IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 2, 27 pages.

https://doi.org/10.62056/andkp2fgx

Check for updates

Decentralized Multi-Client Functional Encryption with Strong Security

Ky Nguyen, David Pointcheval and Robert Schädlich

DIENS, Ecole normale superieure, CNRS, Inria, PSL University, Paris, France

Abstract. Decentralized Multi-Client Functional Encryption (DMCFE) extends the basic functional encryption to multiple clients that do not trust each other. They can independently encrypt the multiple plaintext-inputs to be given for evaluation to the function embedded in the functional decryption key, defined by multiple parameter-inputs. And they keep control on these functions as they all have to contribute to the generation of the functional decryption keys. Tags can be used in the ciphertexts and the keys to specify which inputs can be combined together. As any encryption scheme, DMCFE provides privacy of the plaintexts. But the functions associated to the functional decryption keys might be sensitive too (*e.g.* a model in machine learning). The function-hiding property has thus been introduced to additionally protect the function evaluated during the decryption process.

In this paper, we provide new proof techniques to analyze a new concrete construction of function-hiding DMCFE for inner products, with strong security guarantees in the random oracle model: the adversary can adaptively query multiple challenge ciphertexts and multiple challenge keys, with unbounded repetitions of the same message tags in the ciphertext-queries and a fixed polynomially-large number of repetitions of the same key tags in the key-queries, allowing static corruption of the secret encryption keys. Previous constructions were proven secure in the selective setting only.

Keywords: Functional Encryption · Inner Product · Function-Hiding

1 Introduction

Functional Encryption. Public-Key Encryption (PKE) has become so indispensable that without this building block, secure communication over the Internet would be unfeasible nowadays. However, this concept of PKE limits the access to encrypted data in an *all-or-nothing* fashion: once the recipients have the secret key, they will be able to recover the original data; otherwise, no information is revealed. The concept of Functional Encryption (FE), originally introduced by Boneh, Sahai and Waters [SW05, BSW11], overcomes this limitation: a decryption key can be generated under some specific function F, namely a *functional decryption key*, and enable the evaluation $F(x)$ from an encryption of a plaintext x in order to provide a finer control over the leakage of information about x.

Since its introduction, FE has provided a unified framework for prior advanced encryption notions, such as Identity-Based Encryption [Sha84, Coc01, BF01] or Attribute-Based Encryption [SW05, GPSW06, OSW07, ALdP11, OT12b], and has become a very active domain of research. Abdalla *et al.* [ABDP15] proposed the first FE scheme (in shorthand as ABDP from this point) that allows computing the inner product between a functional vector in the functional decryption key and a data vector in the ciphertext

E-mail: ky.nguyen@ens.fr (Ky Nguyen), david.pointcheval@ens.fr (David Pointcheval), robert.schaedlich@ens.fr (Robert Schädlich)

Received: 2024-01-08 Accepted: 2024-06-04

(IPFE). The interests in FE then increased, either in improving existing constructions for concrete function classes, *e.g.* inner products [ALS16, BBL17, CLT18] and quadratic functions [BCFG17, Gay20, AS17, Lin17], or in pushing the studies of new advanced notions [GVW15] as well as the relationship to other notions in cryptography [AJ15, BV15]. While FE with a single encryptor, *i.e.* single-client FE, is of great theoretical interest, there is also a motivation to investigate a *multi-user* setting, which might be applicable in practical applications when the data is an aggregation of information coming from multiple sources. Another important research question concentrates on the *privacy of functions* under which functional keys are generated. We discuss these two lines of work below.

Extensions of FE in the Multi-User Setting. Goldwasser *et al.* [GGG$^+$14, GKL$^+$13] initiated the study of *Multi-Input Functional Encryption* (MIFE) and *Multi-Client Functional Encryption* (MCFE). In MCFE particularly, the encrypted data is broken into a vector $(x_1, \ldots, x_n)$ and a *client* i among n clients uses their *encryption key* ek_i to encrypt x_i, under some (usually time-based) tag tag. Given a vector of ciphertexts $(\mathsf{ct}_1 \leftarrow \mathsf{Enc}(\mathsf{ek}_1, \mathsf{tag}, x_1), \ldots, \mathsf{ct}_n \leftarrow \mathsf{Enc}(\mathsf{ek}_n, \mathsf{tag}, x_n))$, a decryptor holding a functional decryption key dk_F can decrypt and obtain $F(x_1, \ldots, x_n)$ as long as all $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$ are generated under the same tag. No information beyond $F(x_1, \ldots, x_n)$ is leaked, especially concerning the individual secret components x_i, and combinations of ciphertexts under different message tags provide no further information either. Furthermore, in practice encrypting x_i under different message tags $\mathsf{tag}' \neq \mathsf{tag}$ might bear a different meaning with respect to a client i and thus controls the possibilities constituting ciphertext vectors[1]. This also necessitates the encryption keys ek_i being private. The notion of MCFE can be seen as an extension of FE where multiple clients can contribute into the ciphertext vector independently and non-interactively, where encryption is done by private encryption keys. After their introduction, $\mathsf{MIFE}/\mathsf{MCFE}$ motivated a plethora of works on the subject, notably for the concrete function class of inner products [DOT18, CDG$^+$18a, CDG$^+$18b, ACF$^+$18, ABKW19, ABG19, LT19, CDSG$^+$20, ACGU20, NPP22].

Decentralized Multi-Client Functional Encryption. The setup of MCFE requires some authority (a trusted third party) responsible for the setup and generation of functional decryption keys. The authority possesses a master secret key msk that can be used to handle the distribution of private encryption keys ek_i and deriving functional decryption keys dk_F. When clients do not trust each other, this centralized setting of authority might be a disadvantage. The need for such a central authority is completely eliminated in the so-called *Decentralized Multi-Client Functional Encryption* (DMCFE) introduced by Chotard *et al.* [CDG$^+$18a]. In DMCFE, only during the setup phase do we need interaction for generating parameters that will be needed by the clients later. The key generation is done independently by different *senders*, each has a *secret key* sk_i. Agreeing on a function F, each sender generates their functional key $\mathsf{dk}_{F,i}$ using sk_i, the description of F, and a tag $\mathsf{tag\text{-}f}$. Originally in [CDG$^+$18a], the tag $\mathsf{tag\text{-}f}$ can contain the description of F itself. Using DMCFE, the need of an authority for distributing functional keys is completely removed, with minimal interaction required during setup. The seminal work of [CDG$^+$18a] constructed the first DMCFE for computing inner products ($\mathsf{IP\text{-}DMCFE}$), where n clients can independently contribute to the ciphertext vector $(\mathsf{ct}_1 \leftarrow \mathsf{Enc}(\mathsf{ek}_1, \mathsf{tag}, x_1), \ldots, \mathsf{ct}_n \leftarrow \mathsf{Enc}(\mathsf{ek}_n, \mathsf{tag}, x_n))$ and n senders can independently contribute to the functional keys $\mathsf{dk}_{\mathbf{y},1} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_1, \mathsf{tag\text{-}f}, y_1), \ldots, \mathsf{dk}_{\mathbf{y},n} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_n, \mathsf{tag\text{-}f}, y_n)$ of some vector $\mathbf{y} = (y_1, \ldots, y_n)$. For the function class to compute inner products, many follow-up works improve upon the work of [CDG$^+$18a] on both

[1] In contrast, MIFE involves no message tags and thus a large amount of information can be obtained by arbitrarily combining ciphertexts to decrypt under some functional decryption key.

aspects of efficiency as well as security, or by giving generic transformation to (D)MCFE from single-client FE [LT19, ABKW19, ABG19].

Repetitions under One Tag. Involving tags at the time of encryption and key generation restricts that only ciphertexts and functional keys having the same tag can be combined in the notion of DMCFE. This raises a natural question: what security can we guarantee when one client uses the same tag on multiple data ? We call such multiple usages of the same tag in a DMCFE system *repetitions*. In the formal security model of (D)MCFE in [CDG$^+$18a] and subsequent works [LT19], once the adversary makes a query for (i, tag), further queries for the same pair (i, tag) will be ignored. This means repetitions are not taken into account. The authors of [CDG$^+$18a] argued that it is the responsibility of the users not to use the same tag twice. However, a security notion for DMCFE that captures a sense of protection even when repetitions mistakenly/maliciously happen will be preferable, *e.g.* this is indeed studied in some other works [ABKW19, ABG19]. In addition, when repetitions are allowed for ciphertexts, the security model of MCFE strictly encompasses MIFE by replacing tags with a constant value, as confirmed in recent works [ATY23].

Function Privacy in FE. Standard security notions of FE ensure that adversaries do not learn anything about the content of ciphertexts beyond what is revealed by the functions for which they possess decryption keys. However, it is *not* required that functional decryption keys hide the function they decrypt. In practice, this can pose a serious problem because the function itself could contain confidential data. For example, the evaluated function may represent a neural network. Training such networks is often time-consuming and expensive, which is why companies offer their use as a paid service. However, to ensure that customers continue to pay for the use of the product, it is crucial that the concrete parameters of the network (*i.e.* the computed function) remain secret. This additional security requirement for functional encryption schemes is known as the *function-hiding* property. As another example, suppose one wants to perform statistical analysis (*e.g.* weighted averages) of private data from several companies to get a better understanding of the dynamics of a sector. This can be implemented using a DMCFE for inner products. Consulting firms conduct such analyses as a fee-based service. To ensure that clients continue to pay for updated results in the future, the consulting firm may wish to hide the concrete parameters of their calculations. This can be achieved by using a DMCFE with function-hiding security.

Besides practical applications, function-hiding FE schemes for restricted function classes (such as inner products) have also proven to be an important technical building block for the construction of FE schemes for broader function classes: Lin [Lin17] employed a function-hiding IPFE (FH-IPFE) to obtain an FE scheme for quadratic functions. A different technique was also introduced by Gay in [Gay20] equally aiming at constructing FE for quadratic functions. With several technical novelties, Agrawal *et al.* [AGT21a, AGT22] were able to generalize the aforementioned constructions to obtain MIFE for quadratic functions.

Existing Function-Hiding FE Schemes in the Literature. Bishop *et al.* [BJK15] presented the first IPFE scheme that guaranteed a weak variant of the function-hiding property. This construction was lifted to fully function-hiding security by Datta *et al.* [DDM16]. This was further improved in terms of efficiency and/or computational hardness assumptions by works of [TAO16, KKS19, KLM$^+$18, Tom19, Tom20]. The constructions of [BJK15, DDM16, TAO16] all leverage the power of *dual pairing vector spaces* (DPVSes) developed by Okamoto and Takashima in [OT10, OT12a, OT12b]. Alternatively, Lin [Lin17] used a different approach to get simpler constructions of FH-IPFE from the ABDP IPFE. Using the same blueprint and exploiting the specific algebraic properties of the underlying inner-product MIFE scheme carefully, Abdalla *et al.* [ACF+18] were able to construct function-hiding MIFE for inner products (FH-IP-MIFE). In [AGT21b], Agrawal *et al.* came up with the first construction of function-hiding MCFE for inner products (FH-IP-MCFE) that is inspired by the FH-IP-MIFE by Datta *et al.* [DOT18]. Very recently, Shi and Vanjani [SV23] presented a generic transformation from single-client to multi-client functional encryption, preserving the function-hiding property and leading to the first FH-IP-MCFE with adaptive security. Remarkably, their security proof does not rely on random oracles. We are not aware of any construction of function-hiding DMCFE for inner products (FH-IP-DMCFE) whose security does not rely on the *random oracle model* (ROM).

In [CDSG+20], Chotard *et al.* generalized DMCFE and defined the notion of *Dynamic Decentralized Functional Encryption* (DDFE) that allows users to join at various stages during the lifetime of a system, while maintaining all decentralized features of DMCFE. Notably, the setup of DDFE is non-interactive and decentralized, while that of DMCFE is *a priori* interactive. In the end, a DDFE scheme allows aggregating data from different sources by decrypting an independent list of ciphertexts using an independent list of functional keys, both of which are fabricated in a completely decentralized manner by users with their sk_i, while requiring no trusted third party. To these extents, DDFE is a primitive strictly stronger than DMCFE, given that the function class of the former contains functions that are well-defined relating to a given list of functional keys and those functions can be expressed by the function class of the latter[2]. In [AGT21b], the authors revisits DDFE for the class of inner products (IP-DDFE) and provide a transformation from FH-IP-MCFE to FH-IP-DDFE, following the approach of Chotard *et al.* [CDSG+20] who presented a similar transformation in the non-function-hiding setting. As a consequence, the FH-IP-DDFE scheme of [AGT21b] entails the only FH-IP-DMCFE so far in the literature.

It is worth noting that all known constructions that guarantee function-hiding security rely on pairings. A recent work by Ünal [Üna20] shows that in the manner of most lattice-based approaches, there is little hope to achieve function privacy in IPFE schemes, in the setting of multi-user or not.

Our Contributions

To the best of our knowledge, the only candidate of FH-IP-DMCFE comes from [AGT21b], implicitly as a result of their function-hiding FH-IP-DDFE. The implied security of their FH-IP-DMCFE is *selectively* indistinguishability-based in the ROM under *static* corruption, where the adversary makes all encryption, key generation and corruption queries up front in one shot, with repetitions w.r.t encryption tags and *no repetitions* w.r.t key generation tags. This state-of-the-art leads us to the following question:

> *How far can we raise the security level of pairing-based function-hiding IP-DMCFE in the ROM ?*

In this paper, we strictly improve on various aspects of security compared with [AGT21b]. Below and in Table 1 are presented a summary of our contributions and a comparison with existing works:

1. *Function-Hiding IP-DMCFE.* We construct the first FH-IP-DMCFE that tolerates *adaptive* encryption queries (with unbounded repetitions) and *adaptive* key generation queries *with a fixed polynomially large number repetitions*, under static corruption. The bounded number of repetitions on key generation queries can be polynomially large and is specified at setup time of the scheme. Our FH-DMCFE thus handles up to an *exponentially large* number of mix-and-match of key repetitions under the same tag tag-f, which is determined by the scheme's parameters. It uses pairings

[2] With an appropriate formalization, all function classes in this work, including inner products, satisfy this property.

Table 1: We compare our constructions with existing works, in terms of the type of primitives with function-hiding security (**Type**), whether the encryption oracle ($\mathcal{O}$Enc) and key generation oracle ($\mathcal{O}$KeyGen) can be queried adaptively and with repetitions (**Oracle Queries**), which assumptions are used for the security proof (**Assumptions**), and whether the security is proven in the ROM (✓) or not (✗) (**ROM**). The shorthands (sel, adap) denote selective or adaptive oracle queries. The shorthands (w-rep, bnd-rep, no-rep) indicates whether the adversary can demand repetitive queries to the same slot and tag unboundedly, under a fixed bound, or not, in that order. All schemes are defined for the inner-product functionality of their respective type of primitive (see Def. 6) and consider only static corruption. Preferred properties are underlined.

