds

$$\Pr \left[ \begin{array}{l} (\text{msk}, \text{mpk}) \leftarrow \text{Setup}(1^\lambda, 1^{k'}, 1^m) \\ \text{Dec}(\text{sk}_{\mathbf{u}}, \text{ct}_{\mathbf{v}}) = \langle \mathbf{u}, \mathbf{v} \rangle : \quad \begin{array}{l} \text{sk}_{\mathbf{u}} \leftarrow \text{KeyGen}(\text{msk}, \text{gid}, \mathbf{u}) \\ \text{ct}_{\mathbf{v}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}, \mathbf{v}) \end{array} \end{array} \right] \geq 1 - \text{negl}(\lambda),$$

where the probability is taken over the random coins of Setup, KeyGen and Enc of IBIPFE.

- **Security**. We consider selective security notions of IBIPFE. We build a selectively secure IBIPFE scheme based on a target-group-based assumption (Appendix C).

**Definition 9** (Selective security of IBIPFE). An IBIPFE is said to satisfy selective indistinguishable-based (Sel-IND-CPA) security if for any security parameter  $\lambda \in \mathbb{N}$ , any PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}$  such that the following holds:

$$\Pr[\text{Expt}_{\mathcal{A}, \text{Sel-IND-CPA}}^{\text{IBIPFE}}(\lambda) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

<sup>1</sup>Note that, in existing work [ACGU20], the identity is usually denoted by  $\text{id}$ . However, for the shake of consistency, we use  $\text{gid}$  instead of  $\text{id}$  in this paper.

where the experiment  $\text{Expt}_{\mathcal{A}, \text{Sel-IND-CPA}}^{\text{IBIPFE}}(\lambda)$  is defined as follows:

1.  $(\text{gid}^*, \mathbf{v}^{(0)}, \mathbf{v}^{(1)}) \leftarrow \mathcal{A}(1^\lambda)$
2.  $(\text{msk}, \text{mpk}) \leftarrow \text{Setup}(1^\lambda, 1^{k'}, 1^m)$
3.  $\mathbf{b} \leftarrow \{0, 1\}$
4.  $\text{ct}_{\mathbf{v}^{(\mathbf{b})}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(\mathbf{b})})$
5.  $\mathbf{b}' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{msk}, \cdot, \cdot)}(\text{ct}_{\mathbf{v}^{(\mathbf{b})}})$
6. Output 1 if  $\mathbf{b} = \mathbf{b}'$  else 0.

Figure 10:  $\text{Expt}_{\mathcal{A}, \text{Sel-IND-CPA}}^{\text{IBIPFE}}(\lambda)$

In the experiment, all the key queries of  $\mathcal{A}$  to the KeyGen oracle should be of form  $(\text{gid}, \mathbf{u})$  satisfying  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  if  $\text{gid} = \text{gid}^*$ .

*Remark 4 (IPFE).* If we omit the identity from the above syntax of IBIPFE, then it yields the primitive of *inner product functional encryption (IPFE)*.

## B Security Analysis of EI-TIBIPFE from EIPL-IBIPFE

In the following theorem, we show that the above transformation yields an EI-TIBIPFE scheme with the same security level as in the underlying EIPL-IBIPFE scheme.

**Theorem 4.** *If our EIPL-IBIPFE scheme is 1-query secure as per Definitions 4 to 8 (in the adaptive/selective model), then the above EI-TIBIPFE scheme is secure as per Definitions 2 and 3 (in the adaptive/selective model).*

*Proof.* We prove the Theorem 4 by combining the following Theorem 5 and Theorem 6 in the adaptive model and the proof in selective setting will follow similarly.

**Theorem 5 (Security of indistinguishability).** *If our EIPL-IBIPFE be an adaptively 0-query secure as per Definitions 4 to 8, then our EI-TIBIPFE scheme is adaptively secure as per Definition 2.*

*Proof.* We would like to point out that the scheme EI-TIBIPFE is IND-CPA secure even if the EIPL-IBIPFE scheme satisfies only 0-query security. Let us assume that  $\tau$  is the least integer index queried by the adversary of EI-TIBIPFE to the KeyGen oracle and  $(\text{gid}^*, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$  be the adversary's challenge tuple. As per Definition 2, all secret key queries of  $\mathcal{A}$  to the KeyGen oracle is of the form  $(j, \text{id}, \text{gid}, \mathbf{u})$  where  $j \geq \tau$  and satisfying the relation  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  whenever  $\text{gid} = \text{gid}^*$ . Further all the secret keys are associated with distinct indices.

We construct a sequence of  $2\tau + 3$  hybrid experiments to prove the theorem. The sequence of hybrids starts with the hybrid  $H_0$  and ends with hybrid  $H_0$ , which are exactly the same as IND-CPA game of EI-TIBIPFE where the challenger encrypts  $\mathbf{v}^{(0)}$  and  $\mathbf{v}^{(1)}$  respectively. For any PPT adversary  $\mathcal{A}$ , let  $p_{\mathcal{A}, x}(\cdot)$  be a function of  $\lambda$  that denotes the probability of  $\mathcal{A}$  outputting the challenge bit in Hybrid  $H_x$ . It is sufficient to show that  $|p_{\mathcal{A}, 0} - p_{\mathcal{A}, 2}|$  is negligible in  $\lambda$ .

**Hybrid  $H_0$ :** It is the real IND-CPA game where the challenger computes the ciphertext  $\text{ct}_{\mathbf{v}^{(0)}} \leftarrow \text{EIPL-IBIPFE.Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(0)})$  and sends it to  $\mathcal{A}$ .

**Hybrid  $H_{i,0}$  (for  $i \in [\tau]$ ):** This hybrid is identical to the previous hybrid, except that the challenge ciphertext is a special encryption of  $\mathbf{v}^{(0)}$  to the tuple  $(i, \perp, 0)$ , i.e.,  $\text{ct}_{\mathbf{v}^{(0)}} \leftarrow \text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(0)}, (i, \perp, 0))$ .

Since the indices associate to the queried secret keys are all distinct, the normal-hiding security of EIPL-IBIPFE guarantees that  $\text{EIPL-IBIPFE.Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(0)})$  and  $\text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(0)}, (1, \perp, 0))$  are computationally indistinguishable except with a negligible advantage. Therefore, for any PPT adversary  $\mathcal{A}$ , it holds that  $|p_{\mathcal{A},0} - p_{\mathcal{A},1,0}| \leq \text{negl}(\lambda)$ .

For  $i \in [\tau-1]$ , by index-hiding security, the distributions  $\text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(0)}, (i, \perp, 0))$  and  $\text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(0)}, (i+1, \perp, 0))$  are computationally indistinguishable since the  $\mathcal{A}$  is not allowed to query a secret key with  $j = i$ . Therefore, we have  $|p_{\mathcal{A},i,0} - p_{\mathcal{A},i+1,0}| \leq \text{negl}(\lambda)$  for  $i \in [\tau-1]$  and by the property of triangular inequality, it holds that  $|p_{\mathcal{A},0} - p_{\mathcal{A},\tau,0}| \leq \text{negl}(\lambda)$ .

