Multi-Input Functional Encryption for Inner Products: Function-Hiding Realizations and Constructions without Pairings

Michel Abdalla[1,2], Dario Catalano[3], Dario Fiore[4],
Romain Gay[1,2], and Bogdan Ursu[5]

[1] Département informatique de l'ENS, École normale supérieure, CNRS, PSL University, 75005 Paris, France
{michel.abdalla,romain.gay}@ens.fr
[2] INRIA, Paris, France
[3] Dipartimento di Matematica e Informatica, Università di Catania, Italy.
catalano@dmi.unict.it
[4] IMDEA Software Institute, Madrid, Spain.
dario.fiore@imdea.org
[5] KIT, Karlsruhe, Germany.
bogdan.ursu@kit.edu

Abstract. We present new constructions of multi-input functional encryption (MIFE) schemes for the inner-product functionality that improve the state of the art solution of Abdalla *et al.* (Eurocrypt 2017) in two main directions.

First, we put forward a novel methodology to convert single-input functional encryption for inner products into multi-input schemes for the same functionality. Our transformation is surprisingly simple, general and efficient. In particular, it does not require pairings and it can be instantiated with *all* known single-input schemes. This leads to two main advances. First, we enlarge the set of assumptions this primitive can be based on, notably, obtaining new MIFEs for inner products from plain DDH, LWE, and Decisional Composite Residuosity. Second, we obtain the first MIFE schemes from standard assumptions where decryption works efficiently even for messages of super-polynomial size.

Our second main contribution is the first function-hiding MIFE scheme for inner products based on standard assumptions. To this end, we show how to extend the original, pairing-based, MIFE by Abdalla *et al.* in order to make it function hiding, thus obtaining a function-hiding MIFE from the MDDH assumption.

1 Introduction

Functional Encryption (FE) [SW05,O'N10,BSW11] is an emerging cryptographic paradigm that allows fine-grained access control over encrypted data. Functional encryption schemes come equipped with a key generation mechanism that allows the owner of a master secret key to generate decryption keys that have a somehow restricted capability. Namely, each decryption key sk_f is associated with a function f and using sk_f to decrypt a ciphertext $\mathsf{Enc}(x)$ allows for recovering $f(x)$, with the guarantee that no more information about x is revealed. The basic notion of functional encryption considers functionalities where all the inputs are provided and encrypted by a single party. The more general case of multi-input functionalities is captured by the notion of multi-input functional encryption (MIFE, for short) [GGG+14]. Informally, this notion can be thought of as an FE scheme where n encryption slots are explicitly given, in the sense that a user who is assigned the i-th slot can, independently, create a ciphertext $\mathsf{Enc}(x_i)$ from his own plaintext x_i. Given ciphertexts $\mathsf{Enc}(x_1), \ldots, \mathsf{Enc}(x_n)$, one can use a secret key sk_f to retrieve $f(x_1, \ldots, x_n)$, similarly to the basic FE notion. This multi-input capability makes MIFE particularly well suited for many real life scenarios (such as data mining over encrypted data or multi-client delegation of computation) where the (encrypted) data may come from different and unrelated sources.

The security requirement for both FE and MIFE imposes that decryption keys should be collusion resistant. This means that a group of users, holding different decryption keys, should not be able to gain information about the encrypted messages, beyond the union of what they can individually learn. More precisely, the standard notion of security for functional encryption is *indistinguishability*. Informally, this states that an adversary that obtains the secret keys corresponding to functions $f_1, \ldots, f_n$ should not be able to decide which of the challenge messages x_0, x_1 was encrypted, as long as $f_i(x_0) = f_i(x_1)$ for all i. This indistinguishability notion has been put forward in [BSW11,O'N10] and it has been shown inadequate for certain cases (see [BSW11,O'N10] for details). They also proposed an alternative simulation-based security notion which is also problematic as, for instance, it cannot be satisfied in general.

As an additional security property, functional encryption schemes might also be required to guarantee so-called *function hiding*. Intuitively, this means that a secret key sk_f should not reveal information about the function f it encodes, beyond what is implicitly leaked by $f(x)$. Slightly more in detail, in the indistinguishability setting, this is formalized by imposing that the adversary should not be able to decide for which of the challenge functions $f_i^{(0)}, f_i^{(1)}$ it is holding secret keys, as long as as $f_i^{(0)}(x_0) = f_i^{(1)}(x_1)$ for all i. Over the last few years, functional encryption has attracted a lot of interest, both in its basic and in its multi-input incarnations. Known results can be broadly categorized as focusing on (1) feasibility results for general functionalities, and on (2) concrete, efficient realizations for restricted functionalities of practical interest.

For the specific case of MIFE, which is the focus of this paper, constructions of the first type [GGG+14,BGJS15,AJ15,BKS16] all rely on quite unstable assumptions, such as indistinguishability obfuscation or multilinear maps[6]. The only known construction of the second category has been recently proposed by Abdalla *et al.* in [AGRW17]. There, they propose a (secret-key) MIFE scheme for the inner product functionality that relies on the standard k-linear assumption in (prime-order) bilinear groups[7]. Remarkably, their scheme allows for unbounded collusions and supports any

[6] Here we only consider schemes where *unbounded* collusions are allowed. See [BKS16] and references therein for the bounded collusions case.

[7] As discussed in detail in [AGRW17], we stress that in the public key setting, MIFE for inner products is both easy to achieve (from its single-input counterpart) and of very limited interest, because of its inherent leakage.

(polynomially bounded) number of encryption slots. On the negative side, as in previous discrete-log-based constructions of functional inner-product encryption schemes, it employs an inefficient decryption procedure that requires to extract discrete logarithms and thus imposes serious restrictions on the size of supported messages. Moreover, the scheme is not function hiding as decryption requires the function f to be provided explicitly in the clear.

1.1 Our Contributions

In this paper we propose new constructions of multi-input functional encryption schemes for the inner product functionality that address the aforementioned shortcomings of the state-of-the-art solution of Abdalla et al. [AGRW17].

MIFE for inner products without pairings. Our first contribution consists of (secret-key) MIFE schemes for inner products based on a variety of assumptions, notably *without the need of bilinear maps*, and where *decryption works efficiently*, even for messages of super-polynomial size. We achieve this result by proposing a generic construction of MIFE from any single-input FE (for inner products) in which the encryption algorithm is linearly-homomorphic. Our transformation is surprisingly simple, general and efficient. In particular, it does not require pairings (as in the case of [AGRW17]), and it can be instantiated with *all* known single-input functional encryption schemes (e.g., [ABDP15,ABDP16,ALS16]). This allows us to obtain new MIFE for inner products from plain DDH, composite residuosity, and LWE. Beyond the obvious advantage of enlarging the set of assumptions on which MIFE can be based, this result yields schemes that can be used with a much larger message space. Indeed, dropping the bilinear groups requirement allows us to employ schemes where the decryption time is polynomial, rather than exponential, in the message bit size. From a more theoretical perspective, our results also show that, contrary to what was previously conjectured [AGRW17], MIFE for inner product does not need any (qualitatively) stronger assumption than their single-input counterpart.

Our solution, in more detail. To better describe our solution, let us first explain the basic ideas behind Abdalla *et al.*'s scheme [AGRW17]. Informally, the latter builds upon a clever two-step decryption blueprint. The ciphertexts $\mathsf{ct}_1 = \mathsf{Enc}(\boldsymbol{x}_1), \ldots, \mathsf{ct}_n = \mathsf{Enc}(\boldsymbol{x}_n)$ (corresponding to slots $1, \ldots, n$) are all created using different instances of a single-input FE. Decryption is performed in two stages. One first decrypts each single ct_i separately using the secret key $\mathsf{sk}_{\boldsymbol{y}_i}$ of the underlying single-input FE, and then the outputs of these decryptions are added up to get the final result.

The main technical challenge of this approach is that the stage one of the above decryption algorithm leaks information on each partial inner product $\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle$. To avoid this leakage, their idea is to let source i encrypt its plaintext vector $\boldsymbol{x}_i$ augmented with some fixed (random) value u_i, which is part of the secret key. Moreover, $\mathsf{sk}_{\boldsymbol{y}_i}$ are built by running the single-input FE key generation algorithm on input $\boldsymbol{y}_i || r$, i.e., the vector $\boldsymbol{y}_i$ augmented with fresh randomness r.

By these modifications, and skipping many technical details, stage-one decryption then consists of using pairings to compute, in $\mathbb{G}_T$, the values[8] $[\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + u_i r]_T$ for every slot i. From these quantities, the result $[\langle \boldsymbol{x}, \boldsymbol{y} \rangle]_T$ is obtained as

$$\prod_{i=1}^{n} [\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + u_i r]_T \cdot [-(\sum_{i=1}^{n} u_i) r]_T$$

which can be easily computed if $[-(\sum_{i=1}^{n} u_i) r]_T$ is included in the secret key.

[8] Here we implicitly adopt the, by now standard, bracket notation from [EHK+13].

Intuitively, the scheme is secure as the quantities $[u_i r]_T$ are all pseudo random (under the DDH assumption) and thus hide all the partial information $[\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + u_i r]_T$ may leak. Notice that, in order for this argument to go through, it is crucial that the quantities $[\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + u_i r]_T$ are all encoded in the exponent, and thus decoding is possible only for small norm exponents. Furthermore, this technique seems to inherently require pairings, as both u_i and r have to remain hidden while allowing to compute an encoding of their product at decryption time. This is why the possibility of a scheme without pairings was considered as "quite surprising" in [AGRW17].

We overcome these difficulties via a new FE to MIFE transform, which manages to avoid leakage in a much simpler and efficient way. Our transformation works in two steps. First, we consider a simplified scheme where only one ciphertext query is allowed and messages live in the ring $\mathbb{Z}_L$, for some integer L. In this setting, we build the following multi-input scheme. For each slot i the (master) secret key for slot i consists of one random vector $\boldsymbol{u}_i \in \mathbb{Z}_L^m$. Encrypting $\boldsymbol{x}_i$ merely consists in computing $\boldsymbol{c}_i = \boldsymbol{x}_i + \boldsymbol{u}_i \bmod L$. The secret key for function $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$, is just $z_{\boldsymbol{y}} = \sum_{i=1}^n \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle \bmod L$. To decrypt, one computes

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod L = \langle (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n), \boldsymbol{y} \rangle - z_{\boldsymbol{y}} \bmod L$$

Security comes from the fact that, if only one ciphertext query is allowed, the above can be seen as the functional encryption equivalent of the one-time pad[9].

Next, to guarantee security in the more challenging setting where many ciphertext queries are allowed, we just add a layer of (functional) encryption on top of the above one-time encryption. More specifically, we encrypt each $\boldsymbol{c}_i$ using a FE (supporting inner products) that is both linearly homomorphic and whose message space is compatible with L. So, given ciphertexts $\{\mathsf{ct}_i = \mathsf{Enc}(\boldsymbol{c}_i)\}$ and secret key $\mathsf{sk}_{\boldsymbol{y}} = (\{\mathsf{sk}_{\boldsymbol{y}_i}\}_i, z_{\boldsymbol{y}})$, one can first obtain $\{\langle \boldsymbol{c}_i, \boldsymbol{y}_i \rangle = \mathsf{Dec}(\mathsf{ct}_i, \mathsf{sk}_{\boldsymbol{y}_i})\}$, and then extract the result as $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^n \langle \boldsymbol{c}_i, \boldsymbol{y}_i \rangle - \langle \boldsymbol{u}, \boldsymbol{y} \rangle$.

Our transformation actually comes in two flavors: the first one addresses the case where the underlying FE computes inner products over some finite ring $\mathbb{Z}_L$; the second one instead considers FE schemes that compute bounded-norm inner products over the integers. In both cases the transformations are generic enough to be instantiated with known single-input FE schemes for inner products. This gives us new MIFE relying on plain DDH [ABDP15], LWE [ALS16] and Composite residuosity [ALS16,ABDP16]. Moreover, the proposed transform is security-preserving in the sense that, if the underlying FE achieves adaptive security, so does our resulting MIFE.

Function-Hiding MIFE for inner products. Our second contribution are new MIFE schemes for inner products that achieve function hiding. Our constructions build on the pairing-based solution from [AGRW17] and, as such, they also rely on pairings. More precisely, we propose transformations that, starting from the MIFE from [AGRW17], build function hiding MIFEs using single input FE for inner products as additional building block. Ours transforms are generic with respect to this latter component, in the sense that they can be instantiated using any single input FE satisfying some natural additional requirements (details of which are given in Section 5).

Our methods build from the two-layer encryption technique recently developed by Lin [Lin17] to generically achieve function hiding in the context of (single input) FE for inner products. Intuitively, Lin's idea consists in doing similar operations both at encryption and at key derivation time. Starting from two independent instances of the underlying FE, an "inner" one and an "outer"

[9] We remark that a similar information theoretic construction was put forward by Wee in [Wee17], as a warm-up scheme towards an FE for inner products achieving simulation security.

one, the idea is to encrypt the plaintext $\boldsymbol{x}$ in two steps. One first uses the "inner" FE to compute $\mathsf{ct}_1 = \mathsf{Enc}(\mathsf{msk}_1, \boldsymbol{x})$ and then "extracts" the key corresponding to ct_1, i.e., $\mathsf{ct}_2 = \mathsf{KeyGen}(\mathsf{msk}_2, \mathsf{ct}_1)$. Key derivation is done similarly, one first computes $\mathsf{sk}_1 = \mathsf{KeyGen}(\mathsf{msk}_1, \boldsymbol{y})$ and then encrypts sk_1 using the outer scheme, i.e., $\mathsf{sk}_2 = \mathsf{Enc}(\mathsf{msk}_2, \mathsf{sk}_1)$.

If one encodes ciphertexts in $\mathbb{G}_1$ and secret keys in $\mathbb{G}_2$, then one can use pairings to compute an encoding, in $\mathbb{G}_T$, of $[\langle \mathsf{ct}_2, \mathsf{sk}_2 \rangle]_T$. Since decryption essentially performs inner product, the latter computation actually decrypts also the inner ct_1 component using secret key sk_1, thus yielding an encoding of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$. Moreover, since now $\boldsymbol{y}$ is encrypted, the FE security also provides function hiding[10].

An obvious drawback of Lin's transformation is that, when applied generically, it would induce an extra-level of multilinearity in the process. This means that, starting from a pairing-free FE for inner products, one ends up with a scheme that is function hiding but also pairing-based.

We propose similar two-layer encryption techniques that *do not*, inherently, induce extra levels of multi-linearity with respect to those of the underlying primitives. Our transforms achieve this by using the MIFE from [AGRW17] as inner scheme and, several instances of, a single input FE, one for each encryption slot, as outer schemes. In particular, by carefully exploiting the specific algebraic properties of the MIFE, we manage to achieve function hiding from the Matrix Decisional Diffie Hellman assumption over *standard* bilinear groups (i.e., without resorting to multi-linear maps). Specifically, our schemes come in two flavors: a simpler one for selective security and a more convoluted one achieving adaptive security. A high level overview of our technique appears in Section 5. The MIFE schemes from Lin [Lin17] are selectively secure and function-hiding, but are based on multi-linear maps ($d-1$ slots require a multilinear map of degree d). In comparison, our schemes support a polynomial number of inputs and achieve adaptive-security, while using only pairings and while being based only on standard assumptions.

Generality of our approach. As mentioned above, our function-hiding transforms are not entirely generic as they impose restrictions on the underlying MIFE. These restrictions, while compatible with the pairing-based realization from [AGRW17], do not cope well with our newly constructed MIFEs without pairings. Very informally, this is due to the fact that our transform relies on the two-step decryption blueprint in which one learns $[\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + z_i]$, and each z_i is "sufficiently" random to guarantee security in the MIFE security experiment. Specifically, in Abdalla et al.'s scheme $z_i = u_i r$ whereas in our new scheme $z_i = \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle$. While the latter value is sufficiently random in the MIFE indistinguishability experiment, this is no longer the case in the function-hiding experiment, where the adversary asks for pairs of keys $(\boldsymbol{y}^0, \boldsymbol{y}^1)$, and $z_i = \langle \boldsymbol{u}_i, \boldsymbol{y}_i^\beta \rangle$ may actually leak information about which of the two keys was chosen (i.e. information about the value of the bit β). With a different interpretation, if one sees $[\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle + z_i]$ as a secret sharing of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, then in our new scheme this secret sharing depends on the function $\boldsymbol{y}$ whereas in [AGRW17] this is function independent and more suitable for function-hiding. We believe that coming up with more powerful transforms, capable of exploiting the potential of our efficient MIFEs, is a very natural and interesting open problem.

Concurrent work on function-hiding. Concurrently and independently of our work, Datta et al. [DOT18] proposed a multi-input function-hiding scheme for inner products. Their construction uses the framework of dual pairing vector spaces and require the use of pairings. They achieve

[10] Actually the transform sketched here only manages to guarantee a weaker form of function hiding. However this can be generically turned into standard function hiding [LV16], as described in Appendix B.

slightly shorter ciphertexts and decryption keys (ciphertexts are shorter by 2 group elements, while decryption keys require 2n+1 less group elements). However, this comes at the expense of a larger master secret key, which contains $4n(m^2-1)$ more group elements (a quadratic blow-up in m).

Interestingly, Datta et al. [DOT18] also provide a technique based on pseudorandom functions to extend their multi-input function-hiding scheme to an unbounded number of slots. Although their techniques also appear to be applicable to our schemes, hence capable of extending both the pairing-free and the pairing-based constructions to the unbounded setting, we leave it as future work.

2 Preliminaries

Notation. We denote with $\lambda \in \mathbb{N}$ a security parameter. A *probabilistic polynomial time* (PPT) algorithm $\mathcal{A}$ is a randomized algorithm for which there exists a polynomial $p(\cdot)$ such that for every input x the running time of $\mathcal{A}(x)$ is bounded by $p(|x|)$. We say that a function $\varepsilon : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every positive polynomial $p(\lambda)$ there exists $\lambda_0 \in \mathbb{N}$ such that for all $\lambda > \lambda_0$: $\varepsilon(\lambda) < 1/p(\lambda)$. If S is a set, $x \leftarrow_{\mathrm{R}} S$ denotes the process of selecting x uniformly at random in S. If $\mathcal{A}$ is a probabilistic algorithm, $y \leftarrow_{\mathrm{R}} \mathcal{A}(\cdot)$ denotes the process of running $\mathcal{A}$ on some appropriate input and assigning its output to y. For a positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. We denote vectors $\boldsymbol{x} = (x_i)$ and matrices $\mathbf{A} = (a_{i,j})$ in bold. For a set S (resp. vector $\boldsymbol{x}$) $|S|$ (resp. $|\boldsymbol{x}|$) denotes its cardinality (resp. number of entries). Also, given two vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ we denote by $\boldsymbol{x} \| \boldsymbol{x}'$ their concatenation. By $\equiv$, we denote the equality of statistical distributions, and for any $\varepsilon > 0$, we denote by $\approx_\varepsilon$ the ε-statistical difference of two distributions.