Scheme	Type	Oracle Queries		Assumptions[††]	ROM
		$\mathcal{O}$Enc	$\mathcal{O}$KeyGen		
[AGT21b, Section 6.2]	FH-IP-MCFE	sel, w-rep	sel[†]	SXDH	✓
[SV23, Section B.3]	FH-IP-MCFE	adap, w-rep	adap[†]	D-Lin	✗[‡]
[AGT21b, Section 6.3]	FH-IP-DMCFE[⋆]	sel, w-rep	sel, no-rep	SXDH	✓
Corollary 1	FH-IP-DMCFE	adap, w-rep	adap, bnd-rep	SXDH	✓

[†] For MCFE, there is no notion of tags for key generation, hence no notion of repetitions.

[‡] This is the only FH-MCFE that is provably secure without the ROM. To our knowledge, there is no FH-DMCFE nor FH-DDFE in the literature that does not use ROs.

[††] All mentioned constructions use pairing groups.

[⋆] This FH-IP-DMCFE is implied by the FH-IP-DDFE of [AGT21b, Section 6.3].

and is provably secure in the ROM. Details about our construction are explained in Section 4.2.

2. *Technical Contribution.* Along the way, we push forward the study of DPVS techniques. We state a novel lemma that shows the indistinguishability of two distributions in a setting where not all input data is known up front. This lemma proves to be the key ingredient for the security proof of our FH-IP-DMCFE scheme in the *adaptive* setting. Due to its oracle-based general formulation, we believe that the lemma can find other applications in the future. The formal statement (Lemma 1) and a proof overview can be found in Section 4.1. Basic definitions for the DPVS framework are provided in Section 3.2.

2 High-Level Overview in the Selective Setting

In this section, we first describe a straightforward construction of a selective FH-DMCFE for inner products based on a blackbox FH-IPFE scheme in the spirit of existing FH-IP-MIFE and FH-IP-MCFE constructions such as [DOT18, AGT21b]. Subsequently, we discuss the main difficulties that need to be overcome towards adaptive security. Regarding the notations of the following overview, we let $\mathbb{Z}_q$ denote the ring of integers with addition and multiplication modulo $q \geq 2$. For a vector $\mathbf{x}$ of dimension n, the notation $\mathbf{x}[i]$ indicates the i-th coordinate of $\mathbf{x}$, for $i \in [n]$. We will follow the implicit notation in [EHK⁺13] and use $[\![a]\!]$ to denote g^a in a cyclic group $\mathbb{G}$ of prime order q generated by g, given $a \in \mathbb{Z}_q$. This implicit notation extends to matrices and vectors having entries in $\mathbb{Z}_q$, *e.g.* $[\![(a, b)]\!] = (g^a, g^b) \in \mathbb{G}^2$.

Recap: The Function-Hiding MCFE of [AGT21b]. An FH-IPFE scheme $\mathsf{iFE} = (\mathsf{iSetup}, \mathsf{iKeyGen}, \mathsf{iEnc}, \mathsf{iDec})$ based on a pairing group $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ enables the sampling of a master secret key $\mathsf{imsk} \leftarrow \mathsf{iSetup}(1^\lambda)$ which can be used to generate functional decryption keys $\mathsf{idk} \leftarrow \mathsf{iKeyGen}(\mathsf{imsk}, [\![\mathbf{y}]\!]_2)$ for vectors $\mathbf{y} \in \mathbb{Z}_q^N$ encoded in $\mathbb{G}_2$ and ciphertexts $\mathsf{ict} \leftarrow \mathsf{iEnc}(\mathsf{imsk}, [\![\mathbf{x}]\!]_1)$ associated with vectors $\mathbf{x} \in \mathbb{Z}_q^N$ encoded in $\mathbb{G}_1$. The decryption $\mathsf{iDec}(\mathsf{idk}, \mathsf{ict})$ reveals only the inner product $[\![\langle \mathbf{x}, \mathbf{y} \rangle]\!]_\mathsf{t}$ of $\mathbf{x}$ and $\mathbf{y}$ encoded in $\mathbb{G}_\mathsf{t}$ and hides all other information about $\mathbf{x}$ and $\mathbf{y}$. When we use several IPFE instances with master secret keys $\mathsf{imsk}_1, \ldots, \mathsf{imsk}_n$ in parallel, we use the shorthands $\mathsf{ict}_i([\![\mathbf{x}]\!]_1)$ and $\mathsf{idk}_i([\![\mathbf{y}]\!]_2)$ for $\mathsf{iEnc}(\mathsf{imsk}_i, [\![\mathbf{x}]\!]_1)$ and $\mathsf{iKeyGen}(\mathsf{imsk}_i, [\![\mathbf{y}]\!]_2)$.

Recall that MCFE is a special case of DMCFE where a trusted authority is responsible for the generation of the functional decryption keys as well as the encryption keys $(\mathsf{ek}_i)_{i \in [n]}$ for the n clients. The key held by the authority is called the master secret key msk. In the scheme of [AGT21b], the encryption key ek_i of a client $i \in [n]$ consists of a master secret key imsk_i of a FH-IPFE scheme. The key-generating authority holds $\mathsf{msk} = (\mathsf{imsk}_i)_{i \in [n]}$. Given a tuple $(i, \mathsf{tag}, \mathbf{x}_i)$, the encryption algorithm defines an extended vector of the form $[\![\widehat{\mathbf{x}}_i]\!]_1 = [\![(\mathbf{x}_i, \omega, \mathbf{0})]\!]_1$, where $\omega = \mathsf{H}(\mathsf{tag})$ is a hash of the tag, and returns $\mathsf{ct}_i = \mathsf{ict}_i([\![\widehat{\mathbf{x}}_i]\!]_1)$. The notation $\mathbf{0}$ in the extended vector $\widehat{\mathbf{x}}_i$ represents additional coordinates that are only used in the security proof and are 0 in the real scheme. A functional decryption key for a vector $\mathbf{y} = (\mathbf{y}_i)_{i \in [n]}$ is created by choosing $t_1, \ldots, t_n \xleftarrow{\$} \mathbb{Z}_q$ conditioned on $\sum_{i \in [n]} t_i = 0$, defining $[\![\widehat{\mathbf{y}}_i]\!]_2 = [\![(\mathbf{y}_i, t_i, \mathbf{0})]\!]_2$ and returning $\mathsf{dk} = \{\mathsf{idk}_i([\![\widehat{\mathbf{y}}_i]\!]_2)\}_{i \in [n]}$. Decrypting $\mathsf{ict}_i([\![\widehat{\mathbf{x}}_i]\!]_1)$ with $\mathsf{idk}_i([\![\widehat{\mathbf{y}}_i]\!]_2)$ gives $\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \omega t_i$ encoded in $\mathbb{G}_\mathsf{t}$. Since the value t_i is secret, the term ωt_i serves as a mask that hides the partial inner product $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. On the other hand, if on has a ciphertext ct_i for each client and all ciphertexts are generated w.r.t the same tag, then the sum of the partial decryptions gives $\sum_{i \in [n]} (\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \omega t_i) = \sum_{i \in [n]} \langle \mathbf{x}_i, \mathbf{y}_i \rangle + \omega \cdot \sum_{i \in [n]} t_i = \sum_{i \in [n]} \langle \mathbf{x}_i, \mathbf{y}_i \rangle$, as $\sum_{i \in [n]} t_i = 0$. The scheme is proven to be secure against selective adversaries that submit *all* oracle queries up front.

Our Selectively Function-Hiding DMCFE. In contrast to MCFE, decryption keys in the DMCFE model are generated non-interactively by n different senders each holding a secret key sk_i for $i \in [n]$. Given a tuple $(\mathsf{tag}\text{-}\mathsf{f}, \mathbf{y}_i)$, sender i produces a partial decryption key dk_i, and decryption is possible if all senders provide their partial key w.r.t the same tag $\mathsf{tag}\text{-}\mathsf{f}$. Our selective FH-DMCFE is a straightforward extension of the FH-MCFE of [AGT21b]. Looking at their scheme, we note that decryption keys already consist of n IPFE keys $\{\mathsf{idk}_i([\![\widehat{\mathbf{y}}_i]\!]_2)\}_{i \in [n]}$. Therefore, it seems natural to let each sender generate one IPFE decryption key. The vectors $\{\widehat{\mathbf{y}}_i\}_{i \in [n]}$ encode a secret sharing $(t_i)_{i \in [n]}$ of 0 which must now be sampled in a decentralized manner. To do so, we fix a secret sharing $(\tilde{t}_i)_i \xleftarrow{\$} \mathbb{Z}_q^n$ of 0 during the (interactive) setup procedure and randomize it by setting $(t_i)_i = (\mu \tilde{t}_i)_i$, where $\mu = \mathsf{H}(\mathsf{tag}\text{-}\mathsf{f})$. Roughly, under the DDH assumption in $\mathbb{G}_2$, such a multiple of $(\tilde{t}_i)_i$ cannot be distinguished from a fresh secret sharing of 0 if the adversary does not obtain several keys for the same sender-tag pair $(i, \mathsf{tag}\text{-}\mathsf{f})$. Using this restriction, it is straightforward to generalize the security proof of [AGT21b] to the case of FH-DMCFE.

Note that both the syntax and security of FH-DMCFE for inner products are symmetric w.r.t key generation and encryption. Therefore, it is mostly irrelevant if the secret sharing $(\tilde{t}_i)_i$ is embedded into the decryption keys or the ciphertexts. For the sake of consistency with our adaptive scheme presented in Section 4.2, we prefer to place it in the ciphertexts. However, we emphasize that the proof of selective security works either way. We summarize our construction in Figure 1.

Proof of Selective Security. As for our adaptive scheme, we only consider static corruptions (see Item 1 of Definition 5). Additionally, we only discuss the proof of one-challenge security against complete queries (see Items 3 and 4). This is sufficient as in

$\mathsf{Setup}(1^\lambda)$: Sample $(\tilde{t}_i)_{i\in[n]} \xleftarrow{\$} \mathbb{Z}_q^n$ such that $\sum_i \tilde{t}_i = 0$; generate n IPFE master secret keys $\{\mathsf{imsk}_i\}_{i\in[n]}$; output $\mathsf{sk}_i = \mathsf{imsk}_i$ and $\mathsf{ek}_i = (\tilde{t}_i, \mathsf{imsk}_i)$ for $i \in [n]$.

$\mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, \mathbf{y}_i)$: Compute $[\![\mu]\!]_2 = \mathsf{H}_2(\mathsf{tag\text{-}f})$; output $\mathsf{dk}_i = \mathsf{idk}_i([\![\widehat{\mathbf{y}}_i]\!]_2)$ for $\widehat{\mathbf{y}}_i = (\mathbf{y}_i, \mu, \mathbf{0})$.

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, \mathbf{x}_i)$: Compute $[\![\omega]\!]_1 = \mathsf{H}_1(\mathsf{tag})$; output $\mathsf{ct}_i = \mathsf{ict}_i([\![\widehat{\mathbf{x}}_i]\!]_1)$ for $\widehat{\mathbf{x}}_i = (\mathbf{x}_i, \omega\tilde{t}_i, \mathbf{0})$.

$\mathsf{Dec}(\{(\mathsf{dk}_i, \mathsf{ct}_i)\}_{i\in[n]})$: Run IPFE decryption for all pairs $(\mathsf{idk}_i([\![\widehat{\mathbf{y}}_i]\!]_2), \mathsf{ict}_i([\![\widehat{\mathbf{x}}_i]\!]_1))$ to recover $[\![z_i]\!]_{\mathsf{t}} = [\![\langle \mathbf{x}_i, \mathbf{y}_i\rangle + \mu\omega\tilde{t}_i]\!]_{\mathsf{t}}$ and find discrete log of $[\![z]\!]_{\mathsf{t}} = [\![\sum_{i\in[n]} z_i]\!]_{\mathsf{t}}$.

Figure 1: Our selectively function-hiding DMCFE scheme

Section 5, we show how to remove both restrictions from the security model via a sequence of generic conversions.

Recall that the one-challenge restriction allows only one tag tag^* to the encryption oracle $\mathcal{O}\mathsf{Enc}(i, \mathsf{tag}^*, \mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})$ having $\mathbf{x}_i^{(0)} \neq \mathbf{x}_i^{(1)}$. Other tags $\mathsf{tag}_\ell \neq \mathsf{tag}^*$ and their corresponding inputs $(\mathbf{x}_{\ell,i}^{(0)}, \mathbf{x}_{\ell,i}^{(1)})$ to $\mathcal{O}\mathsf{Enc}$ are indexed by ℓ and it holds that $\mathbf{x}_{\ell,i}^{(0)} = \mathbf{x}_{\ell,i}^{(1)}$, so we can omit the superscript in this case. Furthermore, we add indices to denote repeated queries to the same client-tag pair. That is, the j-th query to $\mathcal{O}\mathsf{Enc}$ for client i and tag tag^* (respectively tag_ℓ) is denoted by $(\mathbf{x}_i^{(0,j)}, \mathbf{x}_i^{(1,j)})$ (respectively $\mathbf{x}_{\ell,i}^{(j)}$). In the same manner, there exists only one $\mathsf{tag\text{-}f}^*$ queried to the key-generation oracle $\mathcal{O}\mathsf{DKeyGen}(i, \mathsf{tag\text{-}f}^*, \mathbf{y}^{(0)}, \mathbf{y}^{(1)})$ having $\mathbf{y}^{(0)} \neq \mathbf{y}^{(1)}$, while for other $\mathsf{tag\text{-}f}_k \neq \mathsf{tag\text{-}f}^*$ it holds that $\mathbf{y}^{(0)} = \mathbf{y}^{(1)}$. We denote the $\tilde{j}$-th query to $\mathcal{O}\mathsf{DKeyGen}$ for client i and tag $\mathsf{tag\text{-}f}^*$ (respectively $\mathsf{tag\text{-}f}_k$) by $(\mathbf{y}_i^{(0,\tilde{j})}, \mathbf{y}_i^{(1,\tilde{j})})$ (respectively $\mathbf{y}_{k,i}^{(\tilde{j})}$). To summarize, in the one-challenge security game with challenge bit $b \xleftarrow{\$} \{0,1\}$, the adversary obtains the following decryption keys and ciphertexts:

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_i^{(b,\tilde{j})}, \mu, 0, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_i^{(b,j)}, t_i := \omega \cdot \tilde{t}_i, 0, \mathbf{0}, 0, 0]\!]_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i} := \omega_\ell \cdot \tilde{t}_i, 0, \mathbf{0}, 0, 0]\!]_1) \end{aligned} \tag{1}$$

During the entire security proof, we restrict all changes to honest slots $i \in \mathcal{H}$ because the admissibility condition (Item 1 of Definition 5) gives that encryption and key generation queries for corrupted slots $i \in \mathcal{C}$ are already independent of the challenge bit b, so there is nothing to show. In the first step, the simulator randomizes the values t_i and $t_{\ell,i}$ for honest clients $i \in \mathcal{H}$ while relying on the DDH assumption in $\mathbb{G}_1$. Subsequently, the simulator introduces the vectors $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_{\ell,i}$ in the additional 0-coordinates of the ciphertexts of honest clients $i \in \mathcal{H}$.