**Hybrid  $H_1$ :** This hybrid is identical to the hybrid  $H_{\tau,0}$  except that the challenge ciphertext is generated as  $\text{ct}_{\mathbf{v}^{(1)}} \leftarrow \text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(1)}, (\tau, \perp, 0))$ .

Observe that, for the index  $\tau$ , the adversary may obtain some secret keys generated for  $\text{gid}^*$  to decrypt the ciphertext of hybrids  $H_{\tau,0}$  and  $H_1$ . However, by the restriction on such secret key queries, we have  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$ . Hence, the message-hiding security of EIPL-IBIPFE ensures that  $|p_{\mathcal{A},1} - p_{\mathcal{A},\tau,0}| \leq \text{negl}(\lambda)$ .

**Hybrid  $H_{\tau-i+1,1}$  (for  $i \in [\tau]$ ):** This experiment is identical to the previous hybrid  $H_1$  except that the adversary gets the challenge ciphertext  $\text{ct}_{\mathbf{v}^{(1)}} \leftarrow \text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}^*, \mathbf{v}^{(1)}, (\tau-i+1, \perp, 0))$  corresponding to the index-position-bit tuple  $(\tau-i+1, \perp, 0)$ . For  $i = 1$ , the hybrid  $H_{\tau,1}$  is exactly identical with the hybrid  $H_1$ .

Also as before, the index-hiding security of EIPL-IBIPFE ensures that  $|p_{\mathcal{A},\tau-i+1,1} - p_{\mathcal{A},\tau-i,1}| \leq \text{negl}(\lambda)$  for each  $i \in [\tau-1]$ .

**Hybrid  $H_2$ :** This hybrid is similar to hybrid  $H_{1,1}$  expect the challenge ciphertext is of the form  $\text{ct}_{\mathbf{v}^{(1)}} \leftarrow \text{EIPL-IBIPFE.Enc}(\text{mpk}, \text{gid}^*, \mathbf{v}^{(1)})$ . Again, by the normal-hiding security of EIPL-IBIPFE, we have  $|p_{\mathcal{A},1,1} - p_{\mathcal{A},2}| \leq \text{negl}(\lambda)$ .

Finally, combining the above claims and using the triangular inequality, we conclude the proof.  $\square$

**Theorem 6** (Security of Tracing). *If our EIPL-IBIPFE scheme is an adaptively 1-query secure as per the Definitions 4 to 8, then our EI-TIBIPFE is adaptively secure as per the Definition 3.*

*Proof. Correctness of Tracing.* Next, we show that the false trace probability is bounded by a negligible function and the correct trace probability is close to the probability of  $\mathcal{A}$  outputting an  $\epsilon$ -successful decoding box for some non-negligible  $\epsilon(\cdot)$ . This proof technique is inspired from the Goyal et al. [GKW19] tracing mechanism.

Let us consider the following notations for the further proof of this Theorem. Given any pirate decoder box  $\mathcal{D}_{\mathbf{u}}$  with a key vector  $\mathbf{u}$  and message vectors pair  $\mathbf{v}^{(0)}, \mathbf{v}^{(1)}$  for all  $i \in [n+1], \ell \in [k]$ , suppose

$$\begin{aligned} p_{i,\perp}^{\mathcal{D}_{\mathbf{u}}} &= \Pr \left[ \begin{array}{l} \mathcal{D}_{\mathbf{u}}(\text{ct}_{\mathbf{v}^{(b)}}) = \mathbf{b} : \\ \text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}, \mathbf{v}^{(b)}, (i, \perp, 0)), \\ \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \\ p_{i,\ell}^{\mathcal{D}_{\mathbf{u}}} &= \Pr \left[ \begin{array}{l} \mathcal{D}_{\mathbf{u}}(\text{ct}_{\mathbf{v}^{(b)}}) = \mathbf{b} : \\ \text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{EIPL-IBIPFE.SplEnc}(\text{key}, \text{gid}, \mathbf{v}^{(b)}, (i, \ell, 0)), \\ \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \\ p_{\text{norm}}^{\mathcal{D}_{\mathbf{u}}} &= \Pr \left[ \begin{array}{l} \mathcal{D}_{\mathbf{u}}(\text{ct}_{\mathbf{v}^{(b)}}) = \mathbf{b} : \\ \text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{EIPL-IBIPFE.Enc}(\text{mpk}, \text{gid}, \mathbf{v}^{(b)}), \\ \mathbf{b} \leftarrow \{0, 1\} \end{array} \right] \end{aligned}$$

The above probabilities are computed over the random coins of the decoder  $\mathcal{D}_{\mathbf{u}}$  as well as the randomness used in the encryption algorithm.



**False Trace Probability.** Now we prove that probability of the false tracing by the Trace algorithm is negligible. Now, we prove the following Lemma.

**Lemma 7.** *If the scheme EIPL-IBIPFE is an adaptively 1-query secure as per Definitions 4 to 8, then for every PPT adversary  $\mathcal{A}$ , polynomial  $q(\cdot)$  and non-negligible function  $\epsilon(\cdot)$ , there exists a negligible function  $\text{negl}(\cdot)$  such that for all  $\lambda \in \mathbb{N}$  satisfying  $\epsilon(\lambda) > 1/q(\lambda)$ ,*

$$\Pr\text{-Fal-Tr}_{\mathcal{A},\epsilon}(\lambda) \leq \text{negl}(\lambda)$$

where  $\Pr\text{-Fal-Tr}_{\mathcal{A},\epsilon}(\cdot)$  is defined in Definition 3.

*Proof.* Let  $S \subseteq [n] \times \{0, 1\}^k$  be the set of index-identity query tuples by the adversary  $\mathcal{A}$  for secret keys and  $S_{\text{index}}$  be the set of indices queried by the adversary  $\mathcal{A}$  for secret keys and let  $\mathcal{D}_{\mathbf{u}}$  be the decoder box output by  $\mathcal{A}$ . For  $i \in [n], \ell \in [k]$  and  $\text{gid} = \text{gid}^*$ , we define events

$$\begin{aligned} A_i^{\mathcal{D}_{\mathbf{u}}} &: p_{i,\perp}^{\mathcal{D}_{\mathbf{u}}} - p_{i+1,\perp}^{\mathcal{D}_{\mathbf{u}}} > \epsilon/8n \\ B_{i,\ell,\text{lwr}}^{\mathcal{D}_{\mathbf{u}}} &: p_{i,\perp}^{\mathcal{D}_{\mathbf{u}}} - p_{i,\ell}^{\mathcal{D}_{\mathbf{u}}} > \epsilon/16n \\ C_{i,\ell,\text{upr}}^{\mathcal{D}_{\mathbf{u}}} &: p_{i,\ell}^{\mathcal{D}_{\mathbf{u}}} - p_{i+1,\perp}^{\mathcal{D}_{\mathbf{u}}} > \epsilon/16n \end{aligned}$$

$$\text{Diff-Adv}^{\mathcal{D}_{\mathbf{u}}} : \bigvee_{i \in [n] \setminus S_{\text{index}}} A_i^{\mathcal{D}_{\mathbf{u}}} \bigvee_{(i,\text{id}) \in S, \ell \in [k] \text{ s.t. } \text{id}_\ell = 1} B_{i,\ell,\text{lwr}}^{\mathcal{D}_{\mathbf{u}}} \bigvee_{(i,\text{id}) \in S, \ell \in [k] \text{ s.t. } \text{id}_\ell = 0} C_{i,\ell,\text{upr}}^{\mathcal{D}_{\mathbf{u}}}$$