2.1 Definitions for Multi-Input Functional Encryption

In this section we recall the definitions of multi-input functional encryption [GGG+14] specialized to the private-key setting, as this is the one relevant for our constructions.

Definition 1 (Multi-input Function Encryption). *Let $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ be an ensemble where each $\mathcal{F}_n$ is a family of n-ary functions. A function $f \in \mathcal{F}_n$ is defined as follows $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}$. A multi-input functional encryption scheme $\mathcal{MIFE}$ for $\mathcal{F}$ consists of the following algorithms:*

- $\mathsf{Setup}(1^\lambda, \mathcal{F}_n)$ *takes as input the security parameter λ and a description of $\mathcal{F}_n \in \mathcal{F}$, and outputs a master public key* mpk[11] *and a master secret key* msk*. The master public key* mpk *is assumed to be part of the input of all the remaining algorithms.*
- $\mathsf{Enc}(\mathsf{msk}, i, x_i)$ *takes as input the master secret key* msk*, an index $i \in [n]$, and a message $x_i \in \mathcal{X}_i$, and it outputs a ciphertext* ct*. Each ciphertext is assumed to be associated with an index i denoting for which slot this ciphertext can be used for. When $n = 1$, the input i is omitted.*
- $\mathsf{KeyGen}(\mathsf{msk}, f)$ *takes as input the master secret key* msk *and a function $f \in \mathcal{F}_n$, and it outputs a decryption key* sk_f*.*
- $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ *takes as input a decryption key sk_f for function f and n ciphertexts, and it outputs a value $y \in \mathcal{Y}$.*

A scheme $\mathcal{MIFE}$ as defined above is correct if for all $n \in \mathbb{N}$, $f \in \mathcal{F}_n$ and all $x_i \in \mathcal{X}_i$ for $1 \le i \le n$, we have

$$\Pr\left[\begin{array}{c}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n); \quad \mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f); \\ \mathsf{Dec}(\mathsf{sk}_f, \mathsf{Enc}(\mathsf{msk}, 1, x_1), \ldots, \mathsf{Enc}(\mathsf{msk}, n, x_n)) = f(x_1, \ldots, x_n)\end{array}\right] = 1,$$

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

[11] In the private key setting, we think of mpk as some public parameters common to all algorithms.

Security notions. Here we recall the definitions of security for multi-input functional encryption. We give both one-time and many-time indistinguishability-based security definitions. Namely, we consider several security notions denoted xx-AD-IND and xx-SEL-IND, where: xx $\in \{\text{one}, \text{many}\}$. We also give simulation-based security definitions in Appendix A.

Definition 2 (xx-AD-IND-secure MIFE). *For every multi-input functional encryption $\mathcal{MIFE}$ for $\mathcal{F}$, every stateful adversary $\mathcal{A}$, every security parameter $\lambda \in \mathbb{N}$, and every xx $\in$ {one,many}, we define the following experiments for $\beta \in \{0,1\}$:*

$$\begin{array}{|l|}\hline \underline{\textit{Experiment}}\ \textbf{xx-AD-IND}_{\beta}^{\mathcal{MIFE}}(1^{\lambda}, \mathcal{A}): \\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda}, \mathcal{F}_n) \\ \alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot),\mathsf{Enc}(\cdot,\cdot,\cdot)}(\mathsf{mpk}) \\ \textbf{\textit{Output:}}\ \alpha \\ \hline \end{array}$$

where Enc *is an oracle that on input (i, x_i^0, x_i^1) outputs $\mathsf{Enc}(\mathsf{msk}, i, x_i^{\beta})$. Also, $\mathcal{A}$ is restricted to only make queries f to $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ satisfying*

$$f(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f(x_1^{j_1,1}, \ldots, x_n^{j_n,1})$$

for all $j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n]$, where for all $i \in [n]$, Q_i denotes the number of encryption queries for input slot i. We denote by Q_f the number of key queries. Note that w.l.o.g. (as shown in [AGRW17, Lemma 3]), we can assume that for all $i \in [n]$, $Q_i > 0$. When xx = one, we also require that $\mathcal{A}$ queries $\mathsf{Enc}(i, \cdot, \cdot)$ once per slot, namely that $Q_i = 1$, for all $i \in [n]$.

A private-key multi-input functional encryption $\mathcal{MIFE}$ for $\mathcal{F}$ is xx-AD-IND-secure if every PPT adversary $\mathcal{A}$ has advantage negligible in λ, where the advantage is defined as:

$$\begin{aligned} &\mathsf{Adv}_{\mathcal{MIFE}}^{\text{xx-AD-IND}}(\lambda, \mathcal{A}) = \\ &\left| \Pr\left[\textbf{xx-AD-IND}_0^{\mathcal{MIFE}}(1^{\lambda}, \mathcal{A}) = 1\right] - \Pr\left[\textbf{xx-AD-IND}_1^{\mathcal{MIFE}}(1^{\lambda}, \mathcal{A}) = 1\right] \right| \end{aligned}$$

Remark 1 (winning condition). The winning condition may not always efficiently checkable because of the combinatorial explosion in the restrictions on the queries.

Definition 3 (xx-SEL-IND-secure MIFE). *For every multi-input functional encryption $\mathcal{MIFE}$ for $\mathcal{F}$, every stateful adversary $\mathcal{A}$, every security parameter $\lambda \in \mathbb{N}$, and every xx $\in$ {one,many}, we define the following experiments for $\beta \in \{0,1\}$:*

$$\begin{array}{|l|}\hline \underline{\textit{Experiment}}\ \textbf{xx-SEL-IND}_{\beta}^{\mathcal{MIFE}}(1^{\lambda}, \mathcal{A}): \\ \{x_i^{j,b}\}_{i\in[n], j\in[Q_i], b\in\{0,1\}} \leftarrow \mathcal{A}(1^{\lambda}, \mathcal{F}_n) \\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^{\lambda}, \mathcal{F}_n) \\ \mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{msk}, x_i^{j,\beta}) \\ \alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, \{\mathsf{ct}_i^j\}_{i\in[n], j\in[Q_i]}\right) \\ \textbf{\textit{Output:}}\ \alpha \\ \hline \end{array}$$

where $\mathcal{A}$ is restricted to only make queries f to $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ satisfying

$$f(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f(x_1^{j_1,1}, \ldots, x_n^{j_n,1})$$

for all $j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n]$. *When* $xx = one$, *we also require that* $Q_i = 1$, *for all* $i \in [n]$.

A $\mathcal{MIFE}$ *for* $\mathcal{F}$ *is xx-SEL-IND-secure if every PPT adversary* $\mathcal{A}$ *has negligible advantage in* λ, *where the advantage is defined as:*

$$\mathsf{Adv}^{\text{xx-SEL-IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = \\ \left| \Pr\left[\textbf{xx-SEL-IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\textbf{xx-SEL-IND}_1^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\right] \right|.$$

Zero vs multiple queries in the private-key setting. A nice feature enjoyed by all the schemes in Section 3 is that the owner of a decryption key $\mathsf{sk}_{\boldsymbol{y}}$ associated with the vector $\boldsymbol{y} = \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n$ does not need to know a specific value ct_i of the ciphertext vector $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ in order to decrypt ct if $\boldsymbol{y}_i = \mathbf{0}$. In other words, Q_i can be 0 whenever $\boldsymbol{y}_i = \mathbf{0}$. In this case, the adversary is only allowed to obtain a secret key $\mathsf{sk}_{\boldsymbol{y}}$ for a vector $\boldsymbol{y}$ satisfying the condition

$$\sum_{i \in I} \langle \boldsymbol{x}_i^{j,0}, \boldsymbol{y}_i \rangle = \sum_{i \in I} \langle \boldsymbol{x}_i^{j,1}, \boldsymbol{y}_i \rangle,$$

for all queries $j \in [Q_i]$, where $I \subseteq [n]$ denotes the set of slots for which the adversary made at least one query to Enc, that is, for which $Q_i > 0$. Though we believe this feature can be useful in practice (for instance, if one of the encrypting parties decides to stop collaborating), certain applications may require at least one ciphertext for each encryption slot in order for decryption to be possible. In such cases, one can apply to our schemes the simple generic compiler given in [AGRW17, Lemma 3] to ensure that the set $I = [n]$, thus obtaining new schemes which leak no information in the setting where some $Q_i = 0$. For this reason, we assume without loss of generality that $Q_i > 0$ in all our security definitions and proofs.

2.2 Function-Hiding Multi-Input Functional Encryption

For function-hiding, we focus on indistinguishability security notions. This is because even single-input function-hiding inner-product encryption is known to be unrealizable in a simulation sense under standard assumptions.

Definition 4 (xx-SEL-Function-hiding MIFE). *For every multi-input functional encryption* $\mathcal{MIFE}$ *for* $\mathcal{F}$, *every security parameter* λ, *every stateful adversary* $\mathcal{A}$, *and every* $xx \in \{one, many\}$, *we define the following experiments for* $\beta \in \{0, 1\}$*:*

Experiment $\textbf{xx-SEL-FH-IND}_\beta^{\mathcal{MIFE}}(1^\lambda, \mathcal{A})$*:*

$\{x_i^{j,b}\}_{i \in [n], j \in [Q_i], b \in \{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$\{f^{j,b}\}_{j \in [Q_f], b \in \{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n)$
$\mathsf{ct}_i^j \leftarrow \mathsf{Enc}(\mathsf{msk}, i, x_i^{j,\beta})\ \forall i \in [n], j \in [Q_i]$
$\mathsf{sk}^j \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f^{j,\beta})\ \forall j \in [Q_f]$
$\alpha \leftarrow \mathcal{A}\left(\mathsf{mpk}, (\mathsf{ct}_i^j)_{i \in [n], j \in [Q_i]}, (\mathsf{sk}^j)_{j \in [Q_f]}\right)$
Output: α

where $\mathcal{A}$ only makes Q_i selective queries of plaintext pairs $(x_i^{j_i,0}, x_i^{j_i,1})$ and Q_f selective queries of key pairs $(f^{j_f,0}, f^{j_f,1})$, that must satisfy:

$$f^{j_f,0}(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f^{j_f,1}(x_1^{j_1,1}, \ldots, x_n^{j_n,1})$$

for all $j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n]$ and for all $j_f \in [Q_f]$.

A $\mathcal{MIFE}$ is xx-SEL-FH-IND-secure if every PPT adversary $\mathcal{A}$ has negligible advantage in λ, where the advantage is defined as:

$$\mathsf{Adv}^{xx\text{-}SEL\text{-}FH\text{-}IND}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = \big|\Pr\big[\textbf{xx-SEL-FH-IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\big] - \Pr\big[\textbf{xx-SEL-FH-IND}_1^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\big]\big|$$

Definition 5 (xx-AD-Function-hiding MIFE). *For every multi-input functional encryption $\mathcal{MIFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for $\mathcal{F}$, every security parameter λ, every stateful adversary $\mathcal{A}$, and every $xx \in \{one,many\}$, we define the following experiments for $\beta \in \{0,1\}$:*

Experiment $\textbf{xx-AD-FH-IND}_\beta^{\mathcal{MIFE}}(1^\lambda, \mathcal{A})$*:*

$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n)$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot,\cdot),\mathsf{Enc}(\mathsf{msk},\cdot,\cdot)}(\mathsf{mpk})$
Output: α

where Enc *is an oracle that on input (i, x_i^0, x_i^1) outputs $\mathsf{Enc}(\mathsf{msk}, i, x_i^\beta)$ and* KeyGen *is an oracle that on input (f^0, f^1) outputs $\mathsf{KeyGen}(\mathsf{msk}, f^\beta)$. Additionally, $\mathcal{A}$ queries must satisfy:*

$$f^{j_f,0}(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f^{j_f,1}(x_1^{j_1,1}, \ldots, x_n^{j_n,1})$$

for all $j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n]$ and for all $j_f \in [Q_f]$.

A $\mathcal{MIFE}$ is xx-AD-FH-IND-secure if every PPT adversary has negligible advantage in λ, where the advantage is defined as:

$$\mathsf{Adv}^{xx\text{-}AD\text{-}FH\text{-}IND}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = \Big|\Pr\big[\textbf{xx-AD-FH-IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\big] - \Pr\big[\textbf{xx-AD-FH-IND}_1^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\big]\big|$$

Definition 6 (Weak function hiding MIFE). *Following the approach from [LV16], we define the notion of weak function hiding (denoted xx-yy-wFH-IND) in the multi-input case, which is as in Definitions 4 and 5, with the exception that the previous constraints on ciphertext and key challenges:*

$$f^{j_f,0}(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f^{j_f,1}(x_1^{j_1,1}, \ldots, x_n^{j_n,1}), \quad \text{for all } j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n] \text{ and for all } j_f \in [Q_f]$$

are extended with additional constraints to help with our hybrid proof:

$$f^{j_f,0}(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f^{j_f,0}(x_1^{j_1,1}, \ldots, x_n^{j_n,1}) = f^{j_f,1}(x_1^{j_1,1}, \ldots, x_n^{j_n,1}), \quad \text{for all } j_1, \ldots, j_n \in [Q_1] \times \cdots \times [Q_n] \text{ and for all } j_f \in [Q_f].$$

2.3 Inner-product functionality

In this paper we construct multi-input functional encryption schemes that support the following two variants of the multi-input inner product functionality:

Multi-Input Inner Product over $\mathbb{Z}_L$. This is a family of functions that is defined as $\mathcal{F}^m_{L,n} = \{f_{\boldsymbol{y}_1,\ldots,\boldsymbol{y}_n} : (\mathbb{Z}^m_L)^n \to \mathbb{Z}_L, \text{ for } \boldsymbol{y}_i \in \mathbb{Z}^m_L\}$ where

$$f_{\boldsymbol{y}_1,\ldots,\boldsymbol{y}_n}(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n) = \sum_{i=1}^{n} \langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle \bmod L.$$

Multi-Input Bounded-Norm Inner Product over $\mathbb{Z}$. This is defined as $\mathcal{F}^{m,X,Y}_n = \{f_{\boldsymbol{y}_1,\ldots,\boldsymbol{y}_n} : (\mathbb{Z}^m)^n \to \mathbb{Z}\}$ where $f_{\boldsymbol{y}_1,\ldots,\boldsymbol{y}_n}(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)$ is the same as above except that the result is not reduced mod L, and vectors are required to satisfy the following bounds: $\|\boldsymbol{x}\|_\infty < X$, $\|\boldsymbol{y}\|_\infty < Y$.

2.4 Computational assumptions

Prime-order groups. Let GGen be a probabilistic polynomial time (PPT) algorithm that on input 1^λ returns a description $\mathcal{G} = (\mathbb{G}, p, g)$ of an cyclic group $\mathcal{G}$ of order p for a 2λ-bit prime p, whose generator is g.

We use implicit representation of group elements as introduced in [EHK+13]. For $a \in \mathbb{Z}_p$, define $[a] = g^a \in \mathbb{G}$ as the *implicit representation* of a in $\mathcal{G}$. More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}^{n\times m}_p$ we define $[\mathbf{A}]$ as the implicit representation of $\mathbf{A}$ in $\mathcal{G}$:

$$[\mathbf{A}] := \begin{pmatrix} g^{a_{11}} & \ldots & g^{a_{1m}} \\ & & \\ g^{a_{n1}} & \ldots & g^{a_{nm}} \end{pmatrix} \in \mathbb{G}^{n\times m}$$

We will always use this implicit notation of elements in $\mathbb{G}$, i.e., we let $[a] \in \mathbb{G}$ be an element in $\mathbb{G}$. Note that from a random $[a] \in \mathbb{G}$ it is generally hard to compute the value a (discrete logarithm problem in $\mathbb{G}$). Obviously, given $[a], [b] \in \mathbb{G}$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax] \in \mathbb{G}$ and $[a + b] \in \mathbb{G}$.

Matrix Diffie-Hellman Assumption for prime-order groups. We recall the definition of the Matrix Decision Diffie-Hellman (MDDH) Assumption [EHK+13].

Definition 7 (Matrix Distribution). *Let $k \in \mathbb{N}$. We call $\mathcal{D}_k$ a matrix distribution if it outputs matrices in $\mathbb{Z}^{(k+1)\times k}_p$ of full rank k in polynomial time.*

W.l.o.g. we assume the first k rows of $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k$ form an invertible matrix. The $\mathcal{D}_k$-Matrix Diffie-Hellman problem is to distinguish the two distributions $([\mathbf{A}], [\mathbf{A}\boldsymbol{w}])$ and $([\mathbf{A}], [\boldsymbol{u}])$ where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\boldsymbol{w} \leftarrow_{\mathrm{R}} \mathbb{Z}^k_p$ and $\boldsymbol{u} \leftarrow_{\mathrm{R}} \mathbb{Z}^\ell_p$.

Definition 8 ($\mathcal{D}_k$-Matrix Diffie-Hellman ($\mathcal{D}_k$-MDDH) assumption in prime-order groups). *Let $\mathcal{D}_k$ be a matrix distribution. The $\mathcal{D}_k$-Matrix Diffie-Hellman ($\mathcal{D}_k$-MDDH) assumption holds relative to GGen if for all PPT adversaries $\mathcal{A}$,*

$$\mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathsf{GGen},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\mathbf{A}\boldsymbol{w}]) = 1] - \Pr[\mathcal{A}(\mathcal{G}, [\mathbf{A}], [\boldsymbol{u}]) = 1]| = \mathsf{negl}(\lambda),$$

where probabilities are over $\mathcal{G} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \boldsymbol{w} \leftarrow_{\mathrm{R}} \mathbb{Z}^k_p, \boldsymbol{u} \leftarrow_{\mathrm{R}} \mathbb{Z}^{k+1}_p$.