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_i^{(b,\tilde{j})}, \mu, 0, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_i^{(b,j)}, t_i, 0, \boxed{\mathbf{x}_i^{(1,j)}}, 0, 0]\!]_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, 0, \boxed{\mathbf{x}_{\ell,i}^{(j)}}, 0, 0]\!]_1) \end{aligned} \tag{2}$$

This change cannot be noticed by the adversary assuming message-privacy of iFE. In the next step, the simulator embeds fresh secret sharings $(\tau_i)_{i\in\mathcal{H}}$ and $(\tau_{\ell,i})_{i\in\mathcal{H}}$ of 0 in the ciphertexts for $i \in \mathcal{H}$ as follows:

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_i^{(b,\tilde{j})}, \mu, \boxed{1}, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i}, \mathbf{x}_i^{(1,j)}, 0, 0]\!]_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i([\![\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0]\!]_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i([\![\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0]\!]_1) \end{aligned} \tag{3}$$

This step is not complicated but requires some care. Roughly, we use a sequence of hybrids over the secret sharings $(t_i)_i$ and $(t_{\ell,i})_i$ for all ℓ, where in each hybrid we first hardwire the product $\mu \cdot t_i$ (respectively $\mu \cdot t_{\ell,i}$) in $\mathbf{d}_i^{(j)}$, then rely on the DDH in $\mathbb{G}_2$ to obtain random values t'_i (respectively $t'_{\ell,i}$). These random values can in turn be split into the original product $\mu \cdot t_i$ (respectively $\mu \cdot t_{\ell,i}$) and a fresh random share τ_i (respectively $\tau_{\ell,i}$). To isolate the values of the current hybrid, we use the additional two coordinates at the end of the vectors[3].

The admissibility conditions (Items 1 and 2 of Definition 5) state for all $j_i, \tilde{j}_i$ that

$$\sum_{i\in[n]} \langle \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)} \rangle = \sum_{i\in[n]} \langle \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i^{(1,\tilde{j}_i)} \rangle \quad \text{and} \quad \sum_{i\in[n]} \langle \mathbf{x}_{\ell,i}^{(j_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)} \rangle = \sum_{i\in[n]} \langle \mathbf{x}_{\ell,i}^{(j_i)}, \mathbf{y}_i^{(1,\tilde{j}_i)} \rangle$$

as well as $\mathbf{x}_i^{(0,j)} = \mathbf{x}_i^{(1,j)}$ and $\mathbf{y}_i^{(0,\tilde{j})} = \mathbf{y}_i^{(1,\tilde{j})}$ if $i \in \mathcal{C}$. From this, it follows for $b \in \{0,1\}$[4] that

$$\Delta_i^{(b)} := \langle \mathbf{x}_i^{(b,j)}, \mathbf{y}_i^{(b,\tilde{j})} \rangle - \langle \mathbf{x}_i^{(1,j)}, \mathbf{y}_i^{(1,\tilde{j})} \rangle \qquad \text{and} \qquad \Delta_{\ell,i}^{(b)} := \langle \mathbf{x}_{\ell,i}^{(j)}, \mathbf{y}_i^{(b,\tilde{j})} - \mathbf{y}_i^{(1,\tilde{j})} \rangle$$

are constant for all repetitions $j, \tilde{j}$, and $\Delta_i^{(b)} = \Delta_{\ell,i}^{(b)} = 0$ if $i \in \mathcal{C}$. Furthermore, we have that $\sum_{i\in\mathcal{H}} \Delta_i^{(b)} = \sum_{i\in\mathcal{H}} \Delta_{\ell,i}^{(b)} = 0$. Together, these conditions imply that the distributions

$$D_0 = \left\{ (\tau_i)_{i\in\mathcal{H}} : (\tau_i)_{i\in\mathcal{H}} \xleftarrow{\$} \mathbb{Z}_q^{|\mathcal{H}|} \text{ s.t. } \sum_{i\in\mathcal{H}} \tau_i = 0 \right\}$$

$$D_1 = \left\{ (\tau_i)_{i\in\mathcal{H}} : (\tau'_i)_{i\in\mathcal{H}} \xleftarrow{\$} \mathbb{Z}_q^{|\mathcal{H}|} \text{ s.t. } \sum_{i\in\mathcal{H}} \tau_i = 0, \tau_i := \tau'_i - \Delta_i^{(b)} \right\}$$

are identical (and a similar result also holds for all $(\tau_{\ell,i})_{i\in\mathcal{H}}$). Thus, it is an information-theoretic change to provide the adversary with

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{y}_i^{(b,\tilde{j})}, \mu, 1, \mathbf{0}, 0, 0 \rrbracket_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i - \Delta_i^{(b)}}, \mathbf{x}_i^{(1,j)}, 0, 0 \rrbracket_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0 \rrbracket_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i} - \Delta_{\ell,i}^{(b)}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0 \rrbracket_1) \end{aligned} \tag{4}$$

Relying again on the function privacy of iFE, the simulator can change to:

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \boxed{\mathbf{0}}, \mu, 1, \boxed{\mathbf{y}_i^{(1,\tilde{j})}}, 0, 0 \rrbracket_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i}, \mathbf{x}_i^{(1,j)}, 0, 0 \rrbracket_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0 \rrbracket_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \tau_{\ell,i} - \Delta_{\ell,i}^{(b)}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0 \rrbracket_1) \end{aligned} \tag{5}$$

When applying similar arguments as in the steps from (3) to (5) in a hybrid over $\mathbf{d}_{k,i}^{(\tilde{j})}$ for all k, we finally arrive at:

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{0}, \mu, 0, \mathbf{y}_i^{(1,\tilde{j})}, 0, 0 \rrbracket_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_i^{(b,j)}, t_i, \tau_i, \mathbf{x}_i^{(1,j)}, 0, 0 \rrbracket_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \boxed{\mathbf{0}}, \mu_k, 0, \boxed{\mathbf{y}_{k,i}^{(\tilde{j})}}, 0, 0 \rrbracket_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0 \rrbracket_1) \end{aligned} \tag{6}$$

At this point, we can remove the vectors $\mathbf{x}_i^{(b,j)}$ in $\mathbf{c}_i^{(j)}$ which gives us a game that is independent of the bit b. So the adversary's advantage is 0 and the proof is finished.

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{0}, \mu, 0, \mathbf{y}_i^{(1,\tilde{j})}, 0, 0 \rrbracket_2) & \mathbf{c}_i^{(j)} &= \mathsf{ict}_i(\llbracket \boxed{\mathbf{0}}, t_i, \tau_i, \mathbf{x}_i^{(1,j)}, 0, 0 \rrbracket_1) \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= \mathsf{idk}_i(\llbracket \mathbf{0}, \mu_k, 0, \mathbf{y}_{k,i}^{(\tilde{j})}, 0, 0 \rrbracket_2) & \mathbf{c}_{\ell,i}^{(j)} &= \mathsf{ict}_i(\llbracket \mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \tau_{\ell,i}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0 \rrbracket_1) \end{aligned} \tag{7}$$

[3] Later in the proof for our $\mathsf{FH\text{-}DMCFE}$ based on DPVS, introducing the new random shares $\tau_i, \tau_{\ell,i}$ is taken care by Lemma 1, using particularly the DPVS basis changes and DDH. We thus do not write the random share introduction explicitly in the $\mathsf{FH\text{-}DMCFE}$ proof and refer to the transitions $\mathsf{G}_0 \to \mathsf{G}_1$ in the proof of Lemma 1 for more details.

[4] More precisely, the case $b = 0$ follows from the admissibility condition while for $b = 1$, we always have $\Delta_i^{(b)} = \Delta_{\ell,i}^{(b)} = 0$

Problems for Adaptive Security. In (4), the simulator embeds $\Delta_i^{(b)}$ and $\Delta_{\ell,i}^{(b)}$ into the ciphertexts. Note that these values do not only depend on the respective encryption query but also on key generation queries. In the selective setting where all queries are submitted up front, this does not pose a problem. In the adaptive setting, however, this can lead to the situation that the challenger needs to embed values into ciphertexts before they were even input to an oracle query.

To overcome this problem, we provide a concrete instantiation of the underlying $\mathsf{FH\text{-}IPFE}$ scheme based on DPVSes. If the simulator gets into a situation where it would have to use inputs that have not yet been queried by the adversary, we make it guess them. Even though this guess degrades the probability of a successful efficient simulation by an exponential factor, it does not help the adversary because we design the games to have perfectly identical views, thanks to *information-theoretic* properties of the DPVS setting. Jumping ahead, dealing with repeated queries for the same tag will be more tricky in this context. We therefore describe the main technical lemma in Section 4.1. It is worth noting what is involved in the lemma and how it is used in the proof. The statement of Lemma 1 considers the indistinguishability of an adversary's views corresponding to their interactions with multiple oracles, *before* and *after* swapping the indexed contents of some oracle's outputs and changing the contents' indices from $b = 1$ to $b = 0$. The aforementioned oracles in Lemma 1 correspond to the execution of key-generation and ciphertext oracles of the $\mathsf{FH\text{-}DMCFE}$ security experiment (see Figure 2), for challenge and non-challenge queries. Lemma 1 allows the adversary to adaptively query the oracles to model the situation in the $\mathsf{FH\text{-}DMCFE}$ security proof, where key-generation and ciphertext oracles can be queried adaptively. Later on, the oracles in Lemma 1 are relevant whenever the lemma is applied in the $\mathsf{FH\text{-}DMCFE}$ security proof. we verify the hypothesis of the lemma and list the $\mathsf{FH\text{-}DMCFE}$ security's oracles outputs in the order of the lemma's oracles to affect the correct vectors. A discussion of our adaptively secure $\mathsf{FH\text{-}DMCFE}$ is given in Section 4.2.

3 Preliminaries

For integers m and n with $m < n$, we write $[m; n]$ to denote the set $\{z \in \mathbb{Z} : m \leq z \leq n\}$ and set $[n] \coloneqq [1; n]$. For a finite set $\mathcal{S}$, we let $2^{\mathcal{S}}$ denote the power set of $\mathcal{S}$, and $U(\mathcal{S})$ denote the uniform distribution over S. For any $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers with addition and multiplication modulo q. Given a prime q and an integer N, we denote by $GL_N(\mathbb{Z}_q)$ the general linear group of degree N over $\mathbb{Z}_q$, and use non-boldface capital letters $B, H, \ldots$ for scalar matrices in $GL_N(\mathbb{Z}_q)$. We write vectors as row-vectors, unless stated otherwise. For a vector $\mathbf{x}$ of dimension n, the notation $\mathbf{x}[i]$ indicates the i-th coordinate of $\mathbf{x}$, for $i \in [n]$. We will follow the implicit notation in [EHK+13] and use $[\![a]\!]$ to denote g^a in a cyclic group $\mathbb{G}$ of prime order q generated by g, given $a \in \mathbb{Z}_q$. This implicit notation extends to matrices and vectors having entries in $\mathbb{Z}_q$, *e.g.* $[\![(a, b)]\!] = (g^a, g^b) \in \mathbb{G}^2$. We use boldface letters $\mathbf{B}, \mathbf{b}, \ldots$ for matrices and vectors of group elements, unless stated otherwise. We use the shorthand ppt for "probabilistic polynomial time". In the security proofs, whenever we use an ordered sequence of games $(\mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_i, \ldots, \mathsf{G}_L)$ indexed by $i \in [0; L]$, we refer to the predecessor of G_j by G_{j-1}, for $j \in [L]$.

3.1 Hardness Assumptions

We state the assumptions needed for our constructions.

Definition 1 (Decisional Diffie-Hellman)**.** In a cyclic group $\mathbb{G}$ of prime order q, the *Decisional Diffie-Hellman* (DDH) problem is to distinguish the distributions

$$D_0 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![ab]\!])\} \qquad D_1 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![c]\!])\}.$$

for $a, b, c \xleftarrow{\$} \mathbb{Z}_q$. The DDH assumption in $\mathbb{G}$ assumes that no ppt adversary can solve the DDH problem with non-negligible probability.

Definition 2 (Decisional Separation Diffie-Hellman)**.** In a cyclic group $\mathbb{G}$ of prime order q, the *Decisional Separation Diffie-Hellman* (DSDH) problem is to distinguish the distributions

$$D_0 = \{(x, y, [\![1]\!], [\![a]\!], [\![b]\!], [\![ab + x]\!])\} \qquad D_1 = \{(x, y, [\![1]\!], [\![a]\!], [\![b]\!], [\![ab + y]\!])\}$$

for any $x, y \in \mathbb{Z}_q$, and $a, b \xleftarrow{\$} \mathbb{Z}_q$. The DSDH assumption in $\mathbb{G}$ assumes that no ppt adversary can solve the DSDH problem with non-negligible probability.

It can be shown straightforwardly that $\mathbf{Adv}_{\mathbb{G}}^{\mathsf{DSDH}}(1^\lambda) \leq 2 \cdot \mathbf{Adv}_{\mathbb{G}}^{\mathsf{DDH}}(1^\lambda)$.