For simplicity of notations, we will drop dependence on decoder  $\mathcal{D}_{\mathbf{u}}$  whenever clear from context. Next, note that the probability of the event *false trace* can be rewritten (using union bound) as follows by conditioning on the events defined above

$$\begin{aligned} \Pr[\text{Fal-Tr}] &\leq \Pr[\text{Fal-Tr} | \overline{\text{Diff-Adv}}] + \sum_{i \in [n]} \Pr[i \notin S_{\text{index}} \wedge A_i] \\ &+ \sum_{(i,\ell) \in [n] \times [k]} \Pr \left[ \exists \text{id} \in \{0, 1\}^k \text{ s.t. } (i, \text{id}) \in S \wedge \left( (B_{i,\ell,\text{lwr}} \wedge \text{id}_\ell = 1) \vee (C_{i,\ell,\text{upr}} \wedge \text{id}_\ell = 0) \right) \right] \end{aligned}$$

The following claims show that a negligible function bounds each term.

**Claim.** *For every PPT adversary  $\mathcal{A}$ , there exists a negligible function  $\text{negl}_1(\cdot)$  such that for all  $\lambda \in \mathbb{N}$ ,  $\Pr[\text{Fal-Tr} | \overline{\text{Diff-Adv}}] \leq \text{negl}_1(\lambda)$ .*

*Proof.* Here we give a high level sketch of proof. The proof follows from Chernoff bounds and similar to Lemma 4.4 of [GKW18], and Lemma 5.3 of [GKRW18]. Note that the tracing algorithm outputs a user identity which was not allowed to key query by the adversary iff the event *Fal-Tr* occurs. As discussed before, the tracing mechanism first trace the indices of the corrupted keys then trace their corresponding identities. There are two sources of error in incorrect tracing. First, during step one of tracing the algorithm (i.e., in *Index-Trace*) might incorrectly include some index  $i \notin S_{\text{index}}$  in the traitor's index-set  $T^{\text{index}}$ . In the second phase of the tracing procedure (i.e., in *ID-Trace*), it may happen that this outputs a non-corrupt identity  $\text{id}$  for some index  $i \in S_{\text{index}}$ , that is for some  $i \in S_{\text{index}}$  the *ID-Trace* algorithm traces the  $\text{id}$  incorrectly at least one bit position. By using the union bound, we can represent it as follows: (recall that  $T$  and  $T^{\text{index}}$  are introduced in the description of Trace algorithm)

$$\begin{aligned} \Pr[\text{Fal-Tr} | \overline{\text{Diff-Adv}}] &\leq \sum_{i \in [n]} \Pr[\text{Fal-Tr} \wedge i \notin S_{\text{index}} \wedge (\exists p, q : (i, p, q) \in T^{\text{index}}) | \overline{\text{Diff-Adv}}] \\ &+ \sum_{(i,\ell) \in [n] \times [k]} \Pr[\text{Fal-Tr} \wedge \exists \text{id}, \tilde{\text{id}} : (i, \text{id}) \in S \wedge \tilde{\text{id}} \in T \wedge \text{id}_\ell \neq \tilde{\text{id}}_\ell | \overline{\text{Diff-Adv}}]. \end{aligned}$$

In the above inequality, the first term on the right side bounds the type 1 error (i.e., faulty step one tracing) and the second term bounds the type 2 error (i.e., faulty step two tracing). Now we explicitly discuss about the first term. Note that, if event

$$\overline{\text{Diff-Adv}} \text{ occurs} \implies \forall i \notin S_{\text{index}}, \overline{A}_i \text{ occurred}$$

Therefore, for every  $i \in [n]$

$$\Pr[i \notin S_{\text{index}} \wedge (\exists p, q : (i, p, q) \in T^{\text{index}}) | \overline{\text{Diff-Adv}}] \leq 2^{-O(\lambda)}.$$

Using Chernoff bound, we can argue that  $\overline{A}_i$  provides  $p_{i,\perp} - p_{i+1,\perp} \leq \epsilon/8n$  and event  $(\exists p, q : (i, p, q) \in T^{\text{index}})$  implies that  $\hat{p}_{i,\perp} - \hat{p}_{i+1,\perp} > \epsilon/4n$  where  $\hat{p}$  denotes the corresponding estimate computed by the tracing algorithm.

We now concentrate the second term. For a fixed index-position pair  $(i, \ell)$  corresponding a particular event where the ID-Trace algorithm outputs a traitor identity  $\tilde{\text{id}}$  such that  $\tilde{\text{id}}_\ell (\neq \text{id}_\ell)$  for the secret key queries corresponding to the index-identity pair  $(i, \text{id})$ . Note that, in each index position  $\overline{A}$  is allowed to ask at most one secret key query. Therefore, by conditioning on the event  $\overline{\text{Diff-Adv}}$  we get that for every  $(i, \text{id}) \in S, \ell \in [k]$ , event  $\overline{B}_{i,\ell,\text{low}}$  occurs if  $\text{id}_\ell = 1$  else event  $\overline{C}_{i,\ell,\text{upr}}$  occurs.

Therefore, for all  $(i, \text{id}) \in S, \ell \in [k]$  the following probability always satisfy:

$$\Pr[\exists \text{id}, \tilde{\text{id}} : (i, \text{id}) \in S \wedge \tilde{\text{id}} \in T \wedge \text{id}_\ell \neq \tilde{\text{id}}_\ell | \overline{\text{Diff-Adv}}] \leq 2^{-O(\lambda)}.$$

If we assume that  $(i, \text{id}, \ell)$  and let  $\text{id}_\ell = 1$ . From the Chernoff bound the above inequality holds as since we know that the event  $\overline{B}_i$  occurs thus we have  $\hat{p}_{i,\perp} - \hat{p}_{i,\ell} \leq \epsilon/16n$  and the event  $\tilde{\text{id}} \in T \wedge \tilde{\text{id}}_\ell = 0$  suggests that  $\hat{p}_{i,\perp} - \hat{p}_{i,\ell} > \epsilon/8n$ . Therefore, combining all the above inclusion we get that

$$\Pr[\text{Fal-Tr} | \overline{\text{Diff-Adv}}] \leq n \cdot 2^{-O(\lambda)} + n \cdot k \cdot 2^{-O(\lambda)} = \text{negl}_1(\lambda)$$

□

**Claim.** *If our EIPL-IBIPFE is an adaptively 1-query index-hiding secure as per Definition 5, then for every PPT adversary  $\mathcal{A}$ , polynomial  $q(\cdot)$  and non-negligible function  $\epsilon(\cdot)$ , there exists a negligible function  $\text{negl}_2(\cdot)$  such that for all  $\lambda \in \mathbb{N}$  satisfying  $\epsilon(\lambda) > 1/q(\lambda)$  and  $i \in [n]$ ,*

$$\Pr[i \notin S_{\text{index}} \wedge A_i] \leq \text{negl}_2(\lambda),$$

where  $n$  is the index bound chosen, and  $S_{\text{index}}$  is the set of indices queried by  $\mathcal{A}$ .