Pairing groups. Let PGGen be a probabilistic polynomial time (PPT) algorithm that on input 1^λ returns a description $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, q, g_1, g_2)$ of asymmetric pairing groups where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic group of order p for a 2λ-bit prime p, g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable (non-degenerate) bilinear map. Define $g_T := e(g_1, g_2)$, which is a generator of $\mathcal{G}_T$. We again use implicit representation of group elements. For $s \in 1, 2, T$ and $a \in \mathbb{Z}_p$, define $[a]_s = g_s^a \in \mathcal{G}_s$ as the implicit representation of a in G_s . Given $[a]_1$, $[a]_2$, one can efficiently compute $[ab]_T$ using the pairing e. For two matrices $\mathbf{A}$, $\mathbf{B}$ with matching dimensions define $e([\mathbf{A}]_1, [\mathbf{B}]_2) := [\mathbf{AB}]_T \in \mathbb{G}_T$.

We define the $\mathcal{D}_k$-MDDH assumption in pairing groups similarly than in prime-order groups (see Definition 8).

Definition 9 ($\mathcal{D}_k$-MDDH assumption in pairing groups). *Let $\mathcal{D}_k$ be a matrix distribution. The $\mathcal{D}_k$-MDDH assumption holds relative to PGGen in $\mathbb{G}_s$, for $s \in \{1, 2, T\}$, if for all PPT adversaries $\mathcal{A}$, the following is $\mathsf{negl}(\lambda)$:*

$$\mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathcal{PG}, [\mathbf{A}]_s, [\mathbf{A}\boldsymbol{w}]_s) = 1] - \Pr[\mathcal{A}(\mathcal{PG}, [\mathbf{A}]_s, [\boldsymbol{u}]_s) = 1]|$$

where probabilities are over $\mathcal{PG} \leftarrow_{\mathrm{R}} \mathsf{PGGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \boldsymbol{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k, \boldsymbol{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$.

Next, we recall a result on the uniform distribution over full-rank matrices:

Definition 10 (Uniform distribution). *Let $\ell, k \in \mathbb{N}$, with $\ell > k$. We denote by $\mathcal{U}_{\ell,k}$ the uniform distribution over all full-rank $\ell \times k$ matrices over $\mathbb{Z}_p$.*

Among all possible matrix distributions $\mathcal{D}_k$, the uniform matrix distribution $\mathcal{U}_{\ell,k}$ is the hardest possible instance, so in particular k-Lin $\Rightarrow \mathcal{U}_k$-MDDH, as stated in Lemma 1.

Lemma 1 ($\mathcal{D}_k$-MDDH $\Rightarrow \mathcal{U}_{\ell,k}$-MDDH, [EHK+13]). *Let $\ell, k \in \mathbb{N}$ and $\mathcal{D}_k$ a matrix distribution. For any PPT adversary $\mathcal{A}$, there exists a PPT $\mathcal{B}$ such that*

$$\mathbf{Adv}^{\mathcal{U}_{\ell,k}\text{-}\mathsf{mddh}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_s,\mathcal{B}}(\lambda).$$

3 From Single to Multi-Input FE for Inner Product

In this section, we give a generic construction of MIFE for inner product from any single-input FE $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the same functionality. More precisely, we show two transformations: the first one addresses FE schemes that compute the inner product functionality over a finite ring $\mathbb{Z}_L$ for some integer L, while the second transformation addresses FE schemes for bounded-norm inner product. The two transformations are almost the same, and the only difference is that in the case of bounded-norm inner product, we require additional structural properties on the single-input FE. Yet we stress that these properties are satisfied by all existing constructions. Both our constructions rely on a simple MIFE scheme that is one-AD-IND secure unconditionally. In particular, our constructions show how to use single-input FE in order to bootstrap the information-theoretic MIFE from one-time to many-time security.

3.1 Information-Theoretic MIFE with One-Time Security

Here we present the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$ for the class $\mathcal{F}^m_{L,n}$, and we prove its one-AD-IND security. The scheme is described in Figure 1.

Theorem 1. *The MIFE described in Figure 1 is one-AD-IND secure. Namely, for any adversary $\mathcal{A}$, $\mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = 0$.*

$\underline{\mathsf{Setup}^{\mathsf{ot}}(1^\lambda, \mathcal{F}^m_{L,n})}$:	$\underline{\mathsf{KeyGen}^{\mathsf{ot}}(\boldsymbol{u}, \boldsymbol{y}_1 \Vert \cdots \Vert \boldsymbol{y}_n)}$:
For all $i \in [n]$, $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}^m_L$	Return $z := \sum_{i\in[n]} \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle \bmod L$
Return $\boldsymbol{u} = \{\boldsymbol{u}_i\}_{i\in[n]}$	
	$\underline{\mathsf{Dec}^{\mathsf{ot}}(z, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)}$:
$\underline{\mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i)}$:	Return $\sum_{i=1}^n \langle \mathsf{ct}_i, \boldsymbol{y}_i \rangle - z \bmod L$
Return $\boldsymbol{x}_i + \boldsymbol{u}_i \bmod L$	

Fig. 1. Private-key, information theoretically secure, multi-input FE scheme $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ for the class $\mathcal{F}^m_{L,n}$.

Proof overview. The proof of Theorem 1 has two main steps. First, we use the fact that any adaptive distinguisher against $\mathcal{MIFE}^{\mathsf{ot}}$ with advantage ε can be transformed into a selective distinguisher with advantage $\varepsilon/|X|^2$ by randomly guessing the two challenge input vectors, where $|X|$ is the size of the input space ($|X| = L^{nm}$ in our case). Then, in a second step, we show that any selective distinguisher against $\mathcal{MIFE}^{\mathsf{ot}}$ has advantage 0 since $\mathcal{MIFE}^{\mathsf{ot}}$ behaves as the FE equivalent of the one-time pad. Hence, it follows that any adaptive distinguisher must also have advantage 0.

Proof. Let $\mathcal{A}$ be an adversary against the one-AD-IND security of the MIFE. First, we use a complexity leveraging argument to build an adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq L^{-2nm} \cdot \mathsf{Adv}^{\textit{one-SEL-IND}}_{\mathcal{MIFE},\mathcal{B}}(\lambda).$$

The adversary $\mathcal{B}$ simply guesses the challenge $\{\boldsymbol{x}_i^b\}_{i\in[n],b\in\{0,1\}}$ in advance, then simulates $\mathcal{A}$'s experiment using its own selective experiment. When $\mathcal{B}$ receives $\mathcal{A}$'s challenge, it checks if the guess was successful (call E that event): if it was, it continues simulating $\mathcal{A}$'s experiment, otherwise, it returns 0. When the guess is successful, $\mathcal{B}$ perfectly simulate $\mathcal{A}$'s view. Since event E happens with probability exactly L^{-2nm}, and is independent of the adversary $\mathcal{A}$'s view, we obtain $\mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq L^{-2nm} \cdot \mathsf{Adv}^{\textit{one-SEL-IND}}_{\mathcal{MIFE},\mathcal{B}}(\lambda)$.

It remains to prove that the MIFE presented in Figure 1 satisfies perfect one-SEL-IND security, namely, for any adversary $\mathcal{B}$, $\mathsf{Adv}^{\textit{one-SEL-IND}}_{\mathcal{MIFE},\mathcal{B}}(\lambda) = 0$. To do so, we introduce hybrid games $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ described in Figure 2. We prove that for all $\beta \in \{0,1\}$, $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ is identical to the experiment one-SEL-IND$^{\mathcal{MIFE}}_\beta(1^\lambda, \mathcal{B})$. This can be seen using the fact that for all $\{\boldsymbol{x}_i^\beta \in \mathbb{Z}^m\}_{i\in[n]}$, the following distributions are identical: $\{\boldsymbol{u}_i \bmod L\}_{i\in[n]}$ and $\{\boldsymbol{u}_i - \boldsymbol{x}_i^\beta \bmod L\}_{i\in[n]}$, with $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}^m_L$. Recall that here $i \in [n]$ is an index for input slots. Note that the independence of the $\boldsymbol{x}_i^\beta$ from the $\boldsymbol{u}_i$ is only true in the selective security game. Finally, we show that $\mathcal{B}$'s view in $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ is independent of β. Indeed, the only information about β that leaks in this experiment is $\sum_i \langle \boldsymbol{x}_i^\beta, \boldsymbol{y}_i \rangle$, which is independent of β by definition of the security game. □

$\underline{\mathbf{HYB}_\beta(1^\lambda, \mathcal{B})}$:	$\underline{\mathcal{O}_H(\boldsymbol{y})}$:
$\{\boldsymbol{x}_i^b\}_{i\in[n],b\in\{0,1\}} \leftarrow \mathcal{B}(1^\lambda, \mathcal{F}^m_{L,n})$	For all $i \in [n]$,
For all $i \in [n]$,	$\quad z := \sum_{i\in[n]} \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle - \langle \boldsymbol{x}_i^\beta, \boldsymbol{y}_i \rangle \bmod L$
$\quad \boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}^m_L$; $\mathsf{ct}_i \leftarrow \boldsymbol{u}_i$	Return z
$\alpha \leftarrow \mathcal{B}^{\mathcal{O}_H(\cdot)}(\{\mathsf{ct}_i\}_{i\in[\ell]})$	
Output α	

Fig. 2. Experiments for the proof of Theorem 1.

Remark 2 (one-SEL-SIM security). As a result of independent interest, in Appendix A we show that the MIFE presented in Figure 1 satisfies perfect one-SEL-SIM security, which implies perfect one-SEL-IND (which itself implies perfect one-AD-IND security via complexity leveraging, as shown in the proof above).

Remark 3 (Linear homomorphism). We use the fact that $\mathsf{Enc}^{\mathsf{ot}}$ is linearly homomorphic, that is, for all input slots $i \in [n]$, $\boldsymbol{x}_i, \boldsymbol{x}'_i \in \mathbb{Z}_p^m$, $\boldsymbol{u} \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, \mathcal{F}_{L,n}^m)$, $\mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i) + \boldsymbol{x}'_i \bmod L = \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i + \boldsymbol{x}'_i)$. This property will be used when using the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ from Figure 1 as a building block to obtain a full-fledged many-AD-IND MIFE.

3.2 Our Transformation for Inner Product over $\mathbb{Z}_L$

We present our multi-input scheme $\mathcal{MIFE}$ for the class $\mathcal{F}_{L,n}^m$ in Figure 3. The construction relies on the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ of Figure 1, and any single-input FE for the class $\mathcal{F}_{L,1}^m$.

$\underline{\mathsf{Setup}'(1^\lambda, \mathcal{F}_{L,n}^m)}$:
$\boldsymbol{u} \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, \mathcal{F}_{L,n}^m)$, for all $i \in [n]$, $(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_{L,1}^m)$
$(\mathsf{mpk}, \mathsf{msk}) := \big(\{\mathsf{mpk}_i\}_{i\in[n]}, (\{\mathsf{msk}_i,\}_{i\in[n]}, \boldsymbol{u})\big)$
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i)}$:
$\boldsymbol{w}_i := \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i)$
Return $\mathsf{Enc}(\mathsf{msk}_i, \boldsymbol{w}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n)}$:
For all $i \in [n]$, $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}_i, \boldsymbol{y}_i)$, $z := \mathsf{KeyGen}^{\mathsf{ot}}(\boldsymbol{u}, \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n)$
$\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \big(\{\mathsf{sk}_i\}_{i\in[n]}, z\big)$
Return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n}$

$\underline{\mathsf{Dec}'\big((\{\mathsf{sk}_i\}_{i\in[n]}, z), \mathsf{ct}_1, \ldots, \mathsf{ct}_n\big)}$:
For all $i \in [n]$, $D_i \leftarrow \mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct}_i)$
Return $\sum_{i\in[n]} D_i - z \bmod L$

Fig. 3. Private-key multi-input FE scheme $\mathcal{MIFE} := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the class $\mathcal{F}_{L,n}^m$ from a public-key single-input FE $\mathcal{FE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the class $\mathcal{F}_{L,1}^m$, and one-time multi-input FE $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ for the class $\mathcal{F}_{L,n}^m$.

The correctness of $\mathcal{MIFE}$ follows from the correctness properties of the single-input scheme $\mathcal{FE}$ and the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$. Indeed, correctness of the former implies that, for all input slots $i \in [n]$, $D_i = \langle \boldsymbol{w}_i, \boldsymbol{y}_i \rangle \bmod L$, while correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ implies that $\sum_{i\in[n]} D_i - z = \mathsf{Dec}^{\mathsf{ot}}(z, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n) = \sum_{i\in[n]} \langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle \bmod L$.

For the security we state the following theorem:

Theorem 2. *If the single-input FE, $\mathcal{FE}$ is many-AD-IND-secure, and the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND-secure, then the multi-input FE, $\mathcal{MIFE}$, described in Figure 3, is many-AD-IND-secure.*

Since the proof of the above theorem is almost the same as the one for the case of bounded-norm inner product, we only provide an overview here, and defer to the proof of Theorem 3 for further details.

Proof overview. Here, for any intput slot $i \in [n]$, we denote by $(\boldsymbol{x}_i^{j,0}, \boldsymbol{x}_i^{j,1})$ the j'th query to $\mathsf{Enc}(i,\cdot,\cdot)$, for any $j \in [Q_i]$, where Q_i is the total number of queries to $\mathsf{Enc}(i,\cdot,\cdot)$.

The proof is in two main steps. First, we switch encryptions of $\boldsymbol{x}_1^{1,0}, \ldots, \boldsymbol{x}_n^{1,0}$ to those of $\boldsymbol{x}_1^{1,1}, \ldots, \boldsymbol{x}_n^{1,1}$, using the one-AD-IND security of $\mathcal{MIFE}^{\mathsf{ot}}$. For the remaining ciphertexts, we switch from an encryption of $\boldsymbol{x}_i^{j,0} = (\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0}) + \boldsymbol{x}_i^{1,0}$ to that of $(\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0}) + \boldsymbol{x}_i^{1,1}$. In this step we use the fact that one can compute an encryption of $\mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, (\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0}) + \boldsymbol{x}_i^{1,0})$ from an encryption $\mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i^{1,0})$, because the encryption algorithm $\mathsf{Enc}^{\mathsf{ot}}$ of $\mathcal{MIFE}^{\mathsf{ot}}$ is linearly homomorphic (see Remark 3). Finally, we apply a hybrid argument across the slots to switch from encryptions of

$$(\boldsymbol{x}_i^{2,0} - \boldsymbol{x}_i^{1,0}) + \boldsymbol{x}_i^{1,1}, \ldots, (\boldsymbol{x}_i^{Q_i,0} - \boldsymbol{x}_i^{1,0}) + \boldsymbol{x}_i^{1,1}$$

to those of

$$(\boldsymbol{x}_i^{2,1} - \boldsymbol{x}_i^{1,1}) + \boldsymbol{x}_i^{1,1}, \ldots, (\boldsymbol{x}_i^{Q_i,1} - \boldsymbol{x}_i^{1,1}) + \boldsymbol{x}_i^{1,1},$$

using the many-AD-IND security of $\mathcal{FE}$.

Instantiations. The construction in Figure 3 can be instantiated using the single-input FE schemes of Agrawal, Libert, and Stehlé [ALS16] that are many-AD-IND-secure and allow for computing inner products over a finite ring. Specifically, we obtain:

- A MIFE for inner product over $\mathbb{Z}_p$ for a prime p, based on the LWE assumption. This is obtained by using the LWE-based scheme of Agrawal et al. [ALS16, Section 4.2].
- A MIFE for inner product over $\mathbb{Z}_N$ where N is an RSA modulus, based on the Composite Residuosity assumption. This is obtained by using the Paillier-based scheme of Agrawal et al. [ALS16, Section 5.2].

We note that since both these schemes in [ALS16] have a stateful key generation, our MIFE inherits this stateful property. Stateless MIFE instantiations are obtained from the transformation in the next section.

3.3 Our Transformation for Inner Product over $\mathbb{Z}$

Here we present our transformation for the case of bounded-norm inner product. In particular, in Figure 4 we present a multi-input scheme $\mathcal{MIFE}$ for the class $\mathcal{F}_n^{m,X,Y}$ from the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ of Figure 1, and a (single-input) scheme $\mathcal{FE}$ for the class $\mathcal{F}_1^{m,3X,Y}$.[12] For our transformation to work, we require $\mathcal{FE}$ to satisfy two properties. The first one, that we call *two-step decryption*, intuitively says that the $\mathcal{FE}$ decryption algorithm works in two steps: the first step uses the secret key to output an encoding of the result, while the second step returns the actual result $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ provided that the bounds $\|\boldsymbol{x}\|_\infty < X$, $\|\boldsymbol{y}\|_\infty < Y$ hold. The second property informally says that the $\mathcal{FE}$ encryption algorithm is additively homomorphic.

We note that the two-step property also says that the encryption algorithm accepts inputs $\boldsymbol{x}$ such that $\|\boldsymbol{x}\|_\infty > X$, yet correctness is guaranteed as long as the encrypted inputs are within the bound at the moment of invoking the second step of decryption.

Two-step decryption is formally defined as follows.

Property 1 (Two-step decryption). An FE scheme $\mathcal{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfies *two-step decryption* if it admits PPT algorithms $\mathsf{Setup}^\star$, $\mathsf{Dec}_1, \mathsf{Dec}_2$ and an encoding function $\mathcal{E}$ such that:

[12] The reason why we need $3X$ instead of X is due to maintain a correct distribution of the inputs in the security proof.