Definition 3 (Symmetric External Diffie-Hellman)**.** In the bilinear setting $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$, the *Symmetric eXternal Diffie-Hellman* (SXDH) assumption makes the DDH assumption in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

3.2 Dual Pairing Vector Spaces

We need the *Decisional Diffie-Hellman* (DDH) assumption in a cyclic group $\mathbb{G}$ of prime order q, which assumes no ppt adversary can distinguish $\{([\![1]\!], [\![a]\!], [\![b]\!], [\![ab]\!])\}$ from $\{([\![1]\!], [\![a]\!], [\![b]\!], [\![c]\!])\}$ with non-negligible probablity, where the probability is taken over the choices $a, b, c \xleftarrow{\$} \mathbb{Z}_q$ and the adversary's coins. In the bilinear setting $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$, the *Symmetric eXternal Diffie-Hellman* (SXDH) assumption makes the DDH assumption in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Formal definitions are given in the full version [NPS24]. Our constructions rely on the *Dual Pairing Vector Spaces* (DPVS) framework in the prime-order bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$, and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are all written additively. The DPVS technique dates back to the seminal work by Okamoto-Takashima [OT10, OT12a, OT12b] aiming at adaptive security for ABE as well as IBE, together with the *dual system methodology* introduced by Waters [Wat09]. In [LW10], the setting for dual systems is composite-order bilinear groups. Continuing on this line of works, Chen *et al.* [CLL+13] used prime-order bilinear groups under the SXDH assumption.

Formalization. Let us fix $N \in \mathbb{N}$ and consider $\mathbb{G}_1^N$ having N copies of $\mathbb{G}_1$. Viewing $\mathbb{Z}_q^N$ as a vector space of dimension N over $\mathbb{Z}_q$ with the notions of bases, we can obtain naturally a similar notion of bases for $\mathbb{G}_1^N$. More specifically, any invertible matrix $B \in GL_N(\mathbb{Z}_q)$ identifies a basis $\mathbf{B}$ of $\mathbb{G}_1^N$, whose i-th row $\mathbf{b}_i$ is $[\![B_i]\!]_1$, where B_i is the i-th row of B. It is straightforward that we can write $\mathbf{B} = [\![B]\!]_1$ for any basis $\mathbf{B}$ of $\mathbb{G}_1^N$ corresponding to an invertible matrix $B \in GL_N(\mathbb{Z}_q)$. We write $\mathbf{x} = (m_1, \ldots, m_N)_\mathbf{B}$ to indicate the representation of $\mathbf{x}$ in the basis $\mathbf{B}$, i.e. $\mathbf{x} = \sum_{i=1}^N m_i \cdot \mathbf{b}_i$. At some point when we focus on the indices in an ordered list L of length ℓ, we write $\mathbf{x} = (m_{L[1]}, \ldots, m_{L[\ell]})_{\mathbf{B}[L]}$. Treating $\mathbb{G}_2^N$ similarly, we can furthermore define a product of two vectors $\mathbf{x} = [\![(m_1, \ldots, m_N)]\!]_1 \in \mathbb{G}_1^N, \mathbf{y} = [\![(k_1, \ldots, k_N)]\!]_2 \in \mathbb{G}_2^N$ by $\mathbf{x} \times \mathbf{y} := \prod_{i=1}^N \mathbf{e}(\mathbf{x}[i], \mathbf{y}[i]) = [\![\langle (m_1, \ldots, m_N), (k_1, \ldots, k_N) \rangle]\!]_\mathsf{t}$. Given a basis $\mathbf{B} = (\mathbf{b}_i)_{i \in [N]}$ of $\mathbb{G}_1^N$, we define $\mathbf{B}^*$ to be a basis of $\mathbb{G}_2^N$ by first defining $B^* := (B^{-1})^\top$ and the i-th row $\mathbf{b}_i^*$ of $\mathbf{B}^*$ is $[\![B_i^*]\!]_2$. It holds that $B \cdot (B^*)^\top = I_N$ the identity matrix and $\mathbf{b}_i \times \mathbf{b}_j^* = [\![\delta_{i,j}]\!]_\mathsf{t}$ for every $i, j \in [N]$, where $\delta_{i,j} = 1$ if and only if $i = j$. We call the pair $(\mathbf{B}, \mathbf{B}^*)$ a *pair of dual orthogonal bases* of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$. If $\mathbf{B}$ is constructed by a random invertible matrix $B \xleftarrow{\$} GL_N(\mathbb{Z}_q)$, we call the resulting $(\mathbf{B}, \mathbf{B}^*)$ a pair of random dual bases. A DPVS is a bilinear group setting $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ with dual orthogonal bases. We denote by $\mathsf{DPVSGen}$ the algorithm that takes as inputs $\mathbb{G}$, a unary 1^N, and some random coins $r \in \{0, 1\}^*$, then outputs a pair of random matrices (B, B^*) that specify dual bases $(\mathbf{B} = [\![B]\!]_1, \mathbf{B}^* = [\![B^*]\!]_2)$ of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$. Further details on DPVS-related techniques can be found in the full version [NPS24].

3.3 Decentralized Multi-Client Functional Encryption

The notion of *Decentralized Multi-Client Functional Encryption* (DMCFE) is introduced in [CDG+18a] where (1) the number of users is fixed in advanced by a (possibly interactive) global setup and (2) the key of a user can be an *encryption key* to encrypt their private individual data (a "client" in the terminology of [CDG+18a]) or a *secret key* to generate a functional key component (a "sender" in the terminology of [CDG+18a]). Moreover, for efficiency, prior papers (such as [CDG+18a, CDG+18b, ABKW19, ABG19, LT19, CDSG+20]) considered an additional *key combination* algorithm that, given n functional key components $(\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i\in[n]}$ generated for the same tag $\mathsf{tag\text{-}f}$, outputs a succinct functional key $\mathsf{dk}_{\mathsf{tag\text{-}f}}$ which can be passed to decryption $\mathsf{Dec}(\mathsf{dk}_{\mathsf{tag\text{-}f}}, \mathbf{c})$. Without loss of generality, the DMCFE notion in this paper implicitly includes the key combination algorithm in the decryption algorithm and whenever we refer to other existing DMCFE schemes, they are syntactically understood as such. The formal definition of DMCFE that is used in this paper is given below.

Let $\{\mathsf{Tag}_\lambda\}_{\lambda\in\mathbb{N}}$, $\{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, $\{\mathcal{R}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{Param}_\lambda\}_{\lambda\in\mathbb{N}}$ be sequences of tag, domain, range and parameter spaces, respectively. We consider a function class $\mathcal{F} = \{\mathcal{F}_{n,\lambda}\}_{n,\lambda\in\mathbb{N}}$, where each $\mathcal{F}_{n,\lambda} = \{f_{n,\lambda,(y_1,\ldots,y_n)}\}_{(y_1,\ldots,y_n)}$ contains functions $f_{n,\lambda,(y_1,\ldots,y_n)} \colon \mathcal{D}_\lambda^n \to \mathcal{R}_\lambda$ described by their parameters $(y_1,\ldots,y_n) \in \mathsf{Param}_\lambda^n$.[5]

Definition 4 (Decentralized Multi-Client Functional Encryption)**.** A *DMCFE scheme* $\mathcal{E}$ for $\mathcal{F}$ between n senders $(\mathcal{S}_i)_{i\in[n]}$ and a functional decrypter $\mathcal{FD}$ consists of the four algorithms defined below:

$\mathsf{Setup}(1^\lambda, 1^n)$**:** This is a protocol between the senders that eventually generate their own secret keys sk_i and encryption keys ek_i, as well as some optional public parameters pp. We will assume that all the secret and encryption keys implicitly contain pp.

$\mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, y_i)$**:** On input a secret key sk_i, a tag $\mathsf{tag\text{-}f} \in \mathsf{Tag}$, and parameter $y_i \in \mathsf{Param}_\lambda$, this algorithm outputs a partial decryption key $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$.

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i)$**:** On input an encryption key ek_i, a tag tag and a message $x_i \in \mathcal{D}_\lambda$, this algorithm outputs a ciphertext $\mathsf{ct}_{\mathsf{tag},i}$.

$\mathsf{Dec}(\mathbf{d}, \mathbf{c})$**:** On input a list of functional decryption keys $\mathbf{d} \coloneqq (\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ and a list of ciphertexts $\mathbf{c} \coloneqq (\mathsf{ct}_{\mathsf{tag},i})_{i=1}^n$, this algorithm runs a key combination if necessary, then outputs an element $d \in \mathcal{R}_\lambda$ or a symbol $\perp$.

Correctness. $\mathcal{E}$ is *correct* if for all $\lambda, n \in \mathbb{N}$, $(x_1,\ldots,x_n) \in \mathcal{D}_\lambda^n$, $f_{n,\lambda,(y_1,\ldots,y_n)} \in \mathcal{F}_{n,\lambda}$ having parameters $(y_1,\ldots,y_n) \in \mathsf{Param}_\lambda^n$, and for any $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}_\lambda$, we have

$$\Pr\left[d = f_{n,\lambda,(y_1,\ldots,y_n)}(x_1,\ldots,x_n) \;\middle|\; \begin{array}{l} (\mathsf{pp}, (\mathsf{sk}_i)_{i\in[n]}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n) \\ \forall i \in [n] \colon \mathsf{dk}_{\mathsf{tag\text{-}f},i} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, y_i) \\ \forall i \in [n] \colon \mathsf{ct}_{\mathsf{tag},i} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i) \\ d \coloneqq \mathsf{Dec}((\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i\in[n]}, (\mathsf{ct}_{\mathsf{tag},i})_{i\in[n]}) \end{array} \right] = 1$$

where the probability is taken over the random coins of the algorithms.

Security. We define function-hiding and standard security for DMCFE. In the seminal work by Chotard *et al.* [CDG+18a] and its follow-up study [CDSG+20], the security notion does not cover the function-hiding requirement for DMCFE or its more general sibling DDFE. Until recently, the work by Agrawal *et al.* [AGT21b] abstracted out DMCFE into

[5] Implicitly, we use a deterministic encoding $\mathsf{p}_\lambda : \mathcal{F}_\lambda \to \mathsf{Param}_\lambda \times \cdots \times \mathsf{Param}_\lambda$ in order to associate each function to its parameters.

the notion of *Multi-Party Functional Encryption* (MPFE). The authors of [AGT21b] also used MPFE to spell out the function-hiding security for MCFE as well as for DDFE. The latter does capture DMCFE as a particular case but for convenience of the reader, we introduce the detailed function-hiding security for DMCFE, without going through all the abstraction of MPFE nor of DDFE. Our security definition follows the *Game-Playing Framework* in [BR06]: Figure 2 defines the experiment $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ with procedures $\mathsf{Initialize}$, $\mathcal{O}\mathsf{DKeyGen}$, $\mathcal{O}\mathsf{Enc}$, $\mathcal{O}\mathsf{Corrupt}$ and $\mathsf{Finalize}$; the adversary $\mathcal{A}$ runs $\mathsf{Initialize}$, can call the oracles in any order and any number of times, and finishes the run by calling $\mathsf{Finalize}$ on input the guess b'.

Definition 5 (Function-Hiding Security)**.** Let $\lambda \in \mathbb{N}$ be a security parameter. For a DMCFE scheme $\mathcal{E}$, a function class $\mathcal{F} = \{\mathcal{F}_{n,\lambda}\}_{n,\lambda}$ and a ppt adversary $\mathcal{A}$ we define the experiment $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ as shown in Figure 2 and set $\mathcal{H} := [n] \setminus \mathcal{C}$. The oracles $\mathcal{O}\mathsf{Enc}$, $\mathcal{O}\mathsf{DKeyGen}$ and $\mathcal{O}\mathsf{Corrupt}$ can be called in any order and any number of times. The adversary $\mathcal{A}$ is *NOT admissible* with respect to $\mathcal{C}$, $\mathcal{Q}_{\mathsf{Enc}}$, $\mathcal{Q}_{\mathsf{KGen}}$, denoted by $\mathsf{adm}(\mathcal{A}) = 0$, if either one of the following holds:

1. There exists a tuple $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ or $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ such that $i \in \mathcal{C}$ and $x_i^{(0)} \neq x_i^{(1)}$[6] or $y_i^{(0)} \neq y_i^{(1)}$.

2. There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, two vectors $(x_i^{(0)})_{i\in[n]}, (x_i^{(1)})_{i\in[n]} \in \mathcal{D}_1 \times \cdots \times \mathcal{D}_n$ and functions $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}, f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})} \in \mathcal{F}$ having parameters $(y_i^{(0)}, y_i^{(1)})_{i\in[n]}$ such that

   - $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ and $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ for all $i \in \mathcal{H}$,
   - $x_i^{(0)} = x_i^{(1)}$ and $y_i^{(0)} = y_i^{(1)}$ for all $i \in \mathcal{C}$, and
   - $f^{(0)}_{n,\lambda,y_1^{(0)},\ldots,y_n^{(0)}}(x_1^{(0)},\ldots,x_n^{(0)}) \neq f^{(1)}_{n,\lambda,y_1^{(1)},\ldots,y_n^{(1)}}(x_1^{(1)},\ldots,x_n^{(1)})$.

Otherwise, we say that $\mathcal{A}$ is *admissible* w.r.t $\mathcal{C}$, $\mathcal{Q}_{\mathsf{Enc}}$ and $\mathcal{Q}_{\mathsf{KGen}}$ and write $\mathsf{adm}(\mathcal{A}) = 1$. We call $\mathcal{E}$ function-hiding if for all ppt adversaries $\mathcal{A}$,

$$\mathbf{Adv}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left| \Pr\left[\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1\right] - \frac{1}{2} \right|$$

is negligible in λ.

Weaker Notions. We define weaker variants of indistinguishability by restricting the access to the oracles and imposing stronger admissibility conditions. In this paper we first present our main technical scheme under some weaker notions, then our final scheme under stronger notions is obtained following some general lemmas (see Section 5).

1. *Security against Static Corruption:* The experiment $\mathbf{Exp}^{\mathsf{stat\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that all queries to the oracle $\mathcal{O}\mathsf{Corrupt}$ must be submitted before $\mathsf{Initialize}$ is called.

2. *Security against Selective Challenges:* The experiment $\mathbf{Exp}^{\mathsf{sel\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that all queries to the oracles $\mathcal{O}\mathsf{KeyGen}$ and $\mathcal{O}\mathsf{Enc}$ must be submitted before $\mathsf{Initialize}$ is called.