*Proof.* The proof of this claim follows from the Lemma 4.5 of [GKW18] and Lemma 5.4 of [GKRW18]. □

**Claim.** *If our EIPL-IBIPFE scheme is an adaptively 1-query lower and upper identity-hiding secure as per Definitions 6 and 7, then for every PPT adversary  $\mathcal{A}$ , polynomial  $q(\cdot)$  and non-negligible function  $\epsilon(\cdot)$ , there exists a negligible function  $\text{negl}_3(\cdot)$  such that for all  $\lambda \in \mathcal{N}$  satisfying  $\epsilon(\lambda) > 1/q(\lambda)$  and  $i \in [n], \ell \in [k]$ ,*

$$\Pr \left[ \exists \text{id} \in \{0, 1\}^k \text{ s.t. } (i, \text{id}) \in S \wedge \left( (B_{i,\ell,\text{low}} \wedge \text{id}_\ell = 1) \vee (C_{i,\ell,\text{upr}} \wedge \text{id}_\ell = 0) \right) \right] \leq \text{negl}_3(\lambda).$$

where  $n$  is the index bound chosen, and  $S$  is the set of all  $(i, \text{id})$  pairs queried by the adversary  $\mathcal{A}$ .

*Proof.* Suppose there exists a PPT adversary  $\mathcal{A}$ , polynomial  $q(\cdot)$  and non-negligible function  $\epsilon(\cdot)$ ,  $\delta(\cdot)$  such that for all  $\lambda \in \mathbb{N}$  satisfying  $\epsilon(\lambda) > 1/q(\lambda)$  and there exists  $i' \in [n]$ ,  $\ell' \in [k]$ ,

$$\Pr \left[ \exists \text{id} \in \{0, 1\}^k \text{ s.t. } (i', \text{id}) \in S \wedge \left( (B_{i', \ell', \text{lower}} \wedge \text{id}_{\ell'} = 1) \vee (C_{i', \ell', \text{upper}} \wedge \text{id}_{\ell'} = 0) \right) \right] \geq \delta(\lambda).$$

Then we can use  $\mathcal{A}$  to build a PPT reduction algorithm  $\mathcal{B}$  that breaks the upper/lower identity hiding security property of EIPL-IBIPFE. The reduction algorithm  $\mathcal{B}$  first receives  $1^n, 1^k, 1^{k'}, 1^m$  from the adversary. It chooses a index  $i \leftarrow [n]$ , position  $\ell \in [k]$ , and bit  $b \in \{0, 1\}$ , and sends the challenge index-position-bit tuple  $(i, \ell, 0)$  and  $(1^n, 1^k, 1^{k'}, 1^m)$  to the EIPL-IBIPFE challenger. Note that if the reduction algorithm randomly guesses  $(i', \ell')$  if  $b = 0$  then it interacts with the EIPL-IBIPFE lower-identity-hiding challenger, else for  $b = 1$ , it interacts with EIPL-IBIPFE upper identity-hiding challenger. It then receives the EIPL-IBIPFE public key  $\text{mpk}$  from the challenger, which it sends to  $\mathcal{A}$ . Then  $\mathcal{A}$  makes secret key queries for the tuple  $(j, \text{id}, \text{gid}, \mathbf{u})$ , if  $j = i$ ,  $\text{id}_\ell = b$  then  $\mathcal{B}$  aborts and sends a random bit as guess bit to the EIPL-IBIPFE challenger. Else, on key query for  $(j, \text{id}, \text{gid}, \mathbf{u})$  from  $\mathcal{A}$ , the reduction algorithm  $\mathcal{B}$  forwards  $(j, \text{id}, \text{gid}, \mathbf{u})$  to the EIPL-IBIPFE challenger, EIPL-IBIPFE's challenger generates his response and sends it to  $\mathcal{B}$  then it forwards the challengers response to the adversary. The adversary outputs the challenge tuple  $(\text{gid}^*, \mathbf{v}^{(0)}, \mathbf{v}^{(1)})$  with a decoding box  $\mathcal{D}_{\mathbf{u}}$  to  $\mathcal{B}$  and then  $\mathcal{B}$  chooses two random bits  $\alpha, \beta \leftarrow \{0, 1\}$ . Then,  $\mathcal{B}$  sends message vector  $\mathbf{v}^{(\alpha)}$  as its challenge message, and receives challenge ciphertext  $\text{ct}^*$  from EIPL-IBIPFE challenger. It also queries the EIPL-IBIPFE challenger for a special-encryption of  $\mathbf{v}^{(\alpha)}$  to the index-position-bit tuple  $(i, \ell, 0)$  if  $\beta = 0$ , else for  $(i + b, \perp, 0)$ . Let  $\text{ct}$  be the challenger's response. Finally,  $\mathcal{B}$  runs decoder box  $\mathcal{D}_{\mathbf{u}}$  on  $\text{ct}$  and  $\text{ct}^*$  independently, and if  $\mathcal{D}_{\mathbf{u}}(\text{ct}) = \mathcal{D}_{\mathbf{u}}(\text{ct}^*)$ , it outputs  $b' = \beta$ , else it outputs  $b' = 1 - \beta$  as it guess. In the upper identity/lower identity-hiding security, consider  $\mathcal{B}$  is an admissible adversary. In other words if  $b = 0$  then  $\mathcal{B}$  is admissible adversary of lower identity-hiding else it upper identity-hiding security respectively. As  $\mathcal{B}$  does not query for the secret key corresponding to the tuple  $(j, \text{id}, \text{gid}, \mathbf{u})$  such that  $j = i$ ,  $\text{id}_\ell = b$  and  $\text{gid} = \text{gid}^*$ . Also,  $\mathcal{B}$  can make only one query to the special encryption oracle to the index-position-bit tuple  $(i, \ell, 0)$  and  $(i + b, \perp, 0)$ . Therefore, from the Lemma 4.1 and 4.5 of [GKW18] and Lemma 5.4 of [GKRW18], we compute the advantage of the reduction algorithm is at least  $\frac{\delta}{2kn} \cdot (\frac{\epsilon}{16n})^2$ . Thus the claim follows.  $\square$

Therefore, it follows that the probability of the false trace is at most  $\text{negl}_1(\lambda) + n \cdot \text{negl}_2(\lambda) + nk \cdot \text{negl}_3(\lambda)$ .  $\square$

**Correct trace probability.** In the following, we show that if the adversary outputs a good decoder, then the tracing algorithm outputs a non-empty set  $T$  with negligible probability. Combining this with Lemma 7 leads to the conclusion that the tracing is correct. We provide a formal reduction for the correct trace in the following.

**Lemma 8.** *If our EIPL-IBIPFE is an adaptively 1-query secure as per Definitions 4 to 8, then for every PPT adversary  $\mathcal{A}$ , polynomial  $q(\cdot)$ , there exist a negligible function  $\text{negl}$  such that  $\lambda \in \mathbb{N}, \epsilon(\lambda) > 1/q(\lambda)$  such that*

$$\Pr\text{-Cor-Tr}_{\mathcal{A}, \epsilon}(\lambda) \geq \Pr\text{-G-D}_{\mathcal{A}, \epsilon}(\lambda) - \text{negl}(\lambda)$$

where  $\Pr\text{-Cor-Tr}_{\mathcal{A}, \epsilon}(\cdot)$  and  $\Pr\text{-G-D}_{\mathcal{A}, \epsilon}(\lambda)$  are defined in Definition 3.