1. For all $\lambda, m, n, X, Y \in \mathbb{N}$, $\mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,X,Y}, 1^n)$ outputs $(\mathsf{msk}, \mathsf{mpk})$ where mpk includes a bound $B \in \mathbb{N}$, and the description of a group $\mathbb{G}$ (with group law $\circ$) of order $L > n \cdot m \cdot X \cdot Y$, which defines the encoding function $\mathcal{E} : \mathbb{Z}_L \times \mathbb{Z} \to \mathbb{G}$.
2. For all $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,X,Y}, 1^n)$, $\boldsymbol{x} \in \mathbb{Z}^m$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{msk}, \boldsymbol{x})$, $\boldsymbol{y} \in \mathbb{Z}^m$, and $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y})$, we have

$$\mathsf{Dec}_1(\mathsf{ct}, \mathsf{sk}) = \mathcal{E}(\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod L, \mathsf{noise}),$$

   for some $\mathsf{noise} \in \mathbb{N}$ that depends on ct and sk. Furthermore, it holds that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^m$, $\Pr[\mathsf{noise} < B] = 1 - \mathsf{negl}(\lambda)$, where the probability is taken over the random coins of $\mathsf{Setup}^\star$, Enc and KeyGen. Note that there is no restriction on the norm of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ here, and that we are assuming that Enc accepts inputs $\boldsymbol{x}$ whose norm may be larger than the bound.
3. Given any $\gamma \in \mathbb{Z}_L$, and mpk, one can efficiently compute $\mathcal{E}(\gamma, 0)$.
4. The encoding $\mathcal{E}$ is linear, that is: for all $\gamma, \gamma' \in \mathbb{Z}_L, , \mathsf{noise}, \mathsf{noise}' \in \mathbb{Z}$, we have

$$\mathcal{E}(\gamma, \mathsf{noise}) \circ \mathcal{E}(\gamma', \mathsf{noise}') = \mathcal{E}(\gamma + \gamma' \bmod L, \mathsf{noise} + \mathsf{noise}').$$

5. For all $\gamma < n \cdot m \cdot X \cdot Y$, and $\mathsf{noise} < n \cdot B$, $\mathsf{Dec}_2\big(\mathcal{E}(\gamma, \mathsf{noise})\big) = \gamma$.

The second property is as follows.

Property 2 (Linear encryption). For any FE scheme $\mathcal{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfying the two-step property, we define the following additional property. There exists a deterministic algorithm Add that takes as input a ciphertext and a message, such that for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^m$, the following are identically distributed:

$$\mathsf{Add}(\mathsf{Enc}(\mathsf{msk}, \boldsymbol{x}), \boldsymbol{x}'), \quad \text{and} \quad \mathsf{Enc}\big(\mathsf{msk}, (\boldsymbol{x} + \boldsymbol{x}' \bmod L)\big).$$

Note that the value $L \in \mathbb{N}$ is defined as part of the output of the algorithm Setup^* (see the two-step property above). We later use a single input FE with this property as a building block for a multi-input FE (see Figure 4); this property however is only used in the security proof of our transformation.

Instantiations. It is not hard to check that these two properties are satisfied by known functional encryption schemes for (bounded-norm) inner product. In particular, in Section 4 we show that this is satisfied by the many-AD-IND secure FE schemes of Agrawal, Libert and Stehlé [ALS16].[13] This allows us to obtain MIFE schemes for bounded-norm inner product based on a variety of assumptions such as plain DDH, Decisional Composite Residuosity, and LWE. In addition to obtaining the first schemes without the need of pairing groups, we also obtain schemes where decryption works efficiently even for large outputs. This stands in contrast to the previous result [AGRW17], where decryption requires to extract discrete logarithms.

Correctness. The correctness of the scheme $\mathcal{MIFE}$ follows from (i) the correctness and Property 1 (two-step decryption) of the single-input scheme, and (ii) from the correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ and the linear property of its decryption algorithm $\mathsf{Dec}^{\mathsf{ot}}$.

[13] While in [ALS16] the FE schemes are proven only one-AD-IND secure (i.e., for adversaries making a single encryption query), note that these are *public-key* schemes and thus many-AD-IND security can be obtained via a standard hybrid argument from one-AD-IND security.

$\underline{\mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:
$\boldsymbol{u} \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, \mathcal{F}_{L,n}^m)$, for all $i \in [n]$, $(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,3X,Y}, 1^n)$
$(\mathsf{mpk}, \mathsf{msk}) := \big(\{\mathsf{mpk}_i\}_{i\in[n]}, (\{\mathsf{msk}_i,\}_{i\in[n]}, \boldsymbol{u})\big)$
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i)}$:
$\boldsymbol{w}_i := \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i)$
Return $\mathsf{Enc}(\mathsf{msk}_i, \boldsymbol{w}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \boldsymbol{y}_1\| \cdots \|\boldsymbol{y}_n)}$:
For all $i \in [n]$, $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}_i, \boldsymbol{y}_i)$, $z \leftarrow \mathsf{KeyGen}^{\mathsf{ot}}(\boldsymbol{u}, \boldsymbol{y}_1\| \cdots \|\boldsymbol{y}_n)$
$\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \big(\{\mathsf{sk}_i\}_{i\in[n]}, z\big)$
Return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n}$

$\underline{\mathsf{Dec}'\big((\{\mathsf{sk}_i\}_{i\in[n]}, z), \mathsf{ct}_1, \ldots, \mathsf{ct}_n\big)}$:
For all $i \in [n]$, $\mathcal{E}(\langle \boldsymbol{x}_i + \boldsymbol{u}_i, \boldsymbol{y}_i\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_1(\mathsf{sk}_i, \mathsf{ct}_i)$
Return $\mathsf{Dec}_2\big(\mathcal{E}(\langle \boldsymbol{x}_1 + \boldsymbol{u}_1, \boldsymbol{y}_1\rangle \bmod L, \mathsf{noise}_1) \circ \cdots \circ \mathcal{E}(\langle \boldsymbol{x}_n + \boldsymbol{u}_n, \boldsymbol{y}_n\rangle \bmod L, \mathsf{noise}_n) \circ \mathcal{E}(-z, 0)\big)$

Fig. 4. Private-key multi-input FE scheme $\mathcal{MIFE} = (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the class $\mathcal{F}_n^{m,X,Y}$ from public-key single-input FE scheme $\mathcal{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the class $\mathcal{F}_1^{m,X,Y}$ and one-time multi-input FE $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$.

More precisely, consider any vector $\boldsymbol{x} := (\boldsymbol{x}_1\| \cdots \|\boldsymbol{x}_n) \in (\mathbb{Z}^m)^n$, $\boldsymbol{y} \in \mathbb{Z}^{mn}$, such that $\|\boldsymbol{x}\|_\infty < X$, $\|\boldsymbol{y}\|_\infty < Y$, and let $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})$, $\mathsf{sk}_{\boldsymbol{y}} \leftarrow \mathsf{KeyGen}'(\mathsf{msk}, \boldsymbol{y})$, and $\mathsf{ct}_i \leftarrow \mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i)$ for all $i \in [n]$.

By (2) of Property 1, the decryption algorithm $\mathsf{Dec}'(\mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ computes $\mathcal{E}(\langle \boldsymbol{w}_i, \boldsymbol{y}_i\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_1(\mathsf{sk}_i, \mathsf{ct}_i)$ where for all $i \in [n]$, $\mathsf{noise}_i < B$, with probability $1 - \mathsf{negl}(\lambda)$.

By (4) of Property 1 (linearity of $\mathcal{E}$), and the correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ we have:

$$\begin{aligned}&\mathcal{E}(\langle \boldsymbol{w}_1, \boldsymbol{y}_1\rangle \bmod L, \mathsf{noise}_1) \circ \cdots \circ \mathcal{E}(\langle \boldsymbol{w}_n, \boldsymbol{y}_n\rangle \bmod L, \mathsf{noise}_n) \circ \mathcal{E}(-z, 0)\\ &= \mathcal{E}\left(\mathsf{Dec}^{\mathsf{ot}}(z, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n), \sum_{i\in[n]} \mathsf{noise}_i\right) = \mathcal{E}\left(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \bmod L, \sum_{i\in[n]} \mathsf{noise}_i\right).\end{aligned}$$

Since $\langle \boldsymbol{x}, \boldsymbol{y}\rangle < n \cdot m \cdot X \cdot Y < L$ and $\sum_{i\in[n]} \mathsf{noise}_i < n \cdot B$, we have

$$\mathsf{Dec}_2\big(\mathcal{E}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \bmod L, \sum_{i\in[n]} \mathsf{noise}_i)\big) = \langle \boldsymbol{x}, \boldsymbol{y}\rangle,$$

by (5) of Property 1.

Proof of Security. In the following theorem we show that our construction is a many-AD-IND-secure MIFE, assuming that the underlying single-input FE scheme is many-AD-IND-secure, and the scheme $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND secure.

Theorem 3. *Assume that the single-input FE is many-AD-IND-secure and the multi-input FE $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND-secure. Then the multi-input FE $\mathcal{MIFE}$ in Figure 4 is many-AD-IND-secure. Namely, for any PPT adversary $\mathcal{A}$, there exist PPT adversaries $\mathcal{B}$ and $\mathcal{B}'$ such that*

$$\mathsf{Adv}^{\textit{many-AD-IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}}(\lambda) + n \cdot \mathsf{Adv}^{\textit{many-AD-IND}}_{\mathcal{FE},\mathcal{B}'}(\lambda).$$

Proof of Theorem 3. The proof proceeds by a sequence of games where G_0 is the $\mathbf{many\text{-}AD\text{-}IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{A})$ experiment. A formal description of all the experiments used in this proof is given in Figure 6, and a high-level summary is provided in Figure 5. For any game G_i, we denote by $\mathbf{Adv}_i(\mathcal{A})$ the advantage of $\mathcal{A}$ in G_i, that is, $\Pr[\mathsf{G}_i(1^\lambda, \mathcal{A}) = 1]$, where the probability is taken over the random coins of G_i and $\mathcal{A}$. In what follows we adopt the same notation from [AGRW17] for queried plaintexts, namely $(\boldsymbol{x}_i^{j,0}, \boldsymbol{x}_i^{j,1})$ denotes the j-th encryption query on the i-th slot.

Game	ct_i^j	justification/remark
G_0	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,0})$	
G_1	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1})$	one-AD-IND of $\mathcal{MIFE}^{\mathsf{ot}}$, Lemma 2
$\mathsf{G}_{1.\ell}$	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1} + \boldsymbol{x}_i^{1,1})$, for $i \leq \ell$ $\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1})$, for $i > \ell$	many-AD-IND of $\mathcal{FE}$, Lemma 3
G_2	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,1})$	$\mathsf{G}_2 = \mathsf{G}_{1.n}$

Fig. 5. An overview of the games used in the proof of Theorem 3.

Game G_1: Here we change the way the challenge ciphertexts are created. In particular, for all slots and all queries simultaneously, we switch from $\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,0})$ to $\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1})$.

G_1 can be proved indistinguishable from G_0 by relying on the one-time security of the multi-input scheme. More formally,

Lemma 2. *There exists a PPT adversary $\mathcal{B}_1$ against the one-AD-IND security of $\mathcal{MIFE}^{\mathsf{ot}}$ scheme such that*

$$|\mathbf{Adv}_0(\mathcal{A}) - \mathbf{Adv}_1(\mathcal{A})| \leq \mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE}^{\mathsf{ot}}, \mathcal{B}_1}(\lambda).$$

Proof. Here we replace encryptions of $\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,0}$ with encryptions of $\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1}$ in all slots simultaneously. Recall that here, j is the index of the encryption query while i is the index for the slot. The argument relies on the one-AD-IND security of the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$ and on the fact that ciphertexts produced by the latter can be used as plaintext for the underlying single input FE scheme $\mathcal{FE}$ that we are using as additional basic building block.

More in details, we build the adversary $\mathcal{B}_1$ so that it simulates G_β to $\mathcal{A}$ when interacting with experiment $\mathbf{one\text{-}AD\text{-}IND}_\beta^{\mathcal{MIFE}}$.

Initially $\mathcal{B}_1$ does not receive anything, since the one-AD-IND information-theoretically secure MIFE does not have any public key. For all $i \in [n]$ it runs $(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,3X,Y}, 1^n)$, and hands the public parameters to $\mathcal{A}$. Also, whenever $\mathcal{A}$ queries a secret key, $\mathcal{B}_1$ first queries its own oracle (on the same input) to get a corresponding key z. Next, for all $i \in [n]$, it sets $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}_i, \boldsymbol{y}_i)$ and gives back to $\mathcal{A}$ the secret key $\mathsf{sk}_{\boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n} := \big(\{\mathsf{sk}_i\}_{i \in [n]}, z\big)$.

When $\mathcal{A}$ asks encryption queries, $\mathcal{B}_1$ proceeds as follows. For each slot i, when receiving the first query $(i, \boldsymbol{x}_i^{1,0}, \boldsymbol{x}_i^{1,1})$, it computes the challenge ciphertext, for slot i, by invoking its own encryption oracle on the same input. Calling $\boldsymbol{w}_i^1 := \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i^{1,\beta})$ the received ciphertext, $\mathcal{B}_1$ computes $\mathsf{ct}_i^1 = \mathsf{Enc}(\mathsf{msk}_i, \boldsymbol{w}_i^1) = \mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i^{1,\beta})$.

Subsequent queries, on slot i, are answered as follows. $\mathcal{B}_1$ produces ct_i^j (for $j > 1$) by encrypting $\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{w}_i^1 \bmod L$, using msk_i. Note that $\mathsf{Enc}^{\mathsf{ot}}$ is linearly homomorphic (see Remark 3), thus, $\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{w}_i^1 \bmod L = \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i^{1,\beta} + \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0})$.

Finally, $\mathcal{B}_1$ outputs 1 iff $\mathcal{A}$ outputs 1. One can see that $\mathcal{B}_1$ provides a perfect simulation to $\mathcal{A}$ and thus:

$$|\mathbf{Adv}_0(\mathcal{A}) - \mathbf{Adv}_1(\mathcal{A})| \leq \mathsf{Adv}^{\textit{one-AD-IND}}_{\mathcal{MIFE},\mathcal{B}_1}(\lambda).$$

□

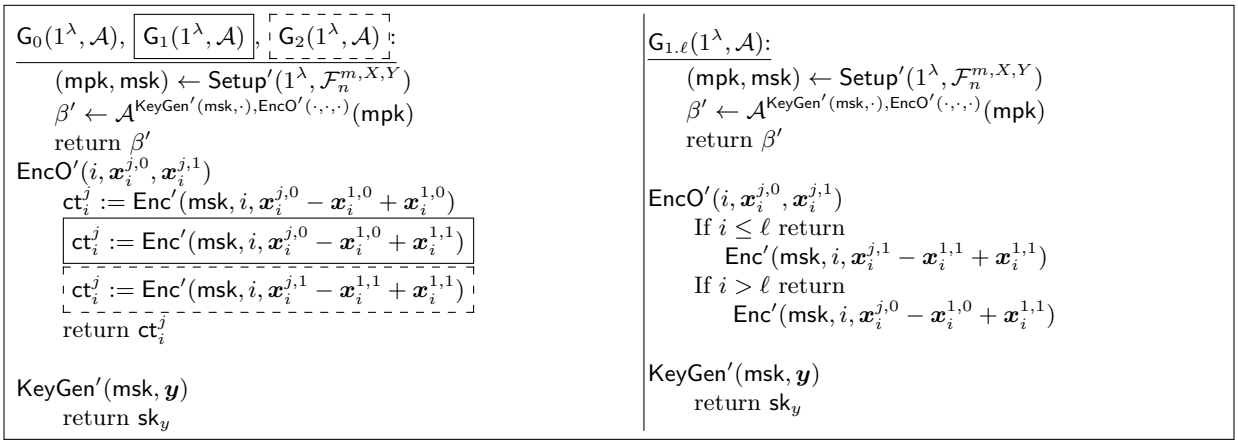

Fig. 6. Experiments for the proof of Theorem 3.

Game G_2: Here we change again the way the challenge ciphertexts are created. In particular, for all slots i and all queries j, we switch ct_i^j from $\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1})$ to $\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1} + \boldsymbol{x}_i^{1,1})$.

G_2 can be proved indistinguishable from G_1 via an hybrid argument over the n slots, relying on the security of the underlying single-input scheme.

By looking at the games defined in Figure 6, one can see that

$$|\mathbf{Adv}_1(\mathcal{A}) - \mathbf{Adv}_2(\mathcal{A})| = \sum_{\ell=1}^{n} |\mathbf{Adv}_{1,\ell-1}(\mathcal{A}) - \mathbf{Adv}_{1,\ell}(\mathcal{A})|$$

since G_1 corresponds to game $\mathsf{G}_{1.0}$ and whereas G_2 is identical to game $\mathsf{G}_{1.n}$.

Therefore, for every ℓ we bound the difference between each consecutive pair of games in the following lemma:

Lemma 3. *For every $\ell \in [n]$, there exists a PPT adversary $\mathcal{B}_{1.\ell}$ against the many-AD-IND security of the single-input scheme $\mathcal{FE}$ such that*

$$|\mathbf{Adv}_{1,\ell-1}(\mathcal{A}) - \mathbf{Adv}_{1,\ell}(\mathcal{A})| \leq \mathsf{Adv}^{\textit{many-AD-IND}}_{\mathcal{FE},\mathcal{B}_{1.\ell}}(\lambda).$$

Proof. Here, we replace encryptions of $\boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1}$ with encryptions of $\boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1} + \boldsymbol{x}_i^{1,1}$ in all slots. Let us recall that j is the index of the encryption query while i is the index for the

slot. The argument relies on (1) the many-AD-IND security of the underlying single input scheme $\mathcal{FE} := (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, (2) the fact that Enc satisfies Property 2 (linear encryption), and (3) the restrictions imposed by the security game (see [AGRW17]). As for this latter point we notice that, indeed, the security experiment restriction in the case of the inner product functionality imposes that $\langle \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0}, \boldsymbol{y}_i \rangle = \langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i \rangle$, for all slots $i \in [n]$. In our scheme this becomes $\langle \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0}, \boldsymbol{y}_i \rangle \bmod L = \langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i \rangle \bmod L$, which in turn is equivalent to

$$\langle \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1} + \boldsymbol{u}_i, \boldsymbol{y}_i \rangle \bmod L = \langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1} + \boldsymbol{x}_i^{1,1} + \boldsymbol{u}_i, \boldsymbol{y}_i \rangle \bmod L.$$

More formally, we build an adversary $\mathcal{B}_{1.\ell}$ that simulates $\mathsf{G}_{1.\ell-1+\beta}$ to $\mathcal{A}$ when interacting with the experiment $\mathbf{many\text{-}AD\text{-}IND}_{\beta}^{\mathcal{FE}}$.

$\mathcal{B}_{1.\ell}$ starts by receiving a public key for the scheme $\mathcal{FE}$, which is set to be the key mpk_ℓ for the ℓ-th instance of $\mathcal{FE}$. Next, it runs $\boldsymbol{u} \leftarrow \mathsf{Setup}^{\mathsf{ot}}$, and for all $i \neq \ell$, it runs $\mathsf{Setup}^{\star}$ to get $(\mathsf{mpk}_i, \mathsf{msk}_i)$. It gives $(\mathsf{mpk}_1, \ldots, \mathsf{mpk}_n)$ to $\mathcal{A}$.