[6] This admissibility condition on $x_i^{(0)} = x_i^{(1)}$ for all $i \in \mathcal{C}$ was introduced in [CDG+18a] then used in all other works on $\mathsf{(D)MCFE}$ [CDG+18a, LT19, ABKW19, ABG19] and later on DDFE [CDSG+20, AGT21b]. A recent work [NPP23] studies the relaxation that removes this condition for $\mathsf{(D)MCFE}$, *i.e.* allowing $x_i^{(0)} \neq x_i^{(1)}$ for $i \in \mathcal{C}$ and more attacks are considered admissible, and gives a provably secure DMCFE candidate computing inner products. We are not aware of any DMCFE scheme in the literature which is proven secure under the stronger notion from [NPP23].

```
$\mathsf{Initialize}(1^\lambda, 1^n)$:
    $\mathcal{C}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{Q}_{\mathsf{KGen}} \leftarrow \varnothing$; $b \xleftarrow{\$} \{0,1\}$
    $(\mathsf{pp}, (\mathsf{sk}_i)_{i\in[n]}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$
    Return $\mathsf{pp}$

$\mathcal{O}\mathsf{DKeyGen}(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})$:
    $\mathcal{Q}_{\mathsf{KGen}} \leftarrow \mathcal{Q}_{\mathsf{KGen}} \cup \{(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})\}$
    Return $\mathsf{dk}_{f,i} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, y_i^{(b)})$

$\mathcal{O}\mathsf{Enc}(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})$:
    $\mathcal{Q}_{\mathsf{Enc}} \leftarrow \mathcal{Q}_{\mathsf{Enc}} \cup \{(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})\}$
    Return $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i^{(b)})$

$\mathcal{O}\mathsf{Corrupt}(i)$:
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{i\}$; return $(\mathsf{sk}_i, \mathsf{ek}_i)$

$\mathsf{Finalize}(b')$:
    If $\mathsf{adm}(\mathcal{A}) = 1$, return $\beta \leftarrow (b' \overset{?}{=} b)$
    Else, return 0
```

Figure 2: Security game $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ for Definition 5

3. *One-time Security:* The experiment $\mathbf{Exp}^{\mathsf{1chal\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that the adversary must declare up front to $\mathsf{Initialize}$ two additional "challenge" tags $\mathsf{tag}^*, \mathsf{tag\text{-}f}^* \in \mathsf{Tag}$ such that for all $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$:

   - if $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ and $\mathsf{tag} \neq \mathsf{tag}^*$, then $x_i^{(0)} = x_i^{(1)}$,
   - if $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ and $\mathsf{tag\text{-}f} \neq \mathsf{tag\text{-}f}^*$, then $y_i^{(0)} = y_i^{(1)}$.

4. *Security against Complete Challenges:* The experiment $\mathbf{Exp}^{\mathsf{pos\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that we add the following condition 3 for $\mathsf{adm}(\mathcal{A}) = 0$ that we call the *complete-query constraint*:

   3. There exists $\mathsf{tag} \in \mathsf{Tag}$ so that a query $\mathcal{O}\mathsf{Enc}(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})$ has been asked for some but not all $i \in \mathcal{H}$, or there exists $\mathsf{tag\text{-}f} \in \mathsf{Tag}$ such that a query $\mathcal{O}\mathsf{KeyGen}(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})$ has been asked for some but not all $i \in \mathcal{H}$.

   In other words, we require for an adversary $\mathcal{A}$ to be *admissible* that, for any tag, either $\mathcal{A}$ makes no encryption (resp. key) query or makes at least one encryption (resp. key) query for each slot $i \in \mathcal{H}$.

5. *Weak Function-Hiding:* We can weaken the function-hiding property by changing condition 2 for $\mathsf{adm}(\mathcal{A}) = 0$. More specifically, we replace it by the following condition 2':

   2'. There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, $(x_i^{(0)})_{i\in[n]}$ *and* $(x_i^{(1)})_{i\in[n]}$ *in* $\mathcal{D}_1 \times \cdots \times \mathcal{D}_n$ *and two functions* $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}, f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})} \in \mathcal{F}$ *having parameters* $(y_i^{(0)}, y_i^{(1)})_{i=1}^n$ *such that*

   - $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ *and* $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ *for all* $i \in \mathcal{H}$*,*
   - $x_i^{(0)} = x_i^{(1)}$ *and* $y_i^{(0)} = y_i^{(1)}$ *for all* $i \in \mathcal{C}$*, and*
   - $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}(x_1^{(0)}, \ldots, x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(1)}, \ldots, x_n^{(1)})$ OR $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}(x_1^{(0)}, \ldots, x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(0)}, \ldots, x_n^{(0)})$ OR $f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(0)}, \ldots, x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(1)}, \ldots, x_n^{(1)})$.

   The experiment in this weak function-hiding model is denoted by $\mathbf{Exp}^{\mathsf{wfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$.

In this paper we focus on the concrete class of inner products. The function family $\mathcal{F}^{\mathsf{ip}}_n$ of bounded-norm inner-product functionalities with n inputs is defined as follows.

Definition 6 (Inner Product Functionality)**.** For $n, \lambda \in \mathbb{N}$, let $\mathcal{D}_\lambda = \mathsf{Param}_\lambda = [-B; B]^N$ and $\mathcal{R}_\lambda = [-nNB^2; nNB^2]$, where $B = B(\lambda)$ and $N = N(\lambda) \colon \mathbb{N} \to \mathbb{N}$ are polynomials. We define the *inner-product functionality* $\mathcal{F}^{\mathsf{ip}} = \{\mathcal{F}^{\mathsf{ip}}_{n,\lambda}\}_{n,\lambda\in\mathbb{N}}$ for $\mathcal{F}^{\mathsf{ip}}_{n,\lambda} = \{f_{n,\lambda,(\mathbf{y}_1,\ldots,\mathbf{y}_n)} \colon \mathcal{D}^n_\lambda \to \mathcal{R}_\lambda\}_{(\mathbf{y}_1,\ldots,\mathbf{y}_n)\in\mathsf{Param}^n_\lambda}$ as the family of functions

$$f_{n,\lambda,(\mathbf{y}_1,\ldots,\mathbf{y}_n)}(\mathbf{x}_1,\ldots,\mathbf{x}_n) = \textstyle\sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i\rangle \enspace .$$

4 A $\mathsf{FH\text{-}DMCFE}$ for Inner Products

4.1 Swapping Lemma

In this section we state a technical lemma that will be the basis of the security analysis of our function-hiding IP-DMCFE. This lemma plays an important role in the proof of Theorem 1 and is revisited in Section 4.2. As a reminder, we refer to the paragraph **Problems for Adaptive Security** in the technical overview of Section 2 for a discussion on why the oracles in the following statement of Lemma 1 are relevant afterwards in the FH-DMCFE proof.

Lemma 1 (Swapping)**.** *Let $\lambda \in \mathbb{N}$ and $H = H(\lambda), K = K(\lambda), L = L(\lambda), J_i = J_i(\lambda), \widetilde{J}_i = \widetilde{J}_i(\lambda), N = N(\lambda) \in \mathbb{N}$ where $i \in [H]$ and $H, K, L, J_i, \widetilde{J}_i, N : \mathbb{N} \to \mathbb{N}$ are polynomials. Let $\tilde{J} := \max_{i\in[H]}\{\tilde{J}_i\}$, where the maximum is over polynomial evaluations $\tilde{J}_i(\lambda) \in \mathbb{N}$. Let $(\mathbf{B}_i, \mathbf{B}^*_i)$, for each $i \in [H]$, be a pair of random dual bases of dimension $2N + 2N \cdot \tilde{J} + 4$ in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_{\mathrm{t}}, g_1, g_2, g_{\mathrm{t}}, \mathbf{e}, q)$. All basis vectors are kept secret. Let $R, R_1, \ldots, R_K \in \mathbb{Z}_q$ be some public scalars. For $i \in [H]$, $\ell \in [L]$ and $k \in [K]$, sample $\sigma_i, \sigma_{i,k}, r, r_\ell \xleftarrow{\$} \mathbb{Z}_q$ conditioned on $\sum_{i\in[H]} \sigma_i = R$ and $\sum_{i\in[H]} \sigma_{k,i} = R_k$.*

We consider the following oracles:

$\underline{\tilde{\mathcal{O}}_{\mathbf{d}}\mathbf{:}}$ *On input $(\ell, i, \mathbf{y}^{(\mathsf{rep})}_{\ell,i}, \mathbf{y}^{(\mathsf{rep})'}_{\ell,i}) \in [L] \times [H] \times \mathbb{Z}^N_q \times \mathbb{Z}^N_q$, where $\mathsf{rep} \in [J_i]$ is a counter for the number of queries of the form $(\ell, i, \star, \star)$, sample $\rho^{(\mathsf{rep})}_{\ell,i} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{d}^{(\mathsf{rep})}_{\ell,i} = (\mathbf{y}^{(\mathsf{rep})}_{\ell,i},\ \mathbf{y}^{(\mathsf{rep})'}_{\ell,i},\ r_\ell,\ 0,\ \rho^{(\mathsf{rep})}_{\ell,i},\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i} \enspace .$$

$\underline{\boxed{\mathcal{O}^b_{\mathbf{d}}}\mathbf{:}}$ *For $b \in \{0,1\}$, on input $(i, \mathbf{y}^{(1,\tilde{j}_i)}_i, \mathbf{y}^{(0,\tilde{j}_i)}_i) \in [H] \times \mathbb{Z}^N_q$, where $\tilde{j}_i \in [\tilde{J}_i]$ is a counter for the number of queries of the form $(i, \star, \star)$, sample $\rho^{(\tilde{j}_i)}_i \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\begin{aligned} &\textit{If } \boxed{b=0}: \quad \mathbf{d}^{(\tilde{j}_i)}_i = (\boxed{\mathbf{y}^{(1,\tilde{j}_i)}_i},\ \boxed{0^N},\ r,\ 0,\ \rho^{(\tilde{j}_i)}_i,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i} \\ &\textit{If } \boxed{b=1}: \quad \mathbf{d}^{(\tilde{j}_i)}_i = (\boxed{0^N},\ \boxed{\mathbf{y}^{(0,\tilde{j}_i)}_i},\ r,\ 0,\ \rho^{(\tilde{j}_i)}_i,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i} \enspace . \end{aligned}$$

$\underline{\mathcal{O}_{\mathbf{c}}\mathbf{:}}$ *On input $(i, \mathbf{x}^{(1,j_i)}_i, \mathbf{x}^{(0,j_i)}_i) \in [H] \times \mathbb{Z}^N_q \times \mathbb{Z}^N_q$, where $j_i \in [J_i]$ is a counter for the number of queries of the form $(i, \star, \star)$, sample $\pi^{(j_i)}_i \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{c}^{(j_i)}_i = (\mathbf{x}^{(1,j_i)}_i,\ \mathbf{x}^{(0,j_i)}_i,\ \sigma_i,\ \pi^{(j_i)}_i,\ 0,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}^*_i} \enspace .$$

$\underline{\tilde{\mathcal{O}}_{\mathbf{c}}\mathbf{:}}$ *On inputs $(k, i, \mathbf{x}^{(\mathsf{rep})}_{k,i}) \in [K] \times [H] \times \mathbb{Z}_q$, where $\mathsf{rep} \in [J_i]$ is a counter for the number of queries of the form $(k, i, \star)$, sample $\pi^{(\mathsf{rep})}_{k,i} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{c}^{(\mathsf{rep})}_{k,i} = (\mathbf{x}^{(\mathsf{rep})}_{k,i},\ \mathbf{x}^{(\mathsf{rep})}_{k,i},\ \sigma_{k,i},\ \pi^{(\mathsf{rep})}_{k,i},\ 0,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}^*_i} \enspace .$$

If $\sum_{i=1}^{H} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle = 0$ *and* $\sum_{i=1}^{H} \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle = 0$ *for all* $\tilde{j}_i \in [\widetilde{J}_i], \mathsf{rep}, j_i \in [J_i]$*, then the following advantage is negligible under the* SXDH *assumption:*

$$\begin{aligned} \Big| &\Pr[\mathcal{A}^{\tilde{\mathcal{O}}_{\mathbf{d}}, \boxed{\mathcal{O}^0_{\mathbf{d}}}}_{\tilde{\mathcal{O}}_{\mathbf{c}}, \mathcal{O}_{\mathbf{c}}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i \in [H]}, R, (R_k)_{k \in [K]}\Big) \to 1] \\ &\quad - \Pr[\mathcal{A}^{\tilde{\mathcal{O}}_{\mathbf{d}}, \boxed{\mathcal{O}^1_{\mathbf{d}}}}_{\tilde{\mathcal{O}}_{\mathbf{c}}, \mathcal{O}_{\mathbf{c}}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i \in [H]}, R, (R_k)_{k \in [K]}\Big) \to 1] \Big| \\ &\leq (4n\tilde{J}N + 4) \cdot \mathbf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1, \mathbb{G}_2}(1^\lambda) \end{aligned}$$

where $\mathcal{A}$ *can query the oracles* $\tilde{\mathcal{O}}_{\mathbf{d}}, \boxed{\mathcal{O}^b_{\mathbf{d}}}, \mathcal{O}_{\mathbf{c}}, \tilde{\mathcal{O}}_{\mathbf{c}}$ adaptively*, i.e. the queries can be made in any order and any number of times respecting the (polynomial) upper bounds* $K, L, (J_i, \widetilde{J}_i)_{i \in [H]}$.

We give an informal proof sketch of the main ideas. The full proof is presented in the full version [NPS24].

Outline of the Proof for Lemma 1. We explain the main steps in our proof as follows, where details about *formal* and *computational* basis changes can be revised from the examples in **Basis Changes** of the full version [NPS24]. The proof is done so that for *all* the repetitions $\tilde{j}_i \in [\tilde{J}_i]$, we perform the change from the repetition $\mathbf{y}_i^{(0,\tilde{j}_i)}$ into $\mathbf{y}_i^{(1,\tilde{j}_i)}$ by the $\tilde{j}_i$-th block of isolated coordinates in the vectors $\mathbf{d}_i^{(\tilde{j}_i)}$. It is crucial that the polynomially large bound $\tilde{J} \geq \max_{i \in [n], \mathsf{tag\text{-}f} \in \mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}}$ is known in advance, so as to well define the dimension of DPVS bases.

We start from the game where the sample given to the adversary $\mathcal{A}$ follows D_0 and the changes on vectors throughout the games are put in $\boxed{\text{boxes}}$. We use the notation $\mathbf{0} \coloneqq 0^N$ and write $\mathbf{0}^{\tilde{J}} \coloneqq \mathbf{0} \parallel \ldots \parallel \mathbf{0}$, for $\tilde{J}$ times. Our first step is to exploit the fact that $r \xleftarrow{\$} \mathbb{Z}_q$ is a uniformly random value and for each $j_i \in [J_i]$ all the secret shares σ_i in $\mathbf{c}_i^{(j_i)}$ sum to a known constant R. This helps us perform a *computational* basis change on $(\mathbf{B}_i, \mathbf{B}_i^*)$ and introduce a value $r' \xleftarrow{\$} \mathbb{Z}_q^*$ in $\mathbf{d}_i[2N + 2N \cdot \tilde{J} + 4]$ as well as random secret sharings of 0, common for $j_i \in [J_i]$, namely $(\tau_i)_{i=1}^H, (\tau'_{k,i})_{i=1}^H$, in $(\mathbf{c}_i^{(j_i)}[2N + 2N \cdot \tilde{j}_i + 4])_{i=1}^H$, $(\mathbf{c}_{k,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{j}_i + 4])_{i=1}^H$. We use the hypothesis that all basis vectors are kept secret so that the computational basis change using DDH cannot be detected by the adversary. More details can be found in the transition $\mathsf{G}_0 \to \mathsf{G}_1$.