*Proof.* First we analysis that the tracing algorithm outputs a non-empty index set  $T$ . As it is known that when the event Good-Decoder occurs, then  $p_{\text{nrml}}^{\mathcal{D}_{\mathbf{u}}}(\lambda) \geq 1/2 + \epsilon$  for some non-negligible function  $\epsilon(\cdot)$ . Let  $S_{\text{index}} \subseteq [n]$  be the set of indices  $i \in [n]$  such that  $p_{i, \perp}^{\mathcal{D}_{\mathbf{u}}} - p_{i+1, \perp}^{\mathcal{D}_{\mathbf{u}}} > \epsilon/2n$ . Similar to Claim B, for all  $i \in S_{\text{index}}$  we have

$$\Pr[\hat{p}_{i, \perp}^{\mathcal{D}_{\mathbf{u}}} - \hat{p}_{i+1, \perp}^{\mathcal{D}_{\mathbf{u}}} < \frac{\epsilon}{4n}] \leq \frac{1}{2^{\mathcal{O}(\lambda)}} = \text{negl}_1(\lambda) \quad [\text{using Chernoff bound}] \quad (19)$$

where  $\hat{p}$  represents the corresponding estimate evaluated by tracing algorithm. The goal is to show that  $S_{\text{index}}$  is non-empty when the Good-Decoder event occurs. To show this, we apply normal-hiding and message-hiding security.

From the 1-query normal-hiding security, we have

$$p_{\text{nrml}}^{\mathcal{D}_u} - p_{1,\perp}^{\mathcal{D}_u} \leq \text{negl}_2(\lambda)$$

for some negligible function  $\text{negl}_2(\cdot)$ . Now, we apply the message-hiding security for  $i^* = n + 1$  (see Definition 8). The security experiment produces  $\text{ct}_{\mathbf{v}^{(b)}} \leftarrow \text{SplEnc}(\cdot, \cdot, \mathbf{v}^{(b)}, (i^*, \perp, 0))$  and it says that an adversary is admissible if all secret key queries  $(i \in [n], \text{id}, \text{gid}^*, \mathbf{u})$  of the adversary must satisfy the condition  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  only for  $i \geq i^*$ . This implies that for  $i^* = n + 1$  there is no index  $i \in [n]$  such that  $i \geq i^*$ . Thus, we do not essentially need the condition  $\langle \mathbf{u}, \mathbf{v}^{(0)} \rangle = \langle \mathbf{u}, \mathbf{v}^{(1)} \rangle$  for such an adversary who selects  $i^* = n + 1$ . Therefore, for some negligible function  $\text{negl}_3(\cdot)$ , we can write

$$p_{n+1,\perp}^{\mathcal{D}_u} \leq 1/2 + \text{negl}_3(\lambda).$$

For some negligible functions  $\text{negl}_2, \text{negl}_3$ . Therefore, we can write

$$p_{1,\perp}^{\mathcal{D}_u} - p_{n+1,\perp}^{\mathcal{D}_u} \geq \epsilon - \text{negl}_2(\lambda) - \text{negl}_3(\lambda) > \epsilon/2.$$

Given this we can conclude that the set  $S_{\text{index}} \neq \phi$  whenever the event Good-Decoder occurs. So from the above Equation 19, it can be concluded that whenever Good-Decoder occurs then with all-but-negligible probability

$$T^{\text{index}} \neq \phi \wedge [\forall (i, p, q) \in T^{\text{index}} : p - q > \epsilon/4n]$$

From Fig. 9, we observe that for every  $(i, p, q)$  tuple, from ID-Trace algorithm, it outputs some identity  $\text{id}$ . Since for all  $\ell \in [k]$ , either  $p_{i,\ell}^{\mathcal{D}_u} > (p+q)/2$  then the algorithm put  $\text{id}_\ell = 1$  otherwise, it sets  $\text{id}_\ell = 0$ . Therefore,  $T^{\text{index}} \neq \phi \implies T \neq \phi$ . Hence, it follows that

$$\Pr[T \neq \phi] \geq (1 - n \cdot \text{negl}_1(\lambda)) \cdot \Pr\text{-G-D}_{\mathcal{A},\epsilon}(\lambda) \geq \Pr\text{-G-D}_{\mathcal{A},\epsilon}(\lambda) - \text{negl}(\lambda).$$

From Lemma 7, we conclude that

$$\Pr\text{-Cor-Tr}_{\mathcal{A},\epsilon}(\lambda) \geq \Pr\text{-G-D}_{\mathcal{A},\epsilon}(\lambda) - \text{negl}(\lambda)$$

This conclude the proof.  $\square$

The concludes the proof of EI-TIBIPFE security Theorem 4.  $\square$

## C IBIPFE from DBDH

In this section, we present our construction of IBIPFE from the DBDH assumption. Technically, we extend the framework of Water's IBE [Wat05] and add inner product functionality into it for building our IBIPFE. Similar to all previous group-based IPFE constructions, we assume that the inner product value  $\langle \mathbf{x}, \mathbf{y} \rangle$  belongs to a polynomial range so that the decryptor can efficiently recover  $\langle \mathbf{x}, \mathbf{y} \rangle$  via a discrete log computation. In the following, we describe our IBIPFE = (Setup, KeyGen, Enc, Dec) scheme based on pairings.

- **Setup**( $1^\lambda, 1^{k'}, 1^m$ ): The setup algorithm performs as follows:
  - Consider a bilinear group  $\mathbb{B}\mathbb{G} \leftarrow \mathcal{G}_{\text{BG,Gen}}(1^\lambda)$  where  $\mathbb{B}\mathbb{G} = (p, g, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot))$  and  $\mathbb{G}$  is a prime-order group with order  $p$ .
  - Choose a random generator  $g \in \mathbb{G}$  and a random group element  $g_2$  from the group  $\mathbb{G}$ .
  - Set  $g_{1,i} = g^{\psi_i}$  and  $g_{2,i} = g_2^{\psi_i}$  for all  $i \in [m]$ , where  $\psi_i$ 's are the random exponents chosen from  $\mathbb{Z}_p$ .