$\mathcal{B}_{1.\ell}$ answers secret key queries $\boldsymbol{y} = \boldsymbol{y}_1 || \ldots || \boldsymbol{y}_n$ by first running $\mathsf{sk}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}_i, y_i)$ for $i \neq \ell$. Also it invokes its own key generation oracle on $\boldsymbol{y}_\ell$, to get sk_ℓ. Finally, it computes $z \leftarrow \mathsf{KeyGen}^{\mathsf{ot}}(\boldsymbol{u}, \boldsymbol{y}_1 || \ldots || \boldsymbol{y}_n)$ (recall that $\mathcal{B}_{1.\ell}$ knows $\boldsymbol{u}$). This key material is then sent to $\mathcal{A}$.

$\mathcal{B}_{1.\ell}$ answers encryption queries $(i, \boldsymbol{x}_i^{j,0}, \boldsymbol{x}_i^{j,1})$ to Enc' as follows.

If $i < \ell$, it computes $\mathsf{Enc}(\mathsf{msk}_i, \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i^{j,1}))$.

If $i > \ell$, it computes $\mathsf{Enc}(\mathsf{msk}_i, \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{u}, i, \boldsymbol{x}_i^{j,0} - \boldsymbol{x}_i^{1,0} + \boldsymbol{x}_i^{1,1}))$.

If $i = \ell$, at the j-th encryption query on slot ℓ, $\mathcal{B}_{1.\ell}$ queries its own oracle on input $(\boldsymbol{x}_\ell^{j,0} - \boldsymbol{x}_\ell^{1,0} + \boldsymbol{x}_\ell^{1,1}, \boldsymbol{x}_\ell^{j,1} - \boldsymbol{x}_\ell^{1,1} + \boldsymbol{x}_\ell^{1,1})$ (note that these vectors have norm less than $3X$, and as such, are valid input to the encryption oracle), to get back $\mathsf{ct}_*^j := \mathsf{Enc}\big(\mathsf{msk}_\ell, \boldsymbol{x}_\ell^{j,\beta} - \boldsymbol{x}_\ell^{1,\beta} + \boldsymbol{x}_\ell^{1,1}\big)$ from the experiment $\mathbf{many\text{-}AD\text{-}IND}_{\beta}^{\mathcal{FE}}$. Then, $\mathcal{B}_{1.\ell}$ computes $\mathsf{ct}_\ell^j := \mathsf{Add}(\mathsf{ct}_*^j, \boldsymbol{u}_\ell)$, and sends it to $\mathcal{A}$.

Note that by Property 2 ct_ℓ^j is identically distributed to $\mathsf{Enc}\big(\mathsf{msk}_\ell, \boldsymbol{x}_\ell^{j,\beta} - \boldsymbol{x}_\ell^{1,\beta} + \boldsymbol{x}_\ell^{1,1} + \boldsymbol{u}_\ell \bmod L\big)$, the latter being equal to $\mathsf{Enc}\big(\mathsf{msk}_\ell, \mathsf{Enc}^{\mathsf{ot}}(\boldsymbol{x}_\ell^{j,\beta} - \boldsymbol{x}_\ell^{1,\beta} + \boldsymbol{x}_\ell^{1,1})\big)$. Also, we remark that because $\mathcal{B}_{1.\ell}$ plays in the many-AD-IND security game, it can make several queries to its encryption oracle, which means that every ct_*^j obtained from the oracle is encrypted under fresh randomness r_j, i.e., $\mathsf{ct}_*^j := \mathsf{Enc}\big(\mathsf{msk}_\ell, \boldsymbol{x}_\ell^{j,\beta} - \boldsymbol{x}_\ell^{1,\beta} + \boldsymbol{x}_\ell^{1,1}; r_j\big)$. Therefore, the simulated ciphertext ct_ℓ^j uses randomness r_j which is independent of the randomness $r_{j'}$ used in $\mathsf{ct}_\ell^{j'}$, for all $j \neq j'$. This means ct_ℓ^j is distributed as in game $\mathsf{G}_{1.\ell-1+\beta}$.

Finally, $\mathcal{B}_{1.\ell}$ outputs the same bit β' returned by $\mathcal{A}$. Thus:

$$|\mathbf{Adv}_{1.\ell-1}(\mathcal{A}) - \mathbf{Adv}_{1.\ell}(\mathcal{A})| \leq \mathsf{Adv}^{\textit{many-AD-IND}}_{\mathcal{FE},\mathcal{B}_{1.\ell}}(\lambda).$$

□

The proof of Theorem 3 follows by combining the bounds obtained in the previous lemmas. □

4 Concrete instances of FE for Inner Product

In this section we discuss three instantiations of our generic construction from Section 3.3. In particular, we show that the existing (single-input) FE schemes proposed by Agrawal et al. [ALS16] (that are proven many-AD-IND-secure) satisfy Property 1 (two-step decryption) and Property 2 (linear encryption). Note that all of these (single-input) FE schemes happen to be public-key, although we stress that this property is not required by our generic construction from Section 3.3.

4.1 Inner Product FE from MDDH

Here we show that the many-AD-IND secure Inner Product FE from [ALS16, Section 3], generalized to the $\mathcal{D}_k$-MDDH setting, as in [AGRW17, Figure 15], and recalled in Figure 7, satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m,X,Y}, \ell)}$:
$\mathcal{G} := (\mathbb{G}, p, g) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$
$\mathsf{mpk} := (\mathcal{G}, [\mathbf{A}], [\mathbf{WA}])$, $\mathsf{msk} := \mathbf{W}$
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x} \in \mathbb{Z}_p^m)}$:
$\boldsymbol{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{c} := \begin{pmatrix} -\mathbf{A}\boldsymbol{r} \\ \boldsymbol{x} + \mathbf{WA}\boldsymbol{r} \end{pmatrix}$
Return $\mathsf{ct}_{\boldsymbol{x}} := [\boldsymbol{c}] \in \mathbb{G}^{k+m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y} \in \mathbb{Z}_p^m)}$:
Return $\mathsf{sk}_{\boldsymbol{y}} := \begin{pmatrix} \mathbf{W}^\top \boldsymbol{y} \\ \boldsymbol{y} \end{pmatrix} \in \mathbb{Z}_p^{k+m+1}$

$\underline{\mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}_{\boldsymbol{x}} := [\boldsymbol{c}], \mathsf{sk}_{\boldsymbol{y}})}$:
$C := [\boldsymbol{c}^\top \mathsf{sk}_{\boldsymbol{y}}]$
Return $\log(C)$

Fig. 7. Functional encryption scheme for the class $\mathcal{F}_1^{m,X,Y}$, based the $\mathcal{D}_k$-MDDH assumption.

Property 1 (two-step decryption).

1. The algorithm $\mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,X,Y}, 1^n)$ works the same as Setup except that it additionally uses n to ensure that $n \cdot m \cdot X \cdot Y = \mathsf{poly}(\lambda)$ (which implies $n \cdot m \cdot X \cdot Y < p$). Also, it returns the bound $B := 0$, $L := p$, $\mathbb{G}$ as the same group of order p generated by $\mathsf{GGen}(1^\lambda)$, and the encoding function $\mathcal{E} : \mathbb{Z}_p \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_p, \mathsf{noise} \in \mathbb{Z}$ as
$$\mathcal{E}(\gamma, \mathsf{noise}) := [\gamma].$$
We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 7 respectively.
2. We have for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^m$,
$$\mathsf{Dec}_1\big(\mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}} := [\boldsymbol{c}]\big) := [\boldsymbol{c}]^\top \mathsf{sk}_{\boldsymbol{y}} = [\langle \boldsymbol{x}, \boldsymbol{y} \rangle] = \mathcal{E}(\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod p, 0).$$
3. It is straightforward to see that $\mathcal{E}(\gamma, 0)$ is efficiently and publicly computable.
4. It is also easy to see that $\mathcal{E}$ is linear.
5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma < n \cdot m \cdot X \cdot Y$,
$$\mathsf{Dec}_2(\mathcal{E}(\gamma \bmod p, 0)) := \log([\gamma \bmod p]) = \gamma \bmod p = \gamma,$$
where the log can be computed efficiently since $\gamma < n \cdot m \cdot X \cdot Y$ is assumed to lie in a polynomial size range.

Property 2 (linear encryption). For all $\boldsymbol{x}' \in \mathbb{Z}^m$ and $[\boldsymbol{c}] \in \mathbb{G}^{m+k+1}$, let $\mathsf{Add}([\boldsymbol{c}], \boldsymbol{x}') := [\boldsymbol{c}] \cdot \begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}' \end{bmatrix}$. Then, for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^m$, and $[\boldsymbol{c}] := \mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x}) = \begin{bmatrix} -\mathbf{A}\boldsymbol{r} \\ \boldsymbol{x} + \mathbf{W}\mathbf{A}\boldsymbol{r} \end{bmatrix}$, we have:

$$\mathsf{Add}([\boldsymbol{c}], \boldsymbol{x}') = [\boldsymbol{c}] \cdot \begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}' \end{bmatrix} = \begin{bmatrix} -\mathbf{A}\boldsymbol{r} \\ \boldsymbol{x} + \boldsymbol{x}' + \mathbf{W}\mathbf{A}\boldsymbol{r} \end{bmatrix} = \mathsf{Enc}\big(\mathsf{mpk}, (\boldsymbol{x} + \boldsymbol{x}' \bmod p)\big).$$

4.2 Inner Product FE from LWE

Here we show that the many-AD-IND secure Inner Product FE from [ALS16, Section 4.1] and recalled in Figure 8, satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m,X,Y})}$:
Let $N = N(\lambda)$, and set integers $M, q \geq 2$, real $\alpha \in (0,1)$, and distribution $\mathcal{D}$ over $\mathbb{Z}^{m \times M}$ as explained in [ALS16]; set $K := m \cdot X \cdot Y$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{M \times N}$, $\mathbf{Z} \leftarrow_{\mathrm{R}} \mathcal{D}$, $\mathbf{U} := \mathbf{Z}\mathbf{A} \in \mathbb{Z}_q^{m \times N}$, $\mathsf{mpk} := (K, \mathbf{A}, \mathbf{U})$, $\mathsf{msk} := \mathbf{Z}$.
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x} \in \mathbb{Z}^m)}$:
$\boldsymbol{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^N$, $\boldsymbol{e}_0 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^M$, $\boldsymbol{e}_1 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^m$
$\boldsymbol{c}_0 := \mathbf{A}\boldsymbol{s} + \boldsymbol{e}_0 \in \mathbb{Z}_q^M$
$\boldsymbol{c}_1 := \mathbf{U}\boldsymbol{s} + \boldsymbol{e}_1 + \boldsymbol{x} \cdot \lfloor \frac{q}{K} \rfloor \in \mathbb{Z}_q^m$
Return $\mathsf{ct}_{\boldsymbol{x}} := (\boldsymbol{c}_0, \boldsymbol{c}_1)$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y} \in \mathbb{Z}^m)}$:
Return $\mathsf{sk}_{\boldsymbol{y}} := \begin{pmatrix} \mathbf{Z}^\top \boldsymbol{y} \\ \boldsymbol{y} \end{pmatrix} \in \mathbb{Z}^{M+m}$

$\underline{\mathsf{Dec}\big(\mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}}\big)}$:
$\mu' := \begin{pmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \end{pmatrix}^\top \mathsf{sk}_{\boldsymbol{y}} \bmod q$.
Return $\mu \in \{-K+1, \ldots, K-1\}$ that minimizes $\big|\lfloor \frac{q}{K} \rfloor \mu - \mu'\big|$.

Fig. 8. Functional encryption scheme for the class $\mathcal{F}_1^{m,X,Y}$, based on the LWE assumption.

Property 1 (two-step decryption).

1. The algorithm $\mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,X,Y}, 1^n)$ works the same as Setup except that it uses n to set $K := n \cdot m \cdot X \cdot Y$,[14] and it also returns the bound $B := \lfloor \frac{q}{K} \rfloor$, $L := q$, $\mathbb{G} := (\mathbb{Z}_q, +)$, and the encoding function $\mathcal{E} : \mathbb{Z}_q \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_q, \mathsf{noise} \in \mathbb{Z}$ as

$$\mathcal{E}(\gamma \bmod q, \mathsf{noise}) := \gamma \cdot \lfloor \tfrac{q}{K} \rfloor + \mathsf{noise} \bmod q.$$

   We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 8 respectively.

[14] Also, parameters M, q, α and distribution $\mathcal{D}$ are chosen as explained in [ALS16], as if working with input vectors of dimension $n \cdot m$.

2. We have for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^m$,

$$\begin{aligned}\mathsf{Dec}_1\big(\mathsf{sk}_{\boldsymbol{y}}, \mathsf{ct}_{\boldsymbol{x}} := (\boldsymbol{c}_0, \boldsymbol{c}_1)\big) &= \begin{pmatrix}\boldsymbol{c}_0\\ \boldsymbol{c}_1\end{pmatrix}^\top \mathsf{sk}_{\boldsymbol{y}} \bmod q \\ &= \langle \boldsymbol{x}, \boldsymbol{y}\rangle \cdot \lfloor \tfrac{q}{K} \rfloor + \boldsymbol{y}^\top \boldsymbol{e}_1 - \boldsymbol{e}_0^\top \mathbf{Z}^\top \boldsymbol{y} \bmod q \\ &= \mathcal{E}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \bmod q, \mathsf{noise}),\end{aligned}$$

   where $\mathsf{noise} := \boldsymbol{y}^\top \boldsymbol{e}_1 - \boldsymbol{e}_0^\top \mathbf{Z}^\top \boldsymbol{y}$, and $\Pr[\mathsf{noise} < B] = 1 - \mathsf{negl}(\lambda)$.
3. It is straightforward to see that $\mathcal{E}(\gamma, 0)$ is efficiently and publicly computable.
4. It is also easy to see that $\mathcal{E}$ is linear.
5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma < n \cdot m \cdot X \cdot Y$, and $\mathsf{noise} < n \cdot B$,

$$\mathsf{Dec}_2(\mathcal{E}(\gamma \bmod q, \mathsf{noise})) = \gamma,$$

   follows by the same decryption correctness argument in [ALS16], with the only difference that here we used a larger bound K.

Property 2 (linear encryption). For all $\boldsymbol{x}' \in \mathbb{Z}^m$ and $(\boldsymbol{c}_0, \boldsymbol{c}_1) \in \mathbb{Z}_q^{M+m}$, let $\mathsf{Add}((\boldsymbol{c}_0, \boldsymbol{c}_1), \boldsymbol{x}') := (\boldsymbol{c}_0, \boldsymbol{c}_1) + (0, \boldsymbol{x}' \cdot \lfloor \frac{q}{K} \rfloor) \bmod q$. Then, for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^m$, and $(\boldsymbol{c}_0, \boldsymbol{c}_1) := (\mathbf{A}\boldsymbol{s} + \boldsymbol{e}_0, \mathbf{U}\boldsymbol{s} + \boldsymbol{e}_1 + \boldsymbol{x} \cdot \lfloor \frac{q}{K} \rfloor) \in \mathbb{Z}_q^{M+m}$, we have:

$$\mathsf{Add}((\boldsymbol{c}_0, \boldsymbol{c}_1), \boldsymbol{x}') = (\mathbf{A}\boldsymbol{s} + \boldsymbol{e}_0, \mathbf{U}\boldsymbol{s} + \boldsymbol{e}_1 + (\boldsymbol{x} + \boldsymbol{x}') \cdot \lfloor \tfrac{q}{K} \rfloor) \bmod q = \mathsf{Enc}\big(\mathsf{mpk}, (\boldsymbol{x} + \boldsymbol{x}' \bmod q)\big).$$

4.3 Inner Product FE from Paillier

Here we show that the Inner Product FE from [ALS16, Section 5.1] and recalled in Figure 9 satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

Property 1 (two-step decryption).

1. The algorithm $\mathsf{Setup}^\star(1^\lambda, \mathcal{F}_1^{m,X,Y}, 1^n)$ works the same as Setup except that it additionally uses n to ensure $n \cdot m \cdot X \cdot Y < N$. Also, it returns the bound $B := 0$, $L := N$, $\mathbb{G}$ as the subgroup of $\mathbb{Z}_{N^2}^*$ of order N generated by $(1 + N)$, and the encoding function $\mathcal{E} : \mathbb{Z}_N \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_N, \mathsf{noise} \in \mathbb{Z}$ as

$$\mathcal{E}(\gamma, \mathsf{noise}) := 1 + \gamma \cdot N \bmod N^2.$$

   We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 9.
2. We have for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^m$,

$$\mathsf{Dec}_1(\mathsf{sk}_{\boldsymbol{y}} := (d, \boldsymbol{y}), \mathsf{ct}_{\boldsymbol{x}}) := \left(\prod_{j \in [m]} C_j^{y_j}\right) \cdot C_0^{-d} \bmod N^2 = \mathcal{E}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \bmod N, 0).$$

3. It is straightforward to see that see that $\mathcal{E}(\gamma, 0)$ can be efficiently computed from public information.
4. It is also easy to see that $\mathcal{E}$ is linear.
5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma \leq n \cdot m \cdot X \cdot Y < N$, it holds

$$\mathsf{Dec}_2(\mathcal{E}(\gamma, 0)) := \frac{\mathcal{E}(\gamma,0) - 1 \bmod N^2}{N} = \gamma.$$

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m,X,Y})}$:
Choose primes $p = 2p' + 1$, $q = q' + 1$ with prime $p', q' > 2^{l(\lambda)}$ for an $l(\lambda) = \mathsf{poly}(\lambda)$ such that factoring is λ-hard, and set $N := pq$ ensuring that $m \cdot X \cdot Y < N$. Sample $g' \leftarrow_R \mathbb{Z}_{N^2}^*$, $g := g'^{2N} \bmod N^2$, $\boldsymbol{s} \leftarrow_R D_{\mathbb{Z}^m,\sigma}$, for standard deviation $\sigma > \sqrt{\lambda} \cdot N^{5/2}$, and for all $j \in [m]$, $h_j := g^{s_j} \bmod N^2$.
$(\mathsf{mpk}, \mathsf{msk}) := \big((N, g, \{h_j\}_{j\in[m]}, X, Y), \{s_j\}_{j\in[m]}\big)$
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{mpk}, \boldsymbol{x} \in \mathbb{Z}^m)}$:
$r \leftarrow_R \{0, \ldots, \lfloor N/4 \rfloor\}$, $C_0 := g^r \in \mathbb{Z}_{N^2}$, for all $j \in [m]$, $C_j := (1 + x_j N) \cdot h_j^r \in \mathbb{Z}_{N^2}$
Return $\mathsf{ct}_{\boldsymbol{x}} := (C_0, \ldots, C_m) \in \mathbb{Z}_{N^2}^{m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y} \in \mathbb{Z}^m)}$:
$d := \sum_{j\in[m]} y_j s_j \in \mathbb{Z}$.
Return $\mathsf{sk}_{\boldsymbol{y}} := (d, \boldsymbol{y})$

$\underline{\mathsf{Dec}\big(\mathsf{sk}_{\boldsymbol{y}} := (d, \boldsymbol{y}), \mathsf{ct}_{\boldsymbol{x}}\big)}$:
$C := \Big(\prod_{j\in[m]} C_j^{y_j}\Big) \cdot C_0^{-d} \bmod N^2$.
Return $\log_{(1+N)}(C) := \frac{C-1 \bmod N^2}{N}$.