After G_1, we perform a *formal* duplication to go to G_2 in which we duplicate coordinates $[1, N], [N+1, 2N]$ to the $\tilde{J}$ blocks $[2N \cdot \tilde{j} + 4, N + 2N \cdot \tilde{j} + 3], [N + 2N \cdot \tilde{j} + 4, 2N + 2N \cdot \tilde{j} + 3]$, where $\tilde{j}$ runs in $[\tilde{J}]$, in vectors $\mathbf{c}_i^{(j_i)}, \mathbf{c}_{k,i}^{(\mathsf{rep})}$ for all $i \in [H], k \in [K], j_i \in [J_i]$.

$$\begin{array}{rcc|c|c|c|c|c|c|cc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \Big(\mathbf{0} & \mathbf{0}\Big)^{\tilde{J}} & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{y}_i^{(1,\tilde{j}_i)} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \Big(\mathbf{0} & \mathbf{0}\Big)^{\tilde{J}} & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \Big(\boxed{\mathbf{x}_i^{(1,j_i)}} & \boxed{\mathbf{x}_i^{(0,j_i)}}\Big)^{\tilde{J}} & \tau_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \Big(\boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}}\Big)^{\tilde{J}} & \tau'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}$$

The duplication is done for *all* vectors $\mathbf{c}_i^{(j_i)}, \mathbf{c}_{k,i}^{(\mathsf{rep})}$ also across all repetitions $\mathsf{rep} \in [J]$. On a more technical level, this formal basis change will affect *all* vectors $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}, \mathbf{d}_i$ as well, also across all repetitions $\tilde{j}_i, \mathsf{rep} \in [\tilde{J}_i]$. Roughly speaking, by the duality of $(\mathbf{B}_i, \mathbf{B}_i^*)$, this basis change will incur "moving" coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{J} + 3], [N + 2N \cdot \tilde{J} + 4, 2N + 2N \cdot \tilde{J} + 3]$, for each $\tilde{j}_i \in [\tilde{J}]$ to $[1, N], [N+1, 2N]$ in the $\mathbf{d}$-vectors. In this simple $\mathsf{G}_1 \to \mathsf{G}_2$, the moved coordinates contain 0, so they do not pose any problems.

After G_2, in all $\mathbf{c}$-vectors, each of the $\tilde{J}$ blocks $[2N \cdot \tilde{j} + 4, N + 2N \cdot \tilde{j} + 3], [N + 2N \cdot \tilde{j} + 4, 2N + 2N \cdot \tilde{J} + 3]$ contains a copy of the coordinates $[1, N], [N + 1, 2N]$. This allows us to perform a *computational* basis change under SXDH in order to swap between $[1, N]$ and $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3]$ in $\mathbf{d}_i^{(\tilde{j}_i)}$, for each $\tilde{j}_i \in [\tilde{J}_i]$ and $\tilde{J}_i \leq \tilde{J}$ by definition. We stress that for different $\tilde{j}_i$, the swap will move contents of $[1, N]$ to separated coordinates in different $\mathbf{d}_i^{(\tilde{j}_i)}$. In other words, for every $\tilde{j}_i, \tilde{j}'_i$, the coordinates $[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3]$ is well defined for $\mathbf{d}_i^{(\tilde{j}_i)}$ because $\tilde{j}_i \leq \tilde{J}_i \leq \tilde{J}$ and we have

$$\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3] = \begin{cases} \mathbf{y}_i^{(1,\tilde{j}'_i)} & \text{if } \tilde{j}_i = \tilde{j}'_i \\ \mathbf{0} & \text{if } \tilde{j}_i \neq \tilde{j}'_i \end{cases} . \tag{8}$$

The randomness is taken from ρ_i at coordinate $2N + 3$ in $\mathbf{d}_i^{(\tilde{j}_i)}$.

$$\begin{array}{rc|c|c|c|c|c|c|c|c|cl}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \boxed{\mathbf{0}} & \mathbf{0} & r & 0 & \boxed{\rho_i^{(\tilde{j}_i)}} & \cdots & \boxed{\mathbf{y}_i^{(1,\tilde{j}_i)}} & \mathbf{0} & \cdots & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \cdots & \tau_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \cdots & \tau'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}$$

As a sanity check, we observe that this change preserves the products $\mathbf{d}_i^{(\tilde{j}_i)} \times \mathbf{c}_i^{(j_i)}$ and $\mathbf{d}_i^{(\tilde{j}_i)} \times \mathbf{c}_{k,i}^{(\mathsf{rep})}$ for all $k \in [K], \tilde{j}_i \in [\tilde{J}_i]$. Moreover, the computational basis change allows us to target only the vectors $(\mathbf{d}_i^{(\tilde{j}_i)})_{i \in [H]}$ while letting $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$ for $\ell \in [L], i \in [H]$ unchanged.

Upon reaching G_3, we are ready to approach the centerpiece of our proof. A *formal* basis change maintains perfectly identical views for the adversary in two games, resulting in a 0 difference in winning advantages under efficient simulation. We combine such formal basis changes with a *complexity leveraging* argument. In general, these kinds of arguments degrade the probability of a succesful simulation by an exponential factor. In our case, however, an exponential multiple of 0 is still 0. This implies that, as long as we restrict ourselves to formal bases changes that do not rely on any computational assumption, the simulator can initially guess all queries submitted by the adversary throughout the game, thus considering the selective game.

Formal basis changes highlight the information-theoretic properties of DPVS. However, they are often much harder to use than computational changes. The reason is that a formal basis change affects *all* vectors, including all repetitions, in the same manner. In contrast to computational changes, it is not possible to apply changes only to *some* vectors. Intuitively, this is why in G_2 and G_3 we had to move all repetitions $\mathbf{d}_i^{(\tilde{j}_i)}$ into separate coordinates to prepare for the formal basis changes.

We now explain the sequence of games on which the complexity leveraging is applied. We want to perform some sort of swapping between coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3]$ and $[N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ of $\mathbf{d}_i^{(\tilde{j}_i)}$ and reach G_6 whose vectors are:

$$\begin{array}{rc|c|c|c|c|c|c|c|c|cl}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{0} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \cdots & \boxed{\mathbf{0}} & \boxed{\mathbf{y}_i^{(0,\tilde{j}_i)}} & \cdots & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \boxed{\mathbf{x}_i^{(1,j_i)}} & \boxed{\mathbf{x}_i^{(0,j_i)}} & \cdots & \tilde{\tau}_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \cdots & \tilde{\tau}'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}$$

The complexity leveraging will be applied to the *selective* versions $\mathsf{G}_3^* \to \mathsf{G}_4^* \to \mathsf{G}_5^* \to \mathsf{G}_6^*$ and only *formal* basis changes will be used in between. In these selective versions the simulator guesses the values $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})_{i \in [H]}^{\tilde{j}_i \in [\tilde{J}_i], j_i \in [J_i]}$ and the hybrids are conditioned on a "good" event that these guesses are correct. The "good" event happens with fixed probability. This leads to an identical adversary's view:

$$\Pr[\mathsf{G}_3^* = 1] = \Pr[\mathsf{G}_4^* = 1] = \Pr[\mathsf{G}_5^* = 1] = \Pr[\mathsf{G}_6^* = 1] \ . \tag{9}$$

We briefly highlight the selective games' ideas below:

- In $\mathsf{G}_3^* \to \mathsf{G}_4^*$ a formal basis change is applied to do a quotient by $\mathbf{y}_i^{(1,\tilde{j}_i)}[z]$ for $z \in [N]$ over all $\tilde{J}$ blocks $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3], [N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$, where $\tilde{j}_i$ runs in $[\tilde{J}_i]$, of $\mathbf{c}$-vectors. We note that thanks to (8), for $\tilde{j}_i \neq \tilde{j}'_i \in [\tilde{J}]$, this change makes $\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3] = \mathbf{1}$ while $\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3] = \mathbf{0}$ for $\tilde{j}'_i \neq \tilde{j}_i$.

- In $\mathsf{G}_4^* \to \mathsf{G}_5^*$, we define a formal basis change that uses the *fixed* randomness $r' \in \mathbb{Z}_q^*$ in $\mathbf{d}_i^{(\tilde{j}_i)}[2N + 2N \cdot \tilde{j}_i + 4]$ (introduced from G_1) to switch 1 to 0 at coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 4]$ while marking 1 at coordinates $[N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ of $\mathbf{d}_i^{(\tilde{j}_i)}$, for *all* $\tilde{j}_i$. Thanks to the observation at the end of G_3^*, for each repetition $\mathbf{d}_i^{(\tilde{j}_i)}$ only the $\tilde{j}_i$-th blocks $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3], [N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ is affected, while other blocks stay $\mathbf{0}$. We note that unlike $\mathbf{d}_i^{(\tilde{j}_i)}$, the vectors $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$ stay invariant because $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{J} + 4] = 0$.

  Dually, because of the formal duplication in G_2 to all $\tilde{J} \geq \tilde{J}_i$ blocks, all $\mathbf{c}$-vectors will be altered such that the *accumulated* differences

$$\sum_{\substack{\tilde{j}_i \in [\tilde{J}_i] \\ z \in [N]}} \mathbf{c}_i^{(j_i)}[2N \cdot \tilde{j}_i + 3 + z] - \mathbf{c}_i^{(j_i)}[N + 2N \cdot \tilde{j}_i + 3 + z] = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle$$

  will be added to τ_i in $\mathbf{c}_i^{(j_i)}[2N + 2N \cdot \tilde{J} + 4]$. For $\mathbf{c}_{k,i}^{(\mathsf{rep})}$, similarly, we have the accumulated differences added to $\tau'_{k,i}$ is

$$\sum_{\substack{\tilde{j}_i \in [\tilde{J}_i] \\ z \in [N]}} \mathbf{c}_{k,i}^{(\mathsf{rep})}[2N \cdot \tilde{j}_i + 3 + z] - \mathbf{c}_{k,i}^{(\mathsf{rep})}[N + 2N \cdot \tilde{j}_i + 3 + z] = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})} \rangle \ .$$

  To show that this compensation for the accumulated differences in the τ_i and $\tau'_{k,i}$ cannot be noticed by the adversary, we exploit the conditions on the oracle queries in the statement of the lemma. Specifically, the condition $\sum_{i=1}^{H} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle = 0$ implies that $\frac{1}{r'}(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle)$ is constant for all $\tilde{j}_i \in [\tilde{J}], j_i \in [J_i]$ and $\sum_{i \in [H]} \frac{1}{r'}(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle) = 0$. From this observation, it follows that after adding the value $1/r' \cdot \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle$ to τ_i for all $i \in [H]$, $(\tau_i)_{i \in [H]}$ is still a secret sharing of 0. The same reasoning applies for $1/r' \cdot \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})} \rangle$ which is added to the secret sharing $(\tau'_{k,i})_{i=1}^{H}$ in $(\mathbf{c}_{k,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{J} + 4])_{i=1}^{H}$.

- In $\mathsf{G}_5^* \to \mathsf{G}_6^*$ we redo the quotient, still being in the selective variants conditioned on the "good" event.

- Finally, we also emphasize that all above DPVS formal basis changes do *not* depend on the *exponentially large* number of combinations $(\mathbf{d}_i^{(\tilde{j}_i)})_{i \in [H]}$, up to repetitions $\tilde{j}_i \in [\tilde{J}_i]$. We use the fact that each $i \in [H]$ has its vectors written in an *independent* pair of bases $(\mathbf{B}_i, \mathbf{B}_i^*)$, along with the crucial property (8) that allows treating each $\tilde{j}_i$-th repetition in an isolated block of the $\mathbf{d}_i^{(\tilde{j}_i)}$ vector, all $(\mathbf{d}_i^{(\tilde{j}_i)})^{\tilde{j}_i \in [\tilde{J}_i]}$ at the same time. To summarize, the specific information theoretic property of DPVS formal basis changes makes sure that *all* vectors in $(\mathbf{B}_i, \mathbf{B}_i^*)$ will be modified according to the basis matrices. For different $\tilde{j}_i \neq \tilde{j}'_i$ property (8) makes sure those matrices' change are trivial, *i.e.* $\mathbf{0}$ stays $\mathbf{0}$, in $\tilde{j}'_i$-th block of $\mathbf{d}_i^{(\tilde{j}_i)}$. Furthermore, even though all $\tilde{J}_i \leq \tilde{J}$ blocks of $\mathbf{c}_i^{(j_i)}$ are changed consistently by the matrices, in terms of the contents of all $\mathbf{d}_i^{(\tilde{j}_i)}$, different $\mathbf{c}_i^{(j_i)}$ from different i cannot be combined because they are in different bases. The only constraint is a fixed polynomially large upper bound $\tilde{J} \geq \max_{i \in [n], \mathsf{tag\text{-}f} \in \mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}}$ so that the dimensions are well defined.

The probability calculation of the complexity leveraging makes use of the fact that the "good" event happens with a fixed probability in conjunction with property (9), leading to $\Pr[\mathsf{G}_3 = 1] = \Pr[\mathsf{G}_4 = 1] = \Pr[\mathsf{G}_5 = 1] = \Pr[\mathsf{G}_6 = 1]$. Coming out of the complexity-leveraging argument, the very last step consists in swapping $\mathbf{x}_i$ from coordinates $[N + 2N \cdot \tilde{j}_i + 3, 2N + 2N \cdot \tilde{j}_i + 3]$ back to $[1, N]$ (see $\mathsf{G}_6 \to \mathsf{G}_7$) and some cleaning in order to make the vectors follow D_1 (see $\mathsf{G}_7 \to \mathsf{G}_8$).

4.2 Basic Construction

This section presents our basic adaptively secure FH-DMCFE construction $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for the function class $\mathcal{F}^{\mathsf{ip}}$, where each client encrypts a vector of length $N \in \mathbb{N}$. We obtain the adaptive scheme by giving a concrete instantiation for the FH-IPFE scheme iFE used in our selectively secure FH-DMCFE from Figure 1. As a reminder, we refer to the beginning of Section 3.3 for the notations, inlcuding those of implicit represention for group elements and the bilinear group setting. The notations of DPVS and the writing of their vectors with respect to the dual bases are recalled in Section 3.2.