- Additionally, the authority chooses a random group element  $u' \in \mathbb{G}$  and a random  $k'$ -length vector  $\mathbf{u} = (u_i) \in \mathbb{G}^{k'}$  whose elements are chosen at random from  $\mathbb{G}$ .
- Output the master public key  $\text{mpk} = (\{g_{1,i}\}_{i=1}^m, g_2, u', \mathbf{u}, g)$  and the master secret key  $\text{msk} = (\{g_{2,i}\}_{i=1}^m)$ .
- **KeyGen(msk, gid,  $\mathbf{y}$ ):** The key generation algorithm executes the following steps:
  - Let  $\text{gid}$  be an  $k'$ -bit string representing an identity, where  $\text{gid}_i$  denotes the  $i$ -th bit of  $\text{gid}$  and  $\mathcal{V} \subseteq \{1, 2, \dots, k'\}$  be set of all  $i$  for which  $\text{gid}_i = 1$ . We consider an identity encoding function  $H : \mathcal{GID} \rightarrow \mathbb{G}$  be defined as  $H(\text{gid}) = u' \prod_{j \in \mathcal{V}} u_j$  for  $\text{gid} \in \mathcal{GID}$  where  $\mathcal{GID}$  be the set of identities.
  - Sample  $r \leftarrow \mathbb{Z}_p$ .
  - Output the secret key  $\text{sk}_{\mathbf{y}} = (d_1, d_2)$  where  $d_1 = g_2^{\langle \psi, \mathbf{y} \rangle} H(\text{gid})^r$ ,  $d_2 = g^r$ .
- **Enc(mpk, gid',  $\mathbf{x}$ ):** The encryption algorithm works as follows:
  - Sample a random value  $t$  from  $\mathbb{Z}_p$ .
  - Output the ciphertext  $\text{ct}_{\mathbf{x}}$  as

$$\text{ct}_{\mathbf{x}} = (C_1 = e(g, g_2)^{\psi t + \mathbf{x}}, C_2 = g^t, C_3 = H(\text{gid}')^t)$$

- **Dec(sk $_{\mathbf{y}}$ , ct $_{\mathbf{x}}$ ):** The decryptor uses the secret key  $\text{sk}_{\mathbf{y}}$  to decrypt the ciphertext  $\text{ct}_{\mathbf{x}}$ . It outputs either  $\zeta = \log_{e(g, g_2)} \eta$  where

$$\eta = \langle C_1, \mathbf{y} \rangle \cdot \frac{e(d_2, C_3)}{e(d_1, C_2)}$$

or outputs  $\perp$ .

## C.1 Correctness

If  $\text{gid} = \text{gid}'$ , then we have

$$\begin{aligned} \eta &= \langle C_1, \mathbf{y} \rangle \cdot \frac{e(d_2, C_3)}{e(d_1, C_2)} \\ &= e(g, g_2)^{\langle \psi t + \mathbf{x}, \mathbf{y} \rangle} \cdot \frac{e(g^r, H(\text{gid})^t)}{e(g_2^{\langle \psi, \mathbf{y} \rangle} H(\text{gid})^r, g^t)} \\ &= e(g, g_2)^{t \langle \psi, \mathbf{y} \rangle} \cdot e(g, g_2)^{\langle \mathbf{x}, \mathbf{y} \rangle} \cdot \frac{e(g, H(\text{gid}))^{rt}}{e(g_2, g)^{t \langle \psi, \mathbf{y} \rangle} e(H(\text{gid}), g)^{rt}} \\ &= e(g, g_2)^{\langle \mathbf{x}, \mathbf{y} \rangle} \end{aligned}$$

Therefore, the correctness follows.

## C.2 Security Analysis

We prove the security of our IBIPFE scheme below.

**Theorem 7.** *If the plain DBDH assumption 4 holds over the bilinear group  $\mathbb{BG}$ , then our IBIPFE scheme is selectively secure as per the Definition 9.*

*Proof.* Suppose  $\mathcal{A}$  be a PPT adversary against the selective security of our IBIPFE scheme. We construct an algorithm  $\mathcal{B}$  for breaking the DBDH assumption that uses  $\mathcal{A}$  as a subroutine. To prove Theorem 7, we consider two hybrid games. The first hybrid is the same as the original selective security experiment of IBIPFE as per Definition 9. In the next hybrid, we change the distribution of the master public key, secret keys, and the challenge ciphertext where we first sample a random vector  $\tilde{\psi}$  and set  $\psi = \mathbf{F}^\top \tilde{\psi}$ . The matrix  $\mathbf{F}$  is a full rank matrix chosen such that  $\mathbf{F}(\mathbf{x}^{(0)} - \mathbf{x}^{(1)}) = (1, 0, \dots, 0)^\top$  where  $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}$  are the challenge message vectors submitted by the  $\mathcal{A}$ . Assuming DBDH holds in the bilinear

group  $\mathbb{B}\mathbb{G}$ , we show that the adversary has a negligible advantage in distinguishing between the challenge ciphertexts.

**Hybrid 0:** This hybrid is exactly same as selective security experiment of IBIPFE.

**Hybrid 1:** This hybrid is same as Hybrid 0 except for each identity, the challenger samples the master secret key  $\text{msk}$  as follows:

- (a) Sample uniformly random vector  $\tilde{\boldsymbol{\psi}} = (\tilde{\psi}_1, \tilde{\psi}_2, \dots, \tilde{\psi}_m)$  for each  $\tilde{\psi}_i \in \mathbb{Z}$ .
- (b) Sample a full rank matrix  $\mathbf{F} \in \mathbb{Z}_p^{m \times m}$  satisfying the relation  $\mathbf{F}(\mathbf{x}^{(0)} - \mathbf{x}^{(1)}) = (1, 0, \dots, 0)^\top$  where  $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}$  are the challenge message vectors of length  $m$ .
- (c) Set  $\boldsymbol{\psi} = \mathbf{F}^\top \tilde{\boldsymbol{\psi}}$  instead of sampling uniformly random as in Hybrid 0.

In adversary's view, the master public key  $\text{mpk}$ , the secret key  $\text{sk}_{\mathbf{y}}$  is associated with a key vector  $\mathbf{y}$  with an identity  $\text{gid}$  and the challenge ciphertext  $\text{ct}_{\mathbf{x}^{(b)}} \leftarrow \text{Enc}(\text{mpk}, \text{gid}^*, \mathbf{x}^{(b)})$  are simulated as follows:

**Public key:**  $\text{mpk} = g^{\mathbf{F}^\top \tilde{\boldsymbol{\psi}}}$ .

**Secret key:** For the secret key query corresponding to the identity  $\text{gid}$  and key vector  $\mathbf{y}$ , we consider  $\mathcal{V} \subseteq \{1, \dots, k'\}$  be the set of all  $i$  for which  $\text{gid}_i = 1$ . Then, the secret key

$$\text{sk}_{\mathbf{y}} = \left( g_2^{(\tilde{\boldsymbol{\psi}} \cdot \mathbf{F} \mathbf{y})} \text{H}(\text{gid})^r, g^r \right)$$

**Challenge ciphertext:** For challenge identity  $\text{gid}^*$ , let  $\mathcal{V}^* \subseteq \{1, \dots, k'\}$  be the set of all  $i$  for which  $\text{gid}_i^* = 1$ .

$$\begin{aligned} C_1 &= e(g, g_2)^{\boldsymbol{\psi} t + \mathbf{b}(\mathbf{x}^{(1)} - \mathbf{x}^{(0)}) + \mathbf{x}^{(0)}} & C_2 &= g^t \\ &= e(g, g_2)^{\mathbf{F}^\top (\tilde{\boldsymbol{\psi}} t - \mathbf{b} \mathbf{e}_1 + \mathbf{F} \mathbf{x}^{(0)})} & C_3 &= \text{H}(\text{gid}^*)^t \end{aligned}$$

Since,  $\mathbf{F} \in \mathbb{Z}_p^{m \times m}$  is an orthogonal matrix, then the following two distributions are identical.

$$\{\boldsymbol{\psi} : \boldsymbol{\psi} \leftarrow \mathbb{Z}_p^m\} \equiv \{\mathbf{F}^\top \tilde{\boldsymbol{\psi}} : \tilde{\boldsymbol{\psi}} \leftarrow \mathbb{Z}_p^m\}$$

Therefore, the advantage of any PPT adversary  $\mathcal{A}$  in distinguishing between Hybrid 0 and Hybrid 1 is negligible in the security parameter  $\lambda$ .