Fig. 9. Functional encryption scheme for the class $\mathcal{F}_1^{m,X,Y}$, based on the Paillier cryptosystem.

Property 2 (linear encryption). For all $\boldsymbol{x}' \in \mathbb{Z}^m$ and $(C_0, C_1', \ldots, C_m') \in \mathbb{Z}_{N^2}^{m+1}$, let $\mathsf{Add}((C_0, C_1, \ldots, C_m), \boldsymbol{x}')$ computes $C_j' := C_j \cdot (1 + x_j' N) \bmod N^2$ for all $j \in [m]$ and outputs $(C_0, C_1', \ldots, C_m')$. Then, for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{Z}^m$, and $(C_0, C_1, \ldots, C_m) := (g^r, (1 + x_1 N) \cdot h_1^r, \ldots, (1 + x_m N) \cdot h_m^r) \in \mathbb{Z}_{N^2}^m$, we have:

$$\begin{aligned}\mathsf{Add}((C_0, C_1, \ldots, C_m), \boldsymbol{x}') &= (g^r, (1 + (x_1 + x_1')N) \cdot h_1^r \bmod N^2, \ldots, (1 + (x_m + x_m')N) \cdot h_m^r \bmod N^2) \\ &= \mathsf{Enc}\big(\mathsf{mpk}, (\boldsymbol{x} + \boldsymbol{x}' \bmod N)\big).\end{aligned}$$

4.4 MIFE for Inner Product from DDH

Here we present a concrete construction of MIFE for Inner Product whose security relies on the DDH assumption (without pairings). This concrete instantiation can be found in Figure 10.

5 Function-Hiding Multi-Input FE for Inner Product

In this section, we give a function-hiding MIFE. We transform the MIFE for inner product proposed by Abdalla et al. in [AGRW17] into a function-hiding scheme, using a double layered encryption approach, similar to the one of Lin [Lin17]. Namely, in Section 5.1, we give a generic construction that use any single-input FE on top of the MIFE from [AGRW17], which can prove selectively secure. Unlike the results in Section 3 that can be instantiated without pairings, for function-hiding we rely on pairing groups. Finally, in Section 5.2, we prove adaptive security, considering a specific instantiation of our construction.

Our construction. We present our function-hiding scheme $\mathcal{MIFE}$ in Figure 12. The construction relies on the multi-input scheme $\mathcal{MIFE}'$ of Abdalla et al. [AGRW17] (recalled in Figure 11), used together with any one-SEL-SIM secure, single-input FE for the functionality

$$\mathcal{F}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T}^{\ell} = \{f_{[\boldsymbol{y}]_1} : \mathbb{G}_2^\ell \to \mathbb{G}_T \text{ for } [\boldsymbol{y}]_1 \in \mathbb{G}_1^\ell\},$$

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:
$\mathcal{G} := (\mathbb{G}, p, g) \leftarrow \mathsf{GGen}(1^\lambda)$, $a \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $\boldsymbol{a} := \binom{1}{a}$, $\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times 2}$, $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^m$
$\mathsf{mpk} := (\mathcal{G}, [\boldsymbol{a}], [\mathbf{W}\boldsymbol{a}])$, $\mathsf{msk} := \big(\mathbf{W}, (\boldsymbol{u}_i)_{i\in[n]}\big)$
Return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i \in \mathbb{Z}_p^m)}$:
$r_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, return $\mathsf{ct}_i := ([\boldsymbol{a}r_i], [\boldsymbol{x}_i + \boldsymbol{u}_i + \mathbf{W}_i\boldsymbol{a}r_i]) \in \mathbb{G}^{m+2}$

$\underline{\mathsf{KeyGen}\big(\mathsf{msk}, \boldsymbol{y} := (\boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n) \in \mathbb{Z}_p^{m\cdot n}\big)}$:
Return $\mathsf{sk}_{\boldsymbol{y}} := \Big(\boldsymbol{y}, (\boldsymbol{y}_i^\top \mathbf{W}_i)_{i\in[n]}, \sum_{i\in[n]} \boldsymbol{y}_i^\top \boldsymbol{u}_i\Big) \in \mathbb{Z}_p^{n(m+2)+1}$

$\underline{\mathsf{Dec}\Big(\mathsf{mpk}, \mathsf{ct}_i := ([\boldsymbol{t}_i], [\boldsymbol{c}_i]), \mathsf{sk}_{\boldsymbol{y}} := \big(\boldsymbol{y}, (\boldsymbol{d}_i^\top)_{i\in[n]}, z\big)\Big)}$:
$C := \Big(\prod_{i\in[n]} \big([\boldsymbol{y}_i^\top \boldsymbol{c}_i] / [\boldsymbol{d}_i^\top \boldsymbol{t}_i]\big)\Big) / [z]$
Return $\log(C)$

Fig. 10. Multi-Input Functional encryption scheme for the class $\mathcal{F}_n^{m,X,Y}$, based on the DDH assumption.

where

$$f_{[\boldsymbol{y}]_1}([\boldsymbol{x}]_2) := [\langle \boldsymbol{x}, \boldsymbol{y}\rangle]_T\,,$$

$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, g_1, g_2)$ is a pairing group, and ℓ is the size of the ciphertext and secret keys in $\mathcal{MIFE}'$.

Concretely, we use the single-input FE from [ALS16], generalized to the MDDH assumption, whose one-SEL-SIM security is proven in [AGRW17,Wee17], and whose description is recalled in Figure 7. Note that this single-input FE happens to be public-key, but this is not a property that we need for our overall MIFE.

Outline of the construction. Our starting point is the MIFE scheme for inner-products from [AGRW17], denoted by $\mathcal{MIFE}' := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ and recalled in Figure 11. This scheme is clearly not function-hiding, as the vector $\boldsymbol{y}$ is given in the clear as part of functional secret key, in order to make decryption possible. In order to avoid the leakage of $\boldsymbol{y}$, we employ an approach similar to the one proposed in [Lin17], which intuitively consists into adding a layer of encryption on top of the MIFE keys and ciphertexts; this is done by using a *single-input* inner product encryption scheme $\mathcal{FE}$. Slightly more in detail, using the $\mathcal{FE}$ and $\mathcal{MIFE}'$ schemes, we design our new function-hiding multi-input scheme $\mathcal{MIFE}$ as follows.

We generate master keys $(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\lambda, \mathcal{F}^\ell_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T})$ for computing inner products on vectors of dimension ℓ, where ℓ is the size of the ciphertexts and secret keys of $\mathcal{MIFE}'$. To encrypt $\boldsymbol{x}_i \in \mathbb{Z}_p^m$ for each slot $i \in [n]$, we first compute $[\mathsf{ct}_i^{\mathsf{in}}]_1$ using $\mathcal{MIFE}'$, and then we compute $\mathsf{ct}_i^{\mathsf{out}} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in}}]_1)$. To generate a key for $\boldsymbol{y} := (\boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n) \in \mathbb{Z}_p^{nm}$, we first compute the keys $\mathsf{sk}^{\mathsf{in}}$ from $\mathcal{MIFE}'$, and then we would like to encrypt these keys using $\mathcal{FE}$ in order to hide information about $\boldsymbol{y}$. A generic way to do it would be to set our secret key to be $\mathsf{Enc}(\mathsf{msk}_i, \mathsf{sk}^{\mathsf{in}})$, for all possible $i \in [n]$, so that we can compute the inner product of $[\mathsf{ct}_i^{\mathsf{in}}]_1$ with $\mathsf{sk}^{\mathsf{in}}$ for all $i \in [n]$. But that would yield keys of size $O(n^2 m)$, since the key $\mathsf{sk}^{\mathsf{in}}$ itself is of size $O(nm)$. We can do better, however. If we consider the specific $\mathcal{MIFE}'$ scheme from [AGRW17], a secret key $\mathsf{sk}^{\mathsf{in}}$ for $\boldsymbol{y}$ consists of the components $([\mathsf{sk}_1^{\mathsf{in}}\| \ldots \|\mathsf{sk}_n^{\mathsf{in}}]_2, [z]_T)$, where each $[\mathsf{sk}_i^{\mathsf{in}}]_2$ only depends on $\boldsymbol{y}_i$ and is of size $O(m)$, while $[z]_T \in \mathbb{G}_T$ does not depend on $\boldsymbol{y}$ at all. Hence, we encrypt each vectors $[\mathsf{sk}_i^{\mathsf{in}}]_2$

to obtain $\mathsf{sk}_i^{\mathsf{out}} := \mathcal{FE}.\mathsf{Enc}(\mathsf{mpk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$, which gives us a secret key $\mathsf{sk}^{\mathsf{out}} := \left(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T\right)$ of total size $O(nm)$.

This way, decrypting the outer layer as $\mathcal{FE}.\mathsf{Dec}(\mathsf{sk}_i^{\mathsf{out}}, \mathsf{ct}_i^{\mathsf{out}})$ yields $[\langle \mathsf{sk}_i^{\mathsf{in}}, \mathsf{ct}_i^{\mathsf{in}}\rangle]_T$, which is what needs to be computed in the $\mathcal{MIFE}'$ decryption algorithm Dec'. More precisely, correctness of $\mathcal{MIFE}$ follows from the correctness of $\mathcal{MIFE}'$, and the structural requirement of $\mathcal{FE}.\mathsf{Dec}$ that is used in the $\mathcal{MIFE}'$ decryption algorithm, namely:

$$\begin{aligned}
&\mathcal{MIFE}.\mathsf{Dec}(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T, \{\mathsf{ct}_i^{\mathsf{out}}\}_{i\in[n]}) \\
&= \prod_{i=1}^{n} \mathcal{FE}.\mathsf{Dec}(\mathsf{ct}_i^{\mathsf{out}}, \mathsf{sk}_i^{\mathsf{out}})/[z]_T = \prod_{i=1}^{n} [\langle \mathsf{sk}_i^{\mathsf{in}}, \mathsf{ct}_i^{\mathsf{in}}\rangle]_T/[z]_T \\
&= \mathcal{MIFE}'.\mathsf{Dec}(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T, \{[\mathsf{ct}_i^{\mathsf{in}}]_1\}_{i\in[n]}).
\end{aligned}$$

Definition 11 (one-SEL-SIM-secure FE). *A single-input functional encryption $\mathcal{FE}$ for the functionality $\mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T}$ is one-SEL-SIM-secure if there exist PPT simulator algorithms $(\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ such that for every PPT (stateful) adversary $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, the following two distributions are computationally indistinguishable:*

Experiment $\mathbf{REAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(1^\lambda, \mathcal{A})$:	*Experiment* $\mathbf{IDEAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(1^\lambda, \mathcal{A})$:
$\boldsymbol{x} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T})$	$\boldsymbol{x} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T})$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T})$	$(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{msk}}) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T})$
$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{msk}, \boldsymbol{x})$	$\mathsf{ct} \leftarrow \widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}})$
$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{mpk}, \mathsf{ct})$	$\alpha \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\widetilde{\mathsf{mpk}}, \mathsf{ct})$
Output: α	***Output:*** α

The oracle $\mathcal{O}(\cdot)$ in the ideal experiment above is given access to another oracle that, given $[\boldsymbol{y}]_1 \in \mathcal{F}^{\ell}_{\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T}$, returns $[\langle \boldsymbol{x}, \boldsymbol{y}\rangle]_1$, and then $\mathcal{O}(\cdot)$ returns $\widetilde{\mathsf{KeyGen}}\left(\widetilde{\mathsf{msk}}, [\boldsymbol{y}]_1, [\langle \boldsymbol{x}, \boldsymbol{y}\rangle]_1\right)$.

For every stateful adversary $\mathcal{A}$, we define its advantage as

$$\begin{aligned}
&\mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{A}}(\lambda) \\
&= \left|\Pr\left[\mathbf{REAL}^{\mathcal{FE}}_{\mathbf{SEL}}(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathbf{IDEAL}^{\mathcal{FE}}_{\mathbf{SEL}}(1^\lambda, \mathcal{A}) = 1\right]\right|,
\end{aligned}$$

and we require that for every PPT $\mathcal{A}$, there exists a negligible function negl *such that for all $\lambda \in \mathbb{N}$, $\mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.*

5.1 Proof of Selective Security

In the following theorem we state the selective security of our scheme $\mathcal{MIFE}$. Precisely, the theorem proves that our scheme is weakly function-hiding. We stress that this does not entail any limitation in the final result, as full-fledged function-hiding can be achieved in a generic way via a simple transformation, proposed in [LV16] (for single-input FE). The main idea is to work with slightly larger vectors where both input vectors $\boldsymbol{x}$ and secret-key vectors $\boldsymbol{y}$ are padded with zeros. In Appendix B we show how to do this transformation in the multi-input setting.

Multi-input scheme $\mathcal{MIFE}'$[AGRW17]

$\underline{\mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:
$\forall i \in [n]$:
  $(\mathsf{mpk}'_i, \mathsf{msk}'_i) \leftarrow \mathcal{FE}'.\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m+k,X,Y})$
$\forall i \in [n] : \boldsymbol{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$
$\mathsf{mpk}' := \left(\{\mathsf{mpk}'_i\}_{i\in[n]}\right)$
$\mathsf{msk}' := \left(\{\mathsf{msk}'_i, \boldsymbol{z}_i\}_{i\in[n]}\right)$
return $(\mathsf{mpk}', \mathsf{msk}')$

$\underline{\mathsf{Dec}'\left(\left(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T\right), \{[\mathsf{ct}_i^{\mathsf{in}}]_1\}_{i\in[n]}\right)}$:
$\forall i \in [n] : [a_i]_T \leftarrow \mathcal{FE}'.\mathsf{Dec}([\mathsf{sk}_i^{\mathsf{in}}]_2, [\mathsf{ct}_i^{\mathsf{in}}]_1)$
return the discrete log of $\left(\prod_{i=1}^n [a_i]_T\right)/[z]_T$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \boldsymbol{x}_i)}$:
$[\mathsf{ct}_i^{\mathsf{in}}]_1 := \mathcal{FE}'.\mathsf{Enc}(\mathsf{mpk}'_i, \boldsymbol{x}_i \| \boldsymbol{z}_i)$
return $[\mathsf{ct}_i^{\mathsf{in}}]_1$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n)}$:
$\boldsymbol{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$
$\forall i \in [n]$:
  $\mathsf{sk}_i^{\mathsf{in}} \leftarrow \mathcal{FE}'.\mathsf{KeyGen}(\mathsf{msk}'_i, \boldsymbol{y}_i \| \boldsymbol{r})$
$z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r} \rangle$
$\mathsf{sk}^{\mathsf{in}} := \left(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T\right)$
return $\mathsf{sk}^{\mathsf{in}}$

Fig. 11. Multi-input, FE for $\mathcal{F}_n^{m,X,Y}$ from [AGRW17], whose many-SEL-IND relies on the $\mathcal{D}_k$-MDDH assumption. Here $\mathcal{FE}' := (\mathcal{FE}'.\mathsf{Setup}, \mathcal{FE}'.\mathsf{Enc}, \mathcal{FE}'.\mathsf{KeyGen}, \mathcal{FE}'.\mathsf{Dec})$ is a one-SEL-SIM secure, public-key, single-input FE for $\mathcal{F}_1^{m+k,X,Y}$, where k is the parameter used by the $\mathcal{D}_k$-MDDH assumption (concretely, $k = 1$ for SXDH, $k = 2$ for DLIN).

Theorem 4 (many-SEL-wFH-IND security). *Let $\mathcal{MIFE}'$ be the many-SEL-IND secure multi-input FE from Figure 11. Suppose the single-input $\mathcal{FE} := (\mathcal{FE}.\mathsf{Setup}, \mathcal{FE}.\mathsf{Enc}, \mathcal{FE}.\mathsf{KeyGen}, \mathcal{FE}.\mathsf{Dec})$ is one-SEL-SIM-secure. Then the multi-input scheme $\mathcal{MIFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ in Figure 12 is many-SEL-wFH-IND-secure.*

Proof Overview. The proof is done via a hybrid argument that consists of two main phases: we first switch the ciphertexts from encryptions of $\boldsymbol{x}_i^{j_i,0}$ to encryptions of $\boldsymbol{x}_i^{j_i,1}$ for all slots $i \in [n]$, and ciphertext queries $j_i \in [Q_i]$, where Q_i denotes the number of ciphertext query on the i'th slot. This change is justified by the many-SEL-IND security of the underlying $\mathcal{MIFE}'$ in a black box manner. In addition, this change relies on the weak-function-hiding property that imposes the constraints $\sum_{i=1}^n \langle \boldsymbol{x}_i^{j_i,0}, \boldsymbol{y}_i^{j_f,0} \rangle = \sum_{i=1}^n \langle \boldsymbol{x}_i^{j_i,1}, \boldsymbol{y}_i^{j_f,0} \rangle$, for all secret key queries $j_f \in [Q_f]$, where Q_f denotes the number of secret key queries, which thus disallow the adversary from trivially distinguishing the two games.