Our FH-IPFE instantiation is extremely simple. The master secret key is a pair of random dual bases $(\mathbf{B}, \mathbf{B}^*)$. To generate a key for some vector $\mathbf{y} \in \mathbb{Z}_q^N$, we sample $\pi \overset{\$}{\leftarrow} \mathbb{Z}_q$ and return $\mathbf{d} = (\mathbf{y}, \pi, 0, \mathbf{0})_{\mathbf{B}^*}$ as decryption key. Similarly, to encrypt a vector $\mathbf{x} \in \mathbb{Z}_q^N$, we sample $\rho \overset{\$}{\leftarrow} \mathbb{Z}_q$ and output $\mathbf{c} = (\mathbf{x}, 0, \pi, \mathbf{0})_{\mathbf{B}}$ as ciphertext. Decryption computes $[\![z]\!]_{\mathsf{t}} = \mathbf{c} \times \mathbf{d}$, then finds and outputs the discrete log z. When plugging this FH-IPFE into Figure 1, we obtain our adaptively secure scheme whose details are given in Figure 3.

Correctness. The correctness property is demonstrated as follows:

$$[\![\mathsf{out}]\!]_{\mathsf{t}} = \sum_{i=1}^{n} \mathbf{c}_i \times \mathbf{d}_i = \sum_{i=1}^{n} [\![\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \mu\omega \cdot \tilde{t}_i]\!]_{\mathsf{t}}$$

$$= \left[\!\!\left[\sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle + \mu\omega \cdot \sum_{i=1}^{n} \tilde{t}_i \right]\!\!\right]_{\mathsf{t}} = \left[\!\!\left[\sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle \right]\!\!\right]_{\mathsf{t}} ,$$

and we are using the fact that $\sum_{i=1}^{n} \tilde{t}_i = 0$. Theorem 1 states that the scheme given in Fig. 3 is *function-hiding*, *one-challenge* secure against *complete queries* under *static corruption*. An unbounded number of ciphertext repetitions is allowed, while the number of key repetitions is fixed as a parameter of the scheme. In Section 5, we argue that most restrictions on the security model can be removed by applying a sequence of generic lemmas.

Theorem 1. *The DMCFE scheme* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *in Fig. 3 for the function class* $\mathcal{F}^{\mathsf{ip}}$ *is* one-challenge, function-hiding *secure against* complete *queries under* static *corruption in the ROM, if the* SXDH *assumption holds for* $(\mathbb{G}_1, \mathbb{G}_2)$.

More specifically, we let q_e and q_k denote the maximum number of distinct tags queried to $\mathcal{O}\mathsf{Enc}$ and $\mathcal{O}\mathsf{KeyGen}$, respectively. Furthermore, for $i \in [n]$ and $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, we define $\tilde{J}_{i,\mathsf{tag\text{-}f}}$ to be the numbers of queries of the form $\mathcal{O}\mathsf{KeyGen}(i, \mathsf{tag\text{-}f}, \star, \star)$. We require that $\max_{i \in [n], \mathsf{tag\text{-}f} \in \mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}} \leq \tilde{J}$, where $\tilde{J}$ is specified by the DMCFE scheme at Setup *time. Then, for any* ppt *adversary $\mathcal{A}$ against $\mathcal{E}$, we have the following bound:*

$$\mathbf{Adv}^{\mathsf{1chal\text{-}pos\text{-}stat\text{-}fh}}_{\mathcal{E}, \mathcal{F}^{\mathsf{ip}}, \mathcal{A}}(1^\lambda) \leq \Big((q_k + 1) \cdot (4n\tilde{J}N + 4) + 4N + q_e + 1 \Big) \cdot \mathbf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1, \mathbb{G}_2}(1^\lambda)$$

The proof of Theorem 1 follows exactly the proof sketch of the selective scheme in Section 2. As explained in the paragraph **Problems for Adaptive Security**, the main difficulty towards adaptive security lies in enabling the steps (3) to (5) in a sequence of hybrids

$\mathsf{Setup}(1^\lambda, 1^n)$**:** Sample matrices $(B_i, B_i^*) \leftarrow \mathsf{DPVSGen}(\mathbb{G}, 1^{2N\cdot(\tilde{J}+1)+4})$, for $i \in [n]$, of dimensions $2N \cdot (\tilde{J}+1)+4$ that specify dual orthogonal bases $(\mathbf{B}_i, \mathbf{B}_i^*)$[a]. Sample $(\tilde{t}_i)_i \xleftarrow{\$} \mathbb{Z}_q^n$ conditioned on $\sum_{i=1}^n \tilde{t}_i = 0$. Output the public parameters $\mathsf{pp} := \mathbb{G}$, secret keys sk_i and the encryption keys ek_i for all $i \in [n]$ as follows:

$$\mathsf{sk}_i := (\mathbf{b}_{i,1}^*, \ldots, \mathbf{b}_{i,N}^*,\ B_{i,N+1}^*,\ \mathbf{b}_{i,N+2}^*), \quad \mathsf{ek}_i := \big(\tilde{t}_i, (\mathbf{b}_{i,1}, \ldots, \mathbf{b}_{i,N},\ B_{i,N+1},\ \mathbf{b}_{i,N+3})\big)$$

$\mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, \mathbf{y}_i)$**:** Parse $\mathsf{sk}_i = (\mathbf{b}_{i,1}^*, \ldots, \mathbf{b}_{i,N}^*,\ B_{i,N+1}^*,\ \mathbf{b}_{i,N+2}^*)$, compute $\mathsf{H}_2(\mathsf{tag\text{-}f}) \rightarrow \llbracket \mu \rrbracket_2$ and sample $\pi_i \xleftarrow{\$} \mathbb{Z}_q$. Then output[b]

$$\mathbf{d}_i = \sum_{k=1}^{N} \mathbf{y}_i[k]\mathbf{b}_{i,k}^* + \llbracket \mu \rrbracket_2 \cdot B_{i,N+1}^* + \pi_i \mathbf{b}_{i,N+2}^* = (\mathbf{y}_i,\ \mu,\ \pi_i,\ 0,\ 0^{N+2N\cdot\tilde{J}+1})_{\mathbf{B}_i^*} \ .$$

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, \mathbf{x}_i)$**:** Parse $\mathsf{ek}_i = (\tilde{t}_i, (\mathbf{b}_{i,1}, \ldots, \mathbf{b}_{i,N},\ B_{i,N+1},\ \mathbf{b}_{i,N+3}))$, compute $\mathsf{H}_1(\mathsf{tag}) \rightarrow \llbracket \omega \rrbracket_1$ and sample a random scalar $\rho_i \xleftarrow{\$} \mathbb{Z}_q$. Then output[c]

$$\mathbf{c}_i = \sum_{k=1}^{N} \mathbf{x}_i[k]\mathbf{b}_{i,1} + \tilde{t}_i \llbracket \omega \rrbracket_1 \cdot B_{i,N+1} + \rho_i \mathbf{b}_{i,N+3} = (\mathbf{x}_i,\ \tilde{t}_i\omega,\ 0,\ \rho_i,\ 0^{N+2N\cdot\tilde{J}+1})_{\mathbf{B}_i} \ .$$

$\mathsf{Dec}(\mathbf{d}, \mathbf{c})$**:** Parse $\mathbf{d} := (\mathbf{d}_i)_{i\in[n]}$ and $\mathbf{c} := (\mathbf{c}_i)_i$. Compute $\llbracket \mathsf{out} \rrbracket_{\mathsf{t}} = \sum_{i=1}^n \mathbf{c}_i \times \mathbf{d}_i$, then find and output the discrete log out.

[a] For each $i \in [n]$, we denote j-th row of $\mathbf{B}_i$ (resp. $\mathbf{B}_i^*$) by $\mathbf{b}_{i,j}$ (resp. $\mathbf{b}_{i,j}^*$). Similarly, $B_{i,k}$ (respectively $B_{i,k}^*$) denotes the k-th row of the basis changing matrix B_i (respectively B_i^*).

[b] Throughout the computation of $\mathbf{d}_i$, only the hash value $\llbracket \mu \rrbracket_2 \in \mathbb{G}_2$ is used, never μ in the clear.

[c] Throughout the computation of $\mathbf{c}_i$, only the hash value $\llbracket \omega \rrbracket_1 \in \mathbb{G}_1$ is used, never ω in the clear.

Figure 3: FH-DMCFE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for inner products. We work in the prime-order bilinear group setting $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_{\mathsf{t}}, g_1, g_2, g_{\mathsf{t}}, \mathbf{e}, q)$ and use two full-domain hash functions $\mathsf{H}_1 \colon \mathsf{Tag} \rightarrow \mathbb{G}_1$ and $\mathsf{H}_2 : \mathsf{Tag} \rightarrow \mathbb{G}_2$. Let $\tilde{J} = \mathrm{poly}(\lambda)$.

without knowing $\Delta_i^{(b)}$ and $\Delta_{\ell,i}^{(b)}$ in advance. In the DPVS setting, the transition from one hybrid to the next corresponds exactly to an application of Lemma 1. Even though $\tilde{J}$ is fixed, it can be polynomially large leading to an exponentially number of combinations of key repetitions, this is also handled by Lemma 1. We refer to the high level in section 4.1. The full proof of theorem 1 can be found in the full version [NPS24].

5 Upgrading Security

In this section, we give a sequence of generic lemmas that can be used to strengthen the security model of our basic FH-DMCFE construction from Section 4.2. Specifically, we show how to remove the *complete-query* constraint and the restriction to *one-challenge* security. In this way, we obtain an FH-DMCFE for inner products whose only restrictions on the security model are *static corruptions* and a *polynomially bounded number of repetitions for decryption keys.*

Security against Incomplete Queries. To remove the complete-queries constraint, previous works [CDSG$^+$20, AGT21b] make use of a technique called *all-or-nothing encapsulation* (AoNE). Roughly, AoNE allows all parties of a group to encapsulate individual messages, that can *all* be extracted by everyone if and only if all parties of the group have sent their contribution. Otherwise, *no* message is revealed. In the constructions of [CDSG$^+$20, AGT21b], such an AoNE layer is added on top of both ciphertexts and keys. Intuitively, this approach allows the following reasoning: if an adversary makes encryption queries for all (honest) clients under some tag tag (i.e. the global query is "complete"), then the AoNE scheme allows to obtain all ciphertexts, and we can rely on the security of the DMCFE scheme that is secure against complete challenges. On the other hand, if the adversary queries only some but not all honest clients (i.e. the global query is "incomplete"), then the security of the AoNE scheme guarantees that the adversary does not learn anything about the encapsulated messages. While this construction is well known, previous constructions prove only selective security, even if the employed AoNE scheme is adaptively secure. Therefore, we think it is important to show that this AoNE layer indeed preserves adaptive security if the underlying scheme, which is only secure against complete queries, has this property.

More specifically, the notion of AoNE is a particular functionality of DDFE introduced by Chotard *et al.* [CDSG+20]. In [AGT21b], AoNE also serves as a building block for their FH-DDFE scheme, and it is pointed out that function-hiding and standard security are the same for AoNE, as there is no concept of keys. Since we are focusing on the less general notion DMCFE, we define AoNE in a less general context as a functionality for DMCFE.

Definition 7 (All-or-Nothing Encapsulation)**.** For $n, \lambda \in \mathbb{N}$, let $\mathsf{Tag}_\lambda = \mathcal{R}_\lambda = \{0,1\}^{\mathrm{poly}(\lambda)}$, $\mathcal{K}_\lambda = \varnothing$, $\mathcal{M}_{n,\lambda,\mathrm{pub}} = [n] \times \mathsf{Tag}_\lambda$ and $\mathcal{M}_{\lambda,\mathrm{pri}} = \{0,1\}^L$ for a polynomial $L = L(\lambda)\colon \mathbb{N} \to \mathbb{N}$. The *all-or-nothing encapsulation* functionality $f^{\mathsf{aone}} = \{f^{\mathsf{aone}}_{n,\lambda}\colon \{[n]\} \times (\{[n]\} \times \mathcal{M}_\lambda)^n \to \mathcal{R}_\lambda\}_{n,\lambda\in\mathbb{N}}$ is defined via

$$f^{\mathsf{aone}}_{n,\lambda}([n], (i, m_i)_{i\in[n]}) = \begin{cases} (x_i)_{i\in[n]} & \text{if condition } (*) \text{ holds} \\ \perp & \text{otherwise} \end{cases}$$

for all $n, \lambda \in \mathbb{N}$, where $\{[n]\}$ is a singleton consisting of $[n]$ as its only member, and condition $(*)$ holds if there exists $\mathsf{tag} \in \mathsf{Tag}_\lambda$ such that for each $i \in [n]$, m_i is of the form $(m_{i,\mathrm{pri}} \coloneqq x_i \in \{0,1\}^L, m_{i,\mathrm{pub}} \coloneqq ([n], \mathsf{tag}) \in \mathcal{M}_{n,\lambda,\mathrm{pub}})$.

This means in particular that DKeyGen is unnecessary and Dec works without taking secret keys as input. The DDFE constructions from [CDSG+20] yield two constructions of DMCFE for the function class AoNE *as per* Definition 7. A first generic construction [CDSG+20, Section 4] from identity-based encryption is secure in the standard model. Another concrete construction [CDSG+20, Section 5] from bilinear maps under the Decisional Bilinear Diffie-Hellman (DBDH) assumption is proven secure in the ROM.

We present our result in form of a generic conversion that turns any one-challenge DMCFE scheme secure against complete queries into one that is also secure against incomplete queries.

Lemma 2. *Assume there exist (1) a one-challenge (weakly function-hiding) DMCFE scheme $\mathcal{E}^{\mathsf{pos}}$ for a function class $\mathcal{F}$ that is secure against complete queries, and (2) an AoNE scheme $\mathcal{E}^{\mathsf{aone}}$ whose message space contains the ciphertext space of $\mathcal{E}^{\mathsf{pos}}$. Then there exists a one-challenge (weakly function-hiding) DMCFE scheme $\mathcal{E}$ for $\mathcal{F}$ that is even secure against incomplete queries. More precisely, for any* ppt *adversary $\mathcal{A}$, there exist* ppt *algorithms $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$\mathbf{Adv}^{\mathsf{1chal\text{-}xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) \le 12 \cdot \mathbf{Adv}^{\mathsf{1chal\text{-}pos\text{-}xxx\text{-}wfh}}_{\mathcal{E}^{\mathsf{pos}},\mathcal{F},\mathcal{B}_1}(1^\lambda) + 12 \cdot \mathbf{Adv}^{\mathsf{1chal\text{-}xxx\text{-}wfh}}_{\mathcal{E}^{\mathsf{aone}},f^{\mathsf{aone}},\mathcal{B}_2}(1^\lambda) \ ,$$

where $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}\}$.