Without loss of generality, we assume that the adversary makes maximum  $Q$  number of secret keys queries and the challenge identity  $\text{gid}^*$ . In this case, the simulator chooses a random integer  $\hat{k} \leftarrow [k']$  and sets an integer  $s = 10Q$ . Then, it chooses a random  $k'$ -length vector  $\mathbf{z} = (z_i) \leftarrow \mathbb{Z}_s^{k'}$  and a value  $z' \leftarrow \mathbb{Z}_s$ . Additionally, simulator also chooses a random value  $w' \leftarrow \mathbb{Z}_p$  and a random  $k'$ -length vector  $\mathbf{w} = (w_i) \leftarrow \mathbb{Z}_p^{k'}$ . All these values are kept secret to the simulator.

Let us consider  $\mathcal{V}^* \subseteq \{1, 2, \dots, k'\}$  be the set of all  $i$  for which the challenge identity  $\text{gid}_i^* = 1$ . Let  $\mathcal{V}^* = \{i_1, i_2, \dots, i_{\kappa}\}$ . Now, we choose the  $z_i$  values from  $\mathbf{z}$  corresponding to the collection of indices  $\mathcal{V}^*$ . Sets  $\sum_{i \in \mathcal{V}^*} z_i = \hat{k}s - z'$  for uniformly chosen  $\hat{k} \in [k']$ . Now, we define the function  $\text{K}(\text{gid})$  as

$$\text{K}(\text{gid}) = \begin{cases} 0, & \text{if } z' + \sum_{i \in \mathcal{V}} z_i \equiv 0 \pmod{s} \\ 1, & \text{elsewhere} \end{cases}$$

So, from the above definition of the function  $\text{K}$ , we can say that  $\text{K}(\text{gid}^*) = 0$  and for all  $\text{gid} (\neq \text{gid}^*)$  it becomes non-zero. Additionally, we set another two functions as  $\text{F}(\text{gid}) = p - s\hat{k} + z' + \sum_{i \in \mathcal{V}} z_i$  and  $\text{J}(\text{gid}) = w' + \sum_{i \in \mathcal{V}} w_i$ . The simulator assigns the public parameters  $u' = g_2^{p - \hat{k}s + z'} \cdot g^{w'}$  and  $u_i = g_2^{z_i} \cdot g^{w_i}$ . From the adversarial perspective, the distribution of the public parameter is identical to real construction.

We construct a PPT reduction  $\mathcal{B}$  which breaks the DBDH assumption 4 with non-negligible advantage. The reduction algorithm  $\mathcal{B}$  first receives DBDH challenge instance from the challenger as  $(\mathbb{B}\mathbb{G}, g^a, g^b, g^c, e(g, g)^\tau)$  where  $\mathbb{B}\mathbb{G} = (p, g, \mathbb{G}, \mathbb{G}_T, e(\cdot, \cdot))$  is a group description with  $a, b, c \leftarrow \mathbb{Z}_p$  and the element  $e(g, g)^\tau \in \mathbb{G}_T$  is either  $e(g, g)^{abc}$  or a random

group element from the target group  $\mathbb{G}_T$ . In the following, we discuss about the simulation of the master public key, queried secret keys and the challenge ciphertext. The algorithm  $\mathcal{B}$  works as follows:

**Public key simulation:** The adversary  $\mathcal{B}$  implicitly sets the following vectors of length  $m$  as

$$\mathbf{a} = (a, a_2, \dots, a_m), \mathbf{b} = (b, b, \dots, b), \mathbf{c} = (c, c, \dots, c)$$

where it randomly samples  $a_2, \dots, a_m \leftarrow \mathbb{Z}_p$ . Let us consider the notation  $\mathbf{u} \odot \mathbf{v}$  by component wise multiplication of the vectors  $\mathbf{u}$  and  $\mathbf{v}$ . In this case, the notation  $\mathbf{a} \odot \mathbf{b} = (ab, a_2b, \dots, a_mb) = \mathbf{ba}$ . To generate the public key,  $\mathcal{B}$  implicitly set  $\tilde{\psi} = \mathbf{a}$  and returns the master public key as

$$\text{mpk} = (g^{\mathbf{F}^\top \mathbf{a}}, g_2 = g^b, u', g)$$

where the exponent  $g^{\mathbf{a}}$  is computed as follows:

$$g^{\mathbf{a}} = (g^a, g^{a_2}, \dots, g^{a_m}) = g^{\tilde{\psi}}$$

Note that,  $a_2, \dots, a_m$  are distributed uniformly over  $\mathbb{Z}_p$  and hence the public key components are properly simulated by using the DBDH instance.

**Challenge ciphertext simulation.** To generate the challenge ciphertext,  $\mathcal{B}$  chooses the random exponent  $t \leftarrow \mathbb{Z}_p$  and implicitly sets  $c = t$ . We now show that how does  $\mathcal{B}$  simulate the challenge ciphertext by using the DBDH instance.

$$\begin{aligned} C_1 &= e(g, g)^{\mathbf{F}^\top bc(a, a_2, \dots, a_m)} \cdot e(g, g)^{\mathbf{F}^\top (-b\mathbf{b}e_1 + b\mathbf{F}\mathbf{x}^{(0)})} & C_2 &= g^t = g^c \\ &= e(g, g_2)^{\mathbf{F}^\top (c(a, a_2, \dots, a_m) - \mathbf{b}e_1 + \mathbf{F}\mathbf{x}^{(0)})} & C_3 &= g^{cJ(\text{gid}^*)} = H(\text{gid}^*)^c \\ &= e(g, g_2)^{\mathbf{F}^\top (c(a, a_2, \dots, a_m) - \mathbf{b}e_1 + \mathbf{F}\mathbf{x}^{(0)})} \\ &= e(g, g_2)^{\mathbf{F}^\top (\tilde{\psi}t - \mathbf{b}e_1 + \mathbf{F}\mathbf{x}^{(0)})} \end{aligned}$$

**Secret key simulation.**  $\mathcal{B}$  answers the secret key  $\text{sk}_{\mathbf{y}}$  associated with an identity  $\text{gid}$  and a predicate vector  $\mathbf{y}$  as described below.

We consider two different cases based on queried identity  $\text{gid}$ . For an identity  $\text{gid}$ , consider  $\mathcal{V} \subseteq \{1, \dots, k'\}$  be the set of all  $i$  for which  $\text{gid}_i = 1$ .