The second main change in the proof is to switch the decryption keys from keys corresponding to $\boldsymbol{y}_1^{j,0} \| \ldots \| \boldsymbol{y}_n^{j,0}$ to keys corresponding to $\boldsymbol{y}_1^{j,1} \| \ldots \| \boldsymbol{y}_n^{j,1}$ for every $j \in [Q_f]$. This in turn requires a hybrid argument over all decryption keys, changing one key at a time. To switch the ρ'th key, we use the selective simulation security of the underlying $\mathcal{FE}$ to embed the value $\langle \boldsymbol{x}_i^{j,1}, \boldsymbol{y}_i^{\rho,\beta} \rangle + \langle \boldsymbol{r}^\rho, \boldsymbol{z}_i \rangle$ in the ciphertexts ct_i^j, for all slots $i \in [n]$ and all $j \in [Q_i]$. Next, we use the $\mathcal{D}_k$-MDDH assumption to argue that $[\langle \boldsymbol{r}^\rho, \boldsymbol{z}_i \rangle]_T$ is indistinguishable from a uniform random value and thus perfectly hides $\langle \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,\beta} \rangle$ for the first ciphertext of each slot: ct_i^1. For all the other remaining $\langle \boldsymbol{x}_i^{j,1}, \boldsymbol{y}_i^{\rho,\beta} \rangle$, for $j \in [Q_i]$, $j > 1$, we use the fact that $\langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho.0} \rangle = \langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho.1} \rangle$, as implied by the game's restrictions.

Proof of Theorem 4. We proceed via a series of Games $\mathsf{G}_0, \mathsf{G}_1, \mathsf{G}_{1.\rho}$, for $\rho \in [Q_f + 1]$, described in Figure 14. An overview is provided in Figure 13. Let $\mathcal{A}$ be a PPT adversary, and $\lambda \in \mathbb{N}$ be the security parameter. We denote by $\mathsf{Adv}_{\mathsf{G}_i}(\mathcal{A})$ the advantage of $\mathcal{A}$ in game G_i.

G_0: is the experiment **many-SEL-wFH-IND**$_0^{\mathcal{MIFE}}$ (see Definition 6).

G_1: we replace the inner encryption of $\boldsymbol{x}_i^{j,0}$ by encryptions of $\boldsymbol{x}_i^{j,1}$, for all $i \in [n]$, $j \in [Q_i]$, using the many-SEL-IND security of $\mathcal{MIFE}'$. This is possible due to the weak function-hiding

New function-hiding scheme $\mathcal{MIFE}$

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:
$(\mathsf{mpk}', \mathsf{msk}') \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})$
$\forall i \in [n] : (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{\ell,X,Y})$
$\mathsf{mpk} := \left(\{\mathsf{mpk}_i\}_{i\in[n]}, \mathsf{mpk}'\right)$, $\mathsf{msk} := \left(\{\mathsf{msk}_i\}_{i\in[n]}, \mathsf{msk}'\right)$
return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i)}$:
$[\mathsf{ct}_i^{\mathsf{in}}]_1 := \mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i)$
$\mathsf{ct}_i^{\mathsf{out}} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in}}]_1)$
return $\mathsf{ct}_i^{\mathsf{out}}$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n)}$:
$(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T) \leftarrow \mathsf{KeyGen}'\left(\mathsf{msk}', \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n\right)$
$\forall i \in [n] : \mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\mathsf{Enc}(\mathsf{msk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$
$\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \left(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T\right)$
return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n}$

$\underline{\mathsf{Dec}\left(\left(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T\right), \{\mathsf{ct}_i^{\mathsf{out}}\}_{i\in[n]}\right)}$:
$\forall i \in [n] : [a_i]_T \leftarrow \mathcal{FE}.\mathsf{Dec}(\mathsf{ct}_i^{\mathsf{out}}, \mathsf{sk}_i^{\mathsf{out}})$
return the discrete log of $\left(\prod_{i=1}^n [a_i]_T\right) / [z]_T$

Fig. 12. Many-SEL-wFH-IND secure, private-key, multi-input, FE for the class $\mathcal{F}_n^{m,X,Y}$. Here $\mathcal{FE} := (\mathcal{FE}.\mathsf{Setup}, \mathcal{FE}.\mathsf{Enc}, \mathcal{FE}.\mathsf{KeyGen}, \mathcal{FE}.\mathsf{Dec})$ is a one-SEL-SIM secure, single-input FE for $\mathcal{F}_1^{\ell,X,Y}$, where by ℓ we denote the output size of Enc' and KeyGen', and $\mathcal{MIFE}' := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ is the many-AD-IND secure, multi-input FE from Figure 11.

constraint, which states in particular that $\sum_{i=1}^n \langle \boldsymbol{x}_i^{j_i,0}, \boldsymbol{y}_i^{j_f,0} \rangle = \sum_{i=1}^n \langle \boldsymbol{x}_i^{j_i,1}, \boldsymbol{y}_i^{j_f,0} \rangle$, for all indices $j_i \in [Q_i], j_f \in [Q_f]$.

$\mathsf{G}_{1.\rho}$: for the first $\rho - 1$ queries to KeyGen, we replace inner secret key $\mathsf{KeyGen}'\left(\mathsf{msk}', \boldsymbol{y}_1^0 \| \cdots \| \boldsymbol{y}_n^0\right)$, by $\mathsf{KeyGen}'\left(\mathsf{msk}', \boldsymbol{y}_1^1 \| \cdots \| \boldsymbol{y}_n^1\right)$. Note that G_1 is the same as $\mathsf{G}_{1.1}$, and $\mathsf{G}_{1.Q_f+1}$ is the same as $\textbf{many-SEL-wFH-IND}_1^{\mathcal{MIFE}}$.

We prove $\mathsf{G}_0 \approx_c \mathsf{G}_1$ in Lemma 4, and $\mathsf{G}_{1.\rho} \approx_c \mathsf{G}_{1.\rho+1}$ for all $\rho \in [Q_f]$ in Lemma 5. □

Lemma 4 (G_0 to G_1). *There exists a PPT adversary $\mathcal{B}_1$ such that*

$$\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\textit{many-SEL-IND}}_{\mathcal{MIFE}', \mathcal{B}_1}(\lambda).$$

Proof. In order to show that we can switch $\boldsymbol{x}_i^{j,0}$ to $\boldsymbol{x}_i^{j,1}$, we rely on the security of the underlying $\mathcal{MIFE}'$ scheme. Intuitively, adding an additional layer of encryption on the decryption keys $\mathsf{sk}_i^{\mathsf{in}}$ cannot invalidate the security of the underlying $\mathcal{MIFE}'$.

More formally, we design an adversary $\mathcal{B}_1$ against the many-SEL-IND security of $\mathcal{MIFE}'$. Adversary $\mathcal{B}_1$ draws public and secret keys for the outer encryption layer and then uses its own experiment to simulate either G_0 or G_1. We describe adversary $\mathcal{B}_1$ in Figure 15 and give a textual description here.

Simulation of master public key mpk. Since the game is selective, the adversary $\mathcal{B}_1$ first gets the challenges $\{\boldsymbol{x}_i^{j,b}\}_{i\in[n], j\in[Q_i], b\in\{0,1\}}$ from $\mathcal{A}$, and it sends them to its experiment $\text{many-SEL-IND}_\beta^{\mathcal{MIFE}'}$. Then, $\mathcal{B}_1$ receives the public key mpk' of the $\mathcal{MIFE}'$ scheme. To construct the full

Game	$[\mathsf{ct}_i^{\mathsf{in},k}]_1$	$[\mathsf{sk}_i^{\mathsf{in},j}]_2$	justification/remark
G_0	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{k,0})$	$\mathsf{KeyGen}'(\mathsf{msk}', \boldsymbol{y}_1^{j,0} \| \dots \| \boldsymbol{y}_n^{j,0})$	many-SEL-wFH-IND$_0$ security game
G_1	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{k,1})$	$\mathsf{KeyGen}'(\mathsf{msk}', \boldsymbol{y}_1^{j,0} \| \dots \| \boldsymbol{y}_n^{j,0})$	many-SEL-IND of $\mathcal{MIFE}'$
$\mathrm{G}_{1.\rho}$	$\mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{k,1})$	$\mathsf{KeyGen}'(\mathsf{msk}', \boldsymbol{y}_1^{j,1} \| \dots \| \boldsymbol{y}_n^{j,1})$, for $j < \rho$ $\mathsf{KeyGen}'(\mathsf{msk}', \boldsymbol{y}_1^{j,0} \| \dots \| \boldsymbol{y}_n^{j,0})$, for $j \geq \rho$	Lemma 5

Fig. 13. An overview of the games used in the proof of Theorem 4. By $[\mathsf{ct}_i^{\mathsf{in},k}]_1$ and $[\mathsf{sk}_i^{\mathsf{in},j}]_2$ we denote the k^{th} ciphertext and the j^{th} decryption key of the inner scheme $\mathcal{MIFE}'$.

G_0, $\boxed{\mathsf{G}_1, \boxed{\mathsf{G}_{1.\rho}}}$, for $\rho \in [Q_f + 1]$:

$\{\boldsymbol{x}_i^{j,\beta}\}_{i\in[n],j\in[Q_i],\beta\in\{0,1\}}, \{\boldsymbol{y}_i^{j,\beta}\}_{i\in[n],j\in[Q_f],\beta\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n^{m,X,Y})$
$(\mathsf{mpk}', \mathsf{msk}') \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})$
$\forall i \in [n] : (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{\ell,X,Y})$
$\mathsf{mpk} := \big(\{\mathsf{mpk}_i\}_{i\in[n]}, \mathsf{mpk}'\big)$, $\mathsf{msk} := \big(\{\mathsf{msk}_i\}_{i\in[n]}, \mathsf{msk}'\big)$
$\forall i \in [n]$, $j \in [Q_i]$:
$[\mathsf{ct}_i^{\mathsf{in},j}]_1 := \mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,0})$, $\boxed{[\mathsf{ct}_i^{\mathsf{in},j}]_1 := \mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,1})}$
$\mathsf{ct}_i^{\mathsf{out},j} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in},j}]_1)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}\big(\mathsf{mpk}, \{\mathsf{ct}_i^{\mathsf{out},j}\}_{i\in[n],j\in[Q_i]}\big)$
Output: α.

$\mathsf{KeyGen}\big(\mathsf{msk}, (\boldsymbol{y}_1^{j,\beta} \| \cdots \| \boldsymbol{y}_n^{j,\beta})_{\beta\in\{0,1\}}\big)$:
$\big(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T\big) \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^0 \| \cdots \| \boldsymbol{y}_n^0\big)$
$\boxed{\text{If } j < \rho\text{: } \big(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T\big) \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^1 \| \cdots \| \boldsymbol{y}_n^1\big)}$
$\mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\mathsf{Enc}(\mathsf{msk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$
$\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \big(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T\big)$
return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n}$

Fig. 14. Games for the proof of Theorem 4. In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame.

public key, it draws $(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\kappa, \mathcal{F}_1^{\ell,X,Y})$, for all slots $i \in [n]$, independently. It then sets $\mathsf{mpk} := \{\mathsf{mpk}_i\}_{i\in[n]} \cup \{\mathsf{mpk}'\}$ and returns mpk to adversary $\mathcal{A}$.

Simulation of the challenge ciphertexts. The adversary $\mathcal{B}_1$ receives $[\mathsf{ct}_i^{\mathsf{in},j}]_1$ from the encryption oracle of the experiment many-SEL-IND$_\beta^{\mathcal{MIFE}'}$, for all $i \in [n]$. This corresponds to encryptions of either $\boldsymbol{x}_i^{j,\beta}$, for $\beta = 0$ or 1. Since it knows msk_i, it computes $\mathsf{ct}_i^{\mathsf{out},j} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in},j}]_1)$ for all $i \in [n]$ and returns $\{\mathsf{ct}_i^{\mathsf{out},j}\}_{i\in[n]}$ to $\mathcal{A}$.

Simulation of $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$. On every secret key query $(\boldsymbol{y}_1^b \| \dots \| \boldsymbol{y}_n^b)_{b\in\{0,1\}}$, adversary $\mathcal{B}_1$ queries the KeyGen' oracle of the experiment many-SEL-IND$_\beta^{\mathcal{MIFE}'}$ on $\boldsymbol{y}_1^0 \| \dots \| \boldsymbol{y}_n^0$. It obtains $\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T$. Finally, it computes $\mathsf{sk}_i^{\mathsf{out}} := \mathcal{FE}.\mathsf{Enc}(\mathsf{mpk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$ and returns $(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T)$ to $\mathcal{A}$.

□

$\mathcal{B}_1\left(1^\lambda, \{\boldsymbol{x}_i^{j,b}\}_{i\in[n],j\in[Q_i],b\in\{0,1\}}, \{\boldsymbol{y}_i^{j,b}\}_{i\in[n],j\in[Q_f],b\in\{0,1\}}\right)$:

-Simulation of the master public key mpk:
sends $\{\boldsymbol{x}_i^{j,b}\}_{i\in[n],j\in[Q_i],b\in\{0,1\}}$ to Exp
$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{\ell,X,Y})$
it receives mpk' from Exp
$\mathsf{mpk} := \{\mathsf{mpk}_i\}_{i\in[n]} \cup \{\mathsf{mpk}'\}$
return mpk

-Simulation of ciphertexts:
receives $[\mathsf{ct}_i^{\mathsf{in},j}]_1$ from Exp
$\mathsf{ct}_i^{\mathsf{out},j} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in},j}]_1)$
return $\mathsf{ct}_i^{\mathsf{out},j}$

-Simulation of $\mathsf{KeyGen}\big(\mathsf{msk}, (\boldsymbol{y}_1^{j,b}\| \cdots \|\boldsymbol{y}_n^{j,b})_{b\in\{0,1\}}\big)$**:**
receive $(\{[\mathsf{sk}_i^{\mathsf{in},j}]_2\}_{i\in[n]}, [z]_T)$ from Exp
$\mathsf{sk}_i^{\mathsf{out},j} := \mathcal{FE}.\mathsf{Enc}(\mathsf{mpk}_i, [\mathsf{sk}_i^{\mathsf{in},j}]_2)$
return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \big(\{\mathsf{sk}_i^{\mathsf{out},j}\}_{i\in[n]}, [z]_T\big)$

Fig. 15. Adversary $\mathcal{B}_1$ distinguishes between two cases, case 1: $\mathbf{Exp} = \mathbf{many\text{-}SEL\text{-}IND}_0^{\mathcal{MIFE}'}$, $[\mathsf{ct}_i^{\mathsf{in},j}]_1 := \mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,0})$ and case 2: $\mathbf{Exp} = \mathbf{many\text{-}SEL\text{-}IND}_1^{\mathcal{MIFE}'}$, $[\mathsf{ct}_i^{\mathsf{in},j}]_1 := \mathsf{Enc}'(\mathsf{msk}', i, \boldsymbol{x}_i^{j,1})$. KeyGen queries are answered identically in both cases without knowing msk' by calling the KeyGen' oracle of Exp and encrypting the $\mathsf{sk}_i^{\mathsf{in},j}$ responses under mpk_i.

Lemma 5 ($\mathsf{G}_{1.\rho}$ **to** $\mathsf{G}_{1.\rho+1}$)**.** *For all $\rho \in [Q_f]$, there exist PPT adversaries $\mathcal{B}_\rho$ and $\mathcal{B}'_\rho$ such that*

$$\mathsf{Adv}_{\mathsf{G}_{1.\rho}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1.\rho+1}}(\mathcal{A}) \leq 2n \cdot \mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{B}_\rho}(\lambda,) + 2 \cdot \mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_1,\mathcal{B}'_\rho}(\lambda) + \tfrac{2k}{p}.$$

Proof of Lemma 5. We proceed via a series of Games $\mathsf{H}_\rho, \mathsf{H}_{\rho.\beta}, \mathsf{H}'_{\rho.\beta}$, for $\rho \in [Q_f+1]$, and $\beta \in \{0,1\}$, described in Figure 17. Note that H_ρ is $\mathsf{G}_{1.\rho}$. We prove that:

$$\mathsf{G}_{1,\rho} \equiv \mathsf{H}_\rho \approx_c \mathsf{H}_{\rho.0} \approx_c \mathsf{H}'_{\rho.0} \equiv \mathsf{H}'_{\rho.1} \approx_c \mathsf{H}_{\rho.1} \approx_c \mathsf{H}_{\rho+1} \equiv \mathsf{G}_{1,\rho+1}.$$

H_ρ to $\mathsf{H}_{\rho.0}$: we replace $(\mathcal{FE}.\mathsf{Setup}, \mathcal{FE}.\mathsf{KeyGen}, \mathcal{FE}.\mathsf{Enc})$ by the efficient simulator $(\mathcal{FE}.\widetilde{\mathsf{Setup}}, \mathcal{FE}.\widetilde{\mathsf{KeyGen}}, \mathcal{FE}.\widetilde{\mathsf{Enc}})$, using the one-SEL-SIM security of the single-input FE, via a hybrid argument across all slots $i \in [n]$.

Lemma 6 ($\mathsf{H}_\rho \approx_c \mathsf{H}_{\rho.0}$**:**)**.** *There exists a PPT adversary $\mathcal{B}_{1.\rho}$ such that*

$$\mathsf{Adv}_{\mathsf{H}_\rho}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}_{\rho.0}}(\mathcal{A}) \leq n \cdot \mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{B}_{1.\rho}}(\lambda)$$

Proof. We use a hybrid argument over all slots $t \in [n]$. That is, we define $\mathsf{H}_{\rho.1.t}$ for $t \in [n]$, where the for the first t slots, $(\mathcal{FE}.\mathsf{Setup}, \mathcal{FE}.\mathsf{KeyGen}, \mathcal{FE}.\mathsf{Enc})$ is replaced with $(\mathcal{FE}.\widetilde{\mathsf{Setup}}, \mathcal{FE}.\widetilde{\mathsf{KeyGen}}, \mathcal{FE}.\widetilde{\mathsf{Enc}})$ (the games are described in Figure 16).