Our conversion simply adds a layer of DMCFE for AoNE on top of both ciphertexts and keys. On an intuitive level, our simulator initially guesses whether or not the oracle queries for the challenge tag $\mathsf{tag\text{-}f}^*$ (or tag^*) will be complete. If the guess was "complete" and this guess turns out to be correct at the end of the game, then the simulator attacks the underlying DMCFE scheme that is assumed to be secure against complete queries. If the guess was "incomplete" and the guess is correct, then the simulator attacks the security of the AoNE scheme. If the guess was incorrect (which happens with probability 1/2), then the simulator aborts with a random bit. In this way, we can upper bound the advantage of a distinguisher between two successive hybrids in terms of the advantages that efficient adversaries can achieve against the underlying AoNE and DMCFE schemes. We point out that this argument crucially relies on the *one-challenge* setting. Due to the guess on the (in)completeness of the oracle queries, we lose a factor 1/2 in the security proof. Thus, a hybrid argument over a polynomial number of incomplete queries would incur an exponential security loss. Therefore, it is important to add security against incomplete queries in the one-challenge model.

Details about the conversion as well as the proof are given in the full version [NPS24]. We mention that a concurrent work by Shi and Vanjani [SV23] presents a similar conversion in the MCFE setting.

Security against Multiple Challenges. It remains to discuss how a one-challenge FH-DMCFE scheme for inner products can be made resistant against multiple challenge queries. First, observe that the equivalence of one-challenge and multi-challenge security in the standard setting (without function privacy) is trivial. Indeed, the proof can be done by a sequence of hybrids over the different tags queried to the encryption oracle. This approach, however, does not directly generalize to the function-hiding setting. The problem is that now both encryption and key-generation queries depend on the challenge bit $b \in \{0,1\}$. Since ciphertexts and keys can be arbitrarily combined in general, such a sequence of hybrids leads to a situation where an adversary is able to mix ciphertexts that encrypt the left message with keys generated for the right function or vice versa. However, the function-hiding admissibility does not provide any security guarantees in the case of such a mixed decryption. Therefore, we cannot change ciphertexts and keys one by one anymore. We solve this problem by first proving security against multiple challenges in the *weakly function-hiding* setting. This model provides us exactly with the necessary guarantee for mixed decryptions, which allows a hybrid argument over all function and message tags to subsequently swap keys and ciphertexts. Afterwards, we apply another standard transformation that turns weakly function-hiding DMCFE schemes for inner products back into full-fledged function-hiding DMCFE (see Lemma 4). Previous works [LV16, ACF$^+$18] presented that transformation for single-input and multi-input FE schemes.

We state the formal lemmas below. The proofs are standard and the latter is very similar to [LV16, ACF$^+$18], but we give them in the full version [NPS24] for completeness.

Lemma 3. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a DMCFE scheme for the function class* $\mathcal{F}$*. If* $\mathcal{E}$ *is one-challenge weakly function-hiding, then it is also weakly function-hiding. More specifically, for any* ppt *adversary* $\mathcal{A}$*, there exists a* ppt *algorithm* $\mathcal{B}$ *such that*

$$\mathbf{Adv}^{\mathsf{xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) \leq (q_e + q_k) \cdot \mathbf{Adv}^{\mathsf{1chal\text{-}xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F},\mathcal{B}}(1^\lambda) \ ,$$

where q_e *and* q_k *denote the maximum numbers of different tags* tag *and* tag-f *that* $\mathcal{A}$ *can query to* $\mathcal{O}\mathsf{Enc}$ *and* $\mathcal{O}\mathsf{DKeyGen}$ *respectively, and* $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}, \mathsf{pos}\}$.

Lemma 4. *If there exists a weakly function-hiding DMCFE scheme* $\mathcal{E}$ *for* $\mathcal{F}^{\mathsf{ip}}$*, then there exists a (fully) function-hiding DMCFE scheme* $\mathcal{E}'$ *for* $\mathcal{F}^{\mathsf{ip}}$*. More precisely, for any* ppt *adversary* $\mathcal{A}$*, there exists a* ppt *algorithm* $\mathcal{B}$ *such that*

$$\mathbf{Adv}^{\mathsf{xxx\text{-}fh}}_{\mathcal{E}',\mathcal{F}^{\mathsf{ip}},\mathcal{A}}(1^\lambda) \leq 3 \cdot \mathbf{Adv}^{\mathsf{xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F}^{\mathsf{ip}},\mathcal{B}}(1^\lambda) \ ,$$

where $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}, \mathsf{1chal}, \mathsf{pos}\}$.

Concrete Instantiation. Given Lemmas 2, 3, and 4, we now generically transform our FH-DMCFE from Section 4.2 to upgrade its security. Specifically, we first apply Lemma 2 and follow the generic IBE-based AoNE from [CDSG+20, Section 4]. We use any adaptively secure pairing-based IBE [CLL+13, JR17] under SXDH[7] to obtain generically a DMCFE for AoNE, in order to allow *incomplete queries*. We then use Lemma 3 to allow *multiple challenges*, while downgrading from function-hiding to weak function-hiding. Finally, we apply Lemma 4 to re-establish full-fledged *function-hiding*. The final scheme is summarized in the below corollary, with newly accomplished properties being underlined.

Corollary 1. *There exists an FH-DMCFE scheme for the function class $\mathcal{F}^{\mathsf{ip}}$ that is adaptively function-hiding secure against static corruption, while allowing unbounded repetitions for ciphertext queries and a fixed polynomially large number of repetitions for key-generation queries, under the* SXDH *assumption in the ROM.*

Acknowledgements

This work was supported in part by the French ANR Project ANR-19-CE39-0011 PRESTO and the France 2030 ANR Project ANR-22-PECY-003 SecureCompute.

References

[ABDP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Heidelberg, March / April 2015. doi:10.1007/978-3-662-46447-2_33.

[ABG19] Michel Abdalla, Fabrice Benhamouda, and Romain Gay. From single-input to multi-client inner-product functional encryption. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 552–582. Springer, Heidelberg, December 2019. doi:10.1007/978-3-030-34618-8_19.

[ABKW19] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. Decentralizing inner-product functional encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 128–157. Springer, Heidelberg, April 2019. doi:10.1007/978-3-030-17259-6_5.

[ACF+18] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 597–627. Springer, Heidelberg, August 2018. doi:10.1007/978-3-319-96884-1_20.

[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, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, Heidelberg, December 2020. doi:10.1007/978-3-030-64840-4_16.

[AGT21a] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption from pairings. In Tal Malkin and Chris Peikert, editors,

[7] The seminal adaptively secure group-based (H)IBE is [Wat09] but it relies on both DDH and D-Lin.

CRYPTO 2021, Part IV, volume 12828 of *LNCS*, pages 208–238, Virtual Event, August 2021. Springer, Heidelberg. doi:10.1007/978-3-030-84259-8_8.

[AGT21b] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 224–255. Springer, Heidelberg, November 2021. doi:10.1007/978-3-030-90453-1_8.

[AGT22] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption: Stronger security, broader functionality. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 711–740. Springer, Heidelberg, November 2022. doi:10.1007/978-3-031-22318-1_25.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 308–326. Springer, Heidelberg, August 2015. doi:10.1007/978-3-662-47989-6_15.

[ALdP11] Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 90–108. Springer, Heidelberg, March 2011. doi:10.1007/978-3-642-19379-8_6.

[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, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53015-3_12.

[AS17] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 152–181. Springer, Heidelberg, April / May 2017. doi:10.1007/978-3-319-56620-7_6.

[ATY23] Shweta Agrawal, Junichi Tomida, and Anshu Yadav. Attribute-based multi-input FE (and more) for attribute-weighted sums. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 464–497. Springer, Heidelberg, August 2023. doi:10.1007/978-3-031-38551-3_15.

[BBL17] Fabrice Benhamouda, Florian Bourse, and Helger Lipmaa. CCA-secure inner-product functional encryption from projective hash functions. In Serge Fehr, editor, *PKC 2017, Part II*, volume 10175 of *LNCS*, pages 36–66. Springer, Heidelberg, March 2017. doi:10.1007/978-3-662-54388-7_2.

[BCFG17] Carmen Elisabetta Zaira Baltico, Dario Catalano, Dario Fiore, and Romain Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 67–98. Springer, Heidelberg, August 2017. doi:10.1007/978-3-319-63688-7_3.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001. doi:10.1007/3-540-44647-8_13.

[BJK15] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Heidelberg, November / December 2015. doi:10.1007/978-3-662-48797-6_20.

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. doi:10.1007/11761679_25.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011. doi:10.1007/978-3-642-19571-6_16.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015. doi:10.1109/FOCS.2015.20.

[CDG$^+$18a] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 703–732. Springer, Heidelberg, December 2018. doi:10.1007/978-3-030-03329-3_24.

[CDG$^+$18b] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with repetition for inner product. Cryptology ePrint Archive, Report 2018/1021, 2018. https://eprint.iacr.org/2018/1021.

[CDSG$^+$20] Jérémy Chotard, Edouard Dufour-Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Dynamic decentralized functional encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 747–775. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56784-2_25.

[CLL$^+$13] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. In Michel Abdalla and Tanja Lange, editors, *PAIRING 2012*, volume 7708 of *LNCS*, pages 122–140. Springer, Heidelberg, May 2013. doi:10.1007/978-3-642-36334-4_8.

[CLT18] Guilhem Castagnos, Fabien Laguillaumie, and Ida Tucker. Practical fully secure unrestricted inner product functional encryption modulo p. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 733–764. Springer, Heidelberg, December 2018. doi:10.1007/978-3-030-03329-3_25.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer, Heidelberg, December 2001. doi:10.1007/3-540-45325-3_32.

[DDM16] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. Functional encryption for inner product with full function privacy. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 164–195. Springer, Heidelberg, March 2016. doi:10.1007/978-3-662-49384-7_7.

[DOT18] Pratish Datta, Tatsuaki Okamoto, and Junichi Tomida. Full-hiding (unbounded) multi-input inner product functional encryption from the k-Linear assumption. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 245–277. Springer, Heidelberg, March 2018. doi:10.1007/978-3-319-76581-5_9.

[EHK$^+$13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013. doi:10.1007/978-3-642-40084-1_8.

[Gay20] Romain Gay. A new paradigm for public-key functional encryption for degree-2 polynomials. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 95–120. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45374-9_4.

[GGG$^+$14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Heidelberg, May 2014. doi:10.1007/978-3-642-55220-5_32.

[GKL$^+$13] S. Dov Gordon, Jonathan Katz, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774, 2013. https://eprint.iacr.org/2013/774.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. doi:10.1145/1180405.1180418.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 503–523. Springer, Heidelberg, August 2015. doi:10.1007/978-3-662-48000-7_25.

[JR17] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. *Journal of Cryptology*, 30(4):1116–1156, October 2017. doi:10.1007/s00145-016-9243-7.

[KKS19] Sungwook Kim, Jinsu Kim, and Jae Hong Seo. A new approach to practical function-private inner product encryption. *Theoretical Computer Science*, 783:22–40, 2019. doi:10.1016/J.TCS.2019.03.016.

[KLM$^+$18] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J. Wu. Function-hiding inner product encryption is practical. In Dario

Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 544–562. Springer, Heidelberg, September 2018. doi:10.1007/978-3-319-98113-0_29.

[Lin17] Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 599–629. Springer, Heidelberg, August 2017. doi:10.1007/978-3-319-63688-7_20.

[LT19] Benoît Libert and Radu Titiu. Multi-client functional encryption for linear functions in the standard model from LWE. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 520–551. Springer, Heidelberg, December 2019. doi:10.1007/978-3-030-34618-8_18.

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In Irit Dinur, editor, *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016. doi:10.1109/FOCS.2016.11.

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Heidelberg, February 2010. doi:10.1007/978-3-642-11799-2_27.

[NPP22] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with fine-grained access control. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 95–125. Springer, Heidelberg, December 2022. doi:10.1007/978-3-031-22963-3_4.

[NPP23] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Optimal security notion for decentralized multi-client functional encryption. In Mehdi Tibouchi and Xiaofeng Wang, editors, *ACNS 23, Part II*, volume 13906 of *LNCS*, pages 336–365. Springer, Heidelberg, June 2023. doi:10.1007/978-3-031-33491-7_13.

[NPS24] Ky Nguyen, David Pointcheval, and Robert Schädlich. Decentralized multi-client functional encryption with strong security. Cryptology ePrint Archive, Paper 2024/764, 2024. URL: https://eprint.iacr.org/2024/764.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 2007*, pages 195–203. ACM Press, October 2007. doi:10.1145/1315245.1315270.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, Heidelberg, August 2010. doi:10.1007/978-3-642-14623-7_11.

[OT12a] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 591–608. Springer, Heidelberg, April 2012. doi:10.1007/978-3-642-29011-4_35.

[OT12b] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 349–366. Springer, Heidelberg, December 2012. doi:10.1007/978-3-642-34961-4_22.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Heidelberg, August 1984. doi:10.1007/3-540-39568-7_5.

[SV23] Elaine Shi and Nikhil Vanjani. Multi-client inner product encryption: Function-hiding instantiations without random oracles. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 622–651. Springer, Heidelberg, May 2023. doi:10.1007/978-3-031-31368-4_22.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. doi:10.1007/11426639_27.

[TAO16] Junichi Tomida, Masayuki Abe, and Tatsuaki Okamoto. Efficient functional encryption for inner-product values with full-hiding security. In Matt Bishop and Anderson C. A. Nascimento, editors, *ISC 2016*, volume 9866 of *LNCS*, pages 408–425. Springer, Heidelberg, September 2016. doi:10.1007/978-3-319-45871-7_24.

[Tom19] Junichi Tomida. Tightly secure inner product functional encryption: Multi-input and function-hiding constructions. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 459–488. Springer, Heidelberg, December 2019. doi:10.1007/978-3-030-34618-8_16.

[Tom20] Junichi Tomida. Tightly secure inner product functional encryption: Multi-input and function-hiding constructions. *Theoretical Computer Science*, 833:56–86, 2020. doi:10.1016/J.TCS.2020.05.008.

[Üna20] Akin Ünal. Impossibility results for lattice-based functional encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 169–199. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45721-1_7.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009. doi:10.1007/978-3-642-03356-8_36.