**Case 1:** If  $\text{gid} \neq \text{gid}^*$ ,  $\mathcal{B}$  simulates the secret key as follows:

By using the technique of Boneh and Boyen [BB04] and Waters IBE [Wat05] scheme,  $\mathcal{B}$  randomly chooses  $r \leftarrow \mathbb{Z}_p$  then it simulates the secret key  $\text{sk}_{\mathbf{y}} = (d_1, d_2)$  corresponding to an identity  $\text{gid}$  and a vector  $\mathbf{u}$  as

$$\begin{aligned} d_1 &= g^{-\frac{J(\text{gid})}{F(\text{gid})} \cdot \langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle} (u' \prod_{i \in \mathcal{V}} u_i)^r \\ &= g_2^{\langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid})} g^{J(\text{gid})})^{-\frac{\langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle}{F(\text{gid})}} (g_2^{F(\text{gid})} g^{J(\text{gid})})^r \\ &= g_2^{\langle \tilde{\psi}, \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid})} g^{J(\text{gid})})^{r - \frac{\langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle}{F(\text{gid})}} \\ &= g_2^{\langle \tilde{\psi}, \mathbf{F}\mathbf{y} \rangle} H(\text{gid})^{r - \frac{\langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle}{F(\text{gid})}} \\ d_2 &= g^r g^{-\langle (a, a_2, \dots, a_m), \mathbf{F}\mathbf{y} \rangle \frac{1}{F(\text{gid})}} \\ &= g^{r - \langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle \frac{1}{F(\text{gid})}} \\ &= g^{r - \langle \tilde{\psi}, \mathbf{F}\mathbf{y} \rangle \frac{1}{F(\text{gid})}} \end{aligned}$$

We implicitly set  $r' = r - \langle \tilde{\psi}, \mathbf{F}\mathbf{y} \rangle \frac{1}{F(\text{gid})}$ . So, from the construction of K function, we can conclude that  $K(\text{gid}) \neq 0$  for any key query corresponding to the identity  $\text{gid}$ . This implies

that the function  $F(\text{gid}) \neq 0 \pmod N$  for a particular identity (as  $p > sn$  for any reasonable values of  $p, n$  and  $s$ ). To prove the above conclusion, we need the following Lemma 9.

**Case 2:** If  $\text{gid} = \text{gid}^*$ ,  $\mathcal{B}$  responds the secret key  $\text{sk}_{\mathbf{y}}$  as follows:

$$\begin{aligned}
d_1 &= g^{b\mu} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g^{\langle \mathbf{a} \odot \mathbf{b}, \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g^{\langle (a, a_2, \dots, a_m), \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g_2^{\langle (a, a_2, \dots, a_m), \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g_2^{\langle (a, a_2, \dots, a_m), \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g_2^{\langle \mathbf{a}, \mathbf{F}\mathbf{y} \rangle} (g_2^{F(\text{gid}^*)} g^{J(\text{gid}^*)})^r \\
&= g_2^{\langle \tilde{\psi}, \mathbf{F}\mathbf{y} \rangle} H(\text{gid}^*)^r \\
d_2 &= g^r
\end{aligned}$$

where the second equality follows from the fact that  $\langle \mathbf{a} \odot \mathbf{b}, \mathbf{F}\mathbf{y} \rangle = b\mu$  and  $\mu \in \mathbb{Z}_p$  is known to the challenger  $\mathcal{B}$ . From the formation of  $\mathbf{F}$ , we have the relation  $\mathbf{F}(\mathbf{x}^{(0)} - \mathbf{x}^{(1)}) = (1, 0, \dots, 0)^\top = \mathbf{e}_1$ . Since in this case,  $\text{gid} = \text{gid}^*$  so, the queried secret key vector  $\mathbf{y}$  should satisfy the relation  $\langle \mathbf{x}^{(0)} - \mathbf{x}^{(1)}, \mathbf{y} \rangle = 0$ . Therefore, we have  $\langle \mathbf{e}_1, \mathbf{F}\mathbf{y} \rangle = 0$  which implies that  $\langle (\mathbf{a} \odot \mathbf{b}), \mathbf{F}\mathbf{y} \rangle = \langle (ab, a_2b, \dots, a_mb), \mathbf{F}\mathbf{y} \rangle = b\mu$  for any  $\mu \in \mathbb{Z}_p$ .

**Guess.** If  $\mathcal{A}$  guesses the challenge bit  $\mathbf{b} \leftarrow \{0, 1\}$  correctly then  $\mathcal{B}$  returns 1, otherwise, it outputs 0. We consider  $\mathbf{w} = bc(a, a_2, \dots, a_m) = (\tau, bca_2, \dots, bca_m)$  where  $e(g, g)^\tau$  is the challenge element. If  $\tau = abc$ , then all the secret keys and challenge ciphertext are properly distributed. In particular, the challenge ciphertext is an encryption of the message vector  $\mathbf{x}^{(b)}$ . Therefore, in this case,  $\mathcal{A}$  outputs  $\mathbf{b}' = \mathbf{b}$  with advantages  $\frac{1}{2} + \text{negl}(\lambda)$  where  $\frac{1}{2} + \text{negl}(\lambda)$  is the advantage of  $\mathcal{A}$  in the selective security experiment of the IBIPFE. Otherwise, if  $\tau$  is randomly generated from  $\mathbb{Z}_p$  then the challenge ciphertext component  $\mathbf{C}_1$  uniform element from the target group  $\mathbb{G}_T$ . So,  $\mathcal{A}$  can not get any information about the challenge bit  $\mathbf{b}$  from this component. So,  $\mathcal{A}$  wins the experiment with the probability  $\frac{1}{2}$ . Hence, from the hardness of DBDH assumption, it can conclude that  $\mathcal{A}$  has a non-negligible advantages against the proposed IBIPFE scheme which achieves the selective security. This completes the proof.  $\square$

**Lemma 9.** For any  $Q$ -secret key query corresponding to the identities  $\text{gid}^{(1)}, \text{gid}^{(2)}, \dots, \text{gid}^{(Q)}$  to the key generation oracle, the probabilities of  $K(\text{gid}^{(\ell)}) = 1$  with  $z' + \sum_{i \in \mathcal{V}^*} z_i = \hat{k}s$  is non-negligible for all  $\ell$ .

*Proof.* For any set of  $Q$ -queries corresponding to the identities  $\text{gid}^{(1)}, \text{gid}^{(2)}, \dots, \text{gid}^{(Q)}$  and the challenge identity  $\text{gid}^*$ , we compute the following probability.

$$\begin{aligned}
&\Pr \left[ \bigwedge_{\ell=1}^Q (K(\text{gid}^{(\ell)}) = 1) \mid (z' + \sum_{i \in \mathcal{V}^*} z_i = \hat{k}s) \right] \\
&= \Pr \left[ \bigwedge_{\ell=1}^Q (K(\text{gid}^{(\ell)}) = 1 \mid K(\text{gid}^*) = 0) \right] \\
&= \left( 1 - \Pr \left[ \bigvee_{\ell=1}^Q (K(\text{gid}^{(\ell)}) = 0) \mid K(\text{gid}^*) = 0 \right] \right) \\
&\geq \left( 1 - \sum_{\ell=1}^Q \Pr \left[ (K(\text{gid}^{(\ell)}) = 0) \mid K(\text{gid}^*) = 0 \right] \right) = \left( 1 - \frac{Q}{s} \right) = 0.9
\end{aligned}$$



We can optimize the last equation by setting  $s = 10Q$  (as we did in the simulation), where  $Q$  is the maximum number of queries. This above result shows that for all queried identities in the key generation oracle except the challenge identity  $gid$ , the  $K$  values should be equals to 1 with overwhelming probability.  $\square$