Note that $\mathsf{H}_{\rho.0.0}$ is H_ρ and $\mathsf{H}_{\rho.0.n}$ is $\mathsf{H}_{\rho.0}$. Using the one-SEL-SIM security of $\mathcal{FE}$, we have

$$\mathsf{Adv}_{\mathsf{H}_{\rho.0.t}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}_{\rho.0.t+1}}(\mathcal{A}) \leq \mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{B}_{\rho.t}}(\lambda).$$

The reduction showing adversary $\mathcal{B}_{\rho.t}$ is rather straightforward and is omitted. The idea is that $\mathcal{B}_{\rho.t}$ generates all the keys involved in the scheme, except for $(\mathsf{mpk}_t, \mathsf{msk}_t)$. For the t-th one, it plugs the one it receives from the one-SEL-SIM$^{\mathcal{FE}}$ experiment, i.e., either mpk_t or $\widetilde{\mathsf{mpk}}_t$; this allows it to perfectly simulate the view of $\mathcal{A}$. □

$\underline{\mathsf{H}_{\rho.0.t} \text{ for } \rho \in [Q_f+1],\, t \in [n+1]}$:

$\{\boldsymbol{x}_i^{j,b}\}_{i\in[n],j\in[Q_i],b\in\{0,1\}}, \{\boldsymbol{y}_i^{j_f,b}\}_{i\in[n],j_f\in[Q_f],b\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n^{m,X,Y})$

$\left(\mathsf{mpk}' := \left(\{\mathsf{mpk}_i'\}_{i\in[n]}\right), \mathsf{msk}' := \left(\{\mathsf{msk}_i', \boldsymbol{z}_i\}_{i\in[n]}\right)\right) \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{F}_n^{m,X,Y})$

$\underline{\forall i < t :} \left(\widetilde{\mathsf{mpk}}_i, \widetilde{\mathsf{msk}}_i\right) \leftarrow \mathcal{FE}.\widetilde{\mathsf{Setup}}(1^\lambda, \mathcal{F}_1^{\ell,X,Y})$

$\quad \forall j \in [Q_i]\colon \mathsf{ct}_i^{\mathsf{out},j} := \mathcal{FE}.\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}_i, [\mathsf{ct}_i^{\mathsf{in},j}]_1, [\langle \boldsymbol{x}_i^{j,1}, \boldsymbol{y}_i^\beta\rangle + \langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_1)$

$\underline{\forall i \geq t :} (\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathcal{FE}.\mathsf{Setup}(1^\lambda, \mathcal{F}_1^{\ell,X,Y})$

$\quad \forall j \in [Q_i]\colon \mathsf{ct}_i^{\mathsf{out},j} := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i, [\mathsf{ct}_i^{\mathsf{in},j}]_1)$

$\mathsf{mpk} := \left(\{\widetilde{\mathsf{mpk}}_i\}_{i\leq t} \cup \{\mathsf{mpk}_i\}_{i>t}, \mathsf{mpk}'\right)$

$\mathsf{msk} := \left(\{\widetilde{\mathsf{msk}}_i\}_{i\leq t} \cup \{\mathsf{msk}_i\}_{i>t}, \mathsf{msk}'\right)$

$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, \{\mathsf{ct}_i^{\mathsf{out},j}\}_{i\in[n],j\in[Q_i]}\right)$

Output: α.

$\underline{\mathsf{KeyGen}\big(\mathsf{msk}, (\boldsymbol{y}_1^{j,b}\| \cdots \|\boldsymbol{y}_n^{j,b})_{b\in\{0,1\}} \in (\mathbb{Z}^m)^n\big)\colon}$

On the j'th query, $j < \rho$:

$\begin{pmatrix} \{[\mathsf{sk}_i^{\mathsf{in}}]_2 := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i', \boldsymbol{y}_i^1\|\boldsymbol{r}^j)\}_{i\in[n]} \\ [z]_T := [\sum_i \langle \boldsymbol{z}_i, \boldsymbol{r}^j\rangle]_T \end{pmatrix} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^1\| \cdots \|\boldsymbol{y}_n^1\big)$

$\forall i \in [n] : \mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\mathsf{Enc}(\mathsf{msk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$

On the j'th query, $j > \rho$:

$\begin{pmatrix} \{[\mathsf{sk}_i^{\mathsf{in}}]_2 := \mathcal{FE}.\mathsf{KeyGen}(\mathsf{msk}_i', \boldsymbol{y}_i^0\|\boldsymbol{r}^j)\}_{i\in[n]} \\ [z]_T := [\sum_i \langle \boldsymbol{z}_i, \boldsymbol{r}^j\rangle]_T \end{pmatrix} \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^0\| \cdots \|\boldsymbol{y}_n^0\big)$

$\forall i \in [n] : \mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\mathsf{Enc}(\mathsf{msk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$

On the j'th query, $j = \rho$:

for $i < t$: $\mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}_i)$

for $i \geq t$: $[\mathsf{sk}_i^{\mathsf{in}}]_2 \leftarrow \mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^0\| \cdots \|\boldsymbol{y}_n^0\big)$, $\mathsf{sk}_i^{\mathsf{out}} \leftarrow \mathcal{FE}.\mathsf{Enc}(\mathsf{msk}_i, [\mathsf{sk}_i^{\mathsf{in}}]_2)$

Return $\mathsf{sk}_{\boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n} := \left(\{\mathsf{sk}_i^{\mathsf{out}}\}_{i\in[n]}, [z]_T\right)$

Fig. 16. Games for the proof of Lemma 6.

$\mathsf{H}_{\rho.\beta} \approx_c \mathsf{H}'_{\rho.\beta}$, for all $\beta \in \{0,1\}$: we replace $\{[\boldsymbol{z}_i]_1, [\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_1\}_{i\in[n]}$ by $\{[\boldsymbol{z}_i]_1, [\widetilde{z}_i]_1\}_{i\in[n]}$, where $\boldsymbol{r}^\rho \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ is the randomness picked by KeyGen on its ρ'th query, $\boldsymbol{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. This is justified using the $\mathcal{D}_k$-MDDH assumption in $\mathbb{G}_1$.

Lemma 7 (From $\mathsf{H}_{\rho.\beta}$ to $\mathsf{H}'_{\rho.\beta}$). *For all $\rho \in [Q_f]$, $\beta \in \{0,1\}$, there exists a PPT adversary $\mathcal{B}_{\rho.\beta}$ such that*

$$\mathsf{Adv}_{\mathsf{H}_{\rho.\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}'_{\rho.\beta}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathcal{D}_k\text{-mddh}}_{\mathbb{G}_1,\mathcal{B}_{\rho.\beta}}(\lambda) + \tfrac{k}{p}.$$

Proof. Precisely we first build an adversary $\mathcal{B}'_{\rho,\beta}$ against $\mathcal{U}_{n,k}$-MDDH, and then the proof is obtained by applying Lemma 1 ($\mathcal{D}_k$-MDDH $\Rightarrow$ $\mathcal{U}_{n,k}$-MDDH). Such adversary can be defined quite straightforwardly so that it provides either $\{[\boldsymbol{z}_i]_1, [\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_1\}_{i\in[n]}$ or $\{[\boldsymbol{z}_i]_1, [\widetilde{z}_i]_1\}_{i\in[n]}$ to $\mathcal{A}$ (thus simulating either $\mathsf{H}_{\rho.\beta}$ or $\mathsf{H}'_{\rho.\beta}$ respectively) according to the distribution of its own input. The only observation is that by the $\mathcal{U}_{n,k}$-MDDH definition, $\left(\boldsymbol{z}_1\| \cdots \|\boldsymbol{z}_n\right)^\top$ is uniformly random over full-rank matrices in $\mathbb{Z}_p^{n\times k}$, whereas in the simulated experiment it is supposed to be uniformly random over $\mathbb{Z}_p^{n\times k}$. However, these two distributions are k/p-close (assuming $n > k$). □

Lemma 8 ($\mathsf{H}'_{\rho.0} \equiv \mathsf{H}'_{\rho.1}$)**.**

$$\mathsf{Adv}_{\mathsf{H}'_{\rho.0}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{H}'_{\rho.1}}(\mathcal{A})$$

Proof. We argue that these games are the same, using the change of variable: $\widetilde{z}_i \to \widetilde{z}_i - \langle \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,b} \rangle$, and the fact that $\sum_i \langle \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,b} \rangle$ is independent of $b \in \{0,1\}$. The fact that our games are selective allows us to perform this change of variables without changing the distribution of the game, since $\widetilde{z}_i$ is independent of the $\boldsymbol{x}$ or $\boldsymbol{y}$ values. Moreover, for all $i \in [n]$, $j \in [Q_i]$, $\langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,b} \rangle$ is independent of $b \in \{0,1\}$, by the definition of the security game. □

Lemma 9 (From $\mathsf{H}_{\rho.1}$ to $\mathsf{H}_{\rho+1}$). *There exists a PPT adversary $\mathcal{B}''_{\rho+1}$ such that*

$$\mathsf{Adv}_{\mathsf{H}_{\rho.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}_{\rho+1}}(\mathcal{A}) \leq n \cdot \mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{B}''_{\rho+1}}(\lambda)$$

This transition is symmetric to the transition between H_ρ and $\mathsf{H}_{\rho.0}$, upper-bounded in Lemma 6. Namely, we replace $(\mathcal{FE}.\widetilde{\mathsf{Setup}}, \mathcal{FE}.\widetilde{\mathsf{KeyGen}}, \mathcal{FE}.\widetilde{\mathsf{Enc}})$ by $(\mathcal{FE}.\mathsf{Setup}, \mathcal{FE}.\mathsf{KeyGen}, \mathcal{FE}.\mathsf{Enc})$, using the one-SEL-SIM security of the single-input FE, via a hybrid argument across all slots $i \in [n]$.

Summing up, obtain $|\mathsf{Adv}_{\mathsf{G}_{1.\rho}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1.\rho+1}}(\mathcal{A})| \leq 2n \cdot \mathsf{Adv}^{\textit{one-SEL-SIM}}_{\mathcal{FE},\mathcal{B}_{1.\rho}}(\lambda,) + 2 \cdot \mathbf{Adv}^{\mathcal{D}_k\text{-mddh}}_{\mathbb{G}_1,\mathcal{B}'_{1.\rho}}(\lambda) + \frac{2k}{p}$, which implies that $\mathsf{G}_{1.\rho} \approx_c \mathsf{G}_{1.\rho+1}$. □

5.2 Adaptively-secure Multi-input Function-Hiding FE for Inner Product

In this section, we prove that if we instantiate the construction described in Figure 12 (Section 5), with the many-AD-IND-secure, single-input FE from [ALS16], we obtain an adaptively secure function-hiding MIFE. Specifically, we consider the generalized version of single-input FE, as described in [AGRW17] (recalled in Figure 7). For completeness, we present this new MIFE instantiation in Figure 18. Proving adaptive security for our construction in a generic way would require the underlying $\mathcal{FE}$ to achieve strong security notions, such as one-AD-SIM (which is not achieved by any known scheme). We overcome this issue, managing to prove adaptive security of our concrete MIFE in Figure 12, using non-generic techniques inspired by [AGRW17].

Theorem 5 (many-AD-IND-wFH security). *If the $\mathcal{D}_k$-MDDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$, then the multi-input FE for $\mathcal{F}_n^{m,X,Y}$ described in Figure 18 is many-AD-IND-wFH-secure.*

Proof overview. Similarly to the selective-security proof presented in Section 5.1, we prove weakly-function-hiding. This is sufficient, since it can be transformed generically into a fully function-hiding MIFE by using techniques from [LV16] (see Appendix B for more details).

To prove weak function-hiding we proceed in two stages. First, we switch from $\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i^{j,0})$ to $\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i^{j,1})$ for all slots $i \in [n]$ and all queries $j \in [Q_i]$ simultaneously, using the many-AD-IND security of $\mathcal{MIFE}'$ (the underlying MIFE from [AGRW17]). For completeness, we also give a concrete description of $\mathcal{MIFE}'$ in Figure 24, Appendix C.

Secondly, we use a hybrid argument over all Q_f queried keys, switching them one by one from $\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y}_1^0 \| \cdots \| \boldsymbol{y}_n^0)$ to $\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y}_1^1 \| \cdots \| \boldsymbol{y}_n^1)$. To switch the ρ'th key, we use the security of $\mathcal{FE}$ in a non-generic way. Structurally, we do a proof similar to the selective one of the previous section. In order to apply complexity leveraging, we first do all the computational steps. Afterwards, only at some particular transition in the proof (transition from $\mathsf{H}''_{\rho.0}$ to $\mathsf{H}''_{\rho.1}$ in the proof

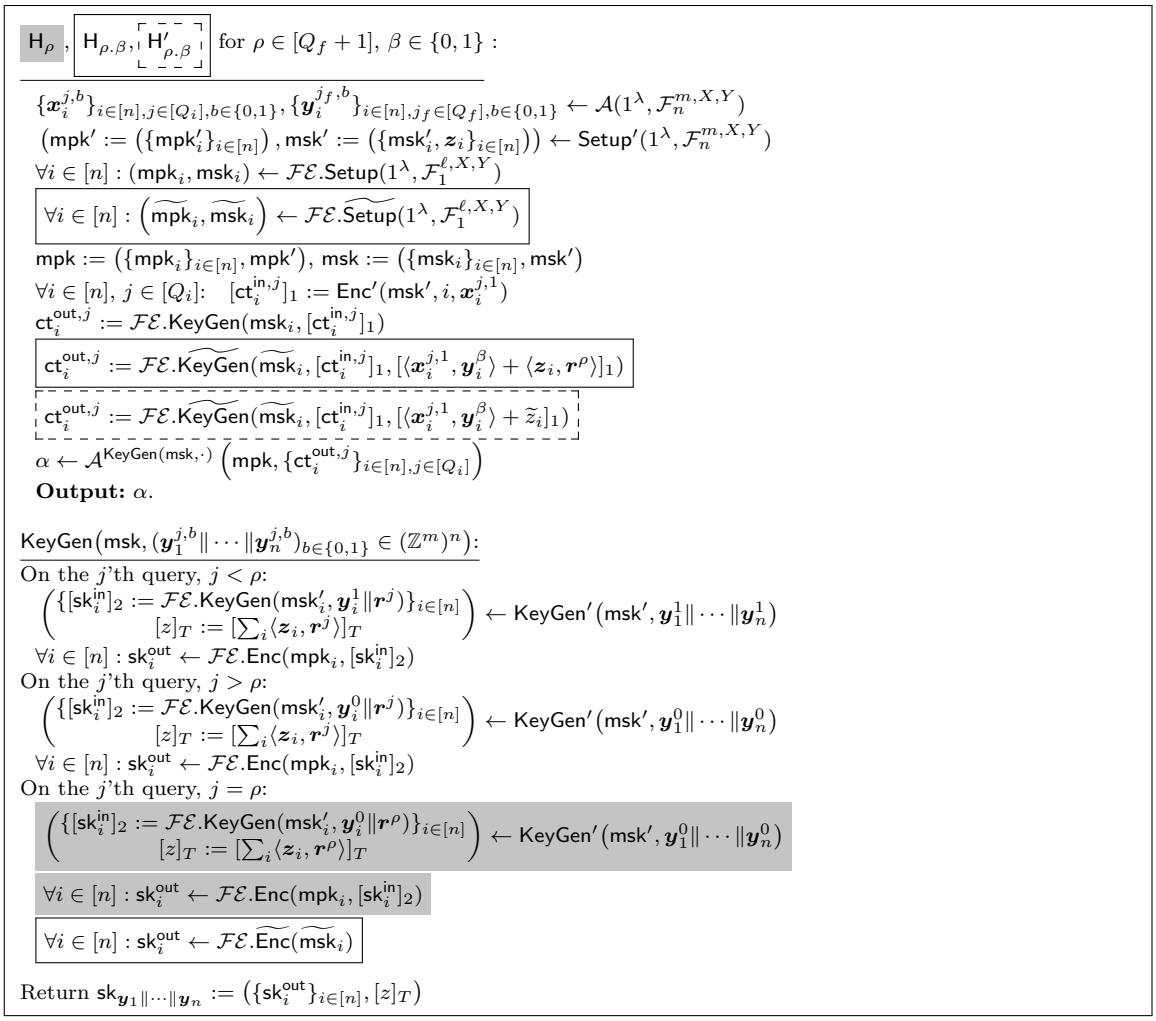

Fig. 17. Games for the proof of Lemma 5. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

of Lemma 14), we use complexity leveraging, and we simulate the selective proof arguments. This multiplies the security loss by an exponential factor. We can do so here because this particular transition is perfect: the exponential term is multiplied by a zero advantage.

Although this proof strategy shares similarities with the adaptive security proof the MIFE in [AGRW17], our proof has some crucial differences: mainly, the role of the keys and ciphertexts in our proof is switched. Since the multi-input model is asymmetric with respect to the ciphertexts and decryption keys (only ciphertexts can be mixed-and-matched), this results in a different proof strategy.

Proof of Theorem 5. We proceed via a series of Games $\mathsf{G}_0, \mathsf{G}_1, \mathsf{G}_{1.\rho}$, for $\rho \in [Q_f+1]$, described in Figure 20. Let $\mathcal{A}$ be a PPT adversary, and $\lambda \in \mathbb{N}$ be the security parameter. For all games G_i, we denote by $\mathsf{Adv}_{\mathsf{G}_i}(\mathcal{A})$ the advantage of $\mathcal{A}$ in G_i.

G_0: is the experiment **many-AD-wFH-IND**$_0^{\mathcal{MIFE}}$ (see Definition 5).

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:

$\mathcal{PG} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda), \mathbf{A}_1, \mathbf{B}_1, \ldots, \mathbf{A}_n, \mathbf{B}_n \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{U}_1, \ldots, \mathbf{U}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(k+m)\times(k+1)}$
$\mathbf{V}_1, \ldots, \mathbf{V}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(2k+m+1)\times(k+1)}$, $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\mathsf{mpk} := \mathcal{PG}$, $\mathsf{msk} := \{\mathbf{A}_i, \mathbf{B}_i, \mathbf{U}_i, \mathbf{V}_i, \boldsymbol{z}_i\}_{i\in[n]}$
return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i \in \mathbb{Z}_p^m)}$:

$\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{c}_i := \mathbf{A}_i\boldsymbol{s}_i$, $\boldsymbol{c}'_i := \begin{pmatrix}\boldsymbol{x}_i\\ \boldsymbol{z}_i\end{pmatrix} + \mathbf{U}_i\mathbf{A}_i\boldsymbol{s}_i$, $\boldsymbol{c}''_i := \begin{pmatrix}\boldsymbol{c}_i\\ \boldsymbol{c}'_i\end{pmatrix}^\top \mathbf{V}_i$
return $([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1, [\boldsymbol{c}''_i]_1)$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y}_1\|\cdots\|\boldsymbol{y}_n \in (\mathbb{Z}_p^m)^n)}$:

$\boldsymbol{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}\rangle$
$\forall i \in [n] : \boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{d}_i := \mathbf{B}_i\boldsymbol{t}_i$, $\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix}\boldsymbol{y}_i\\ \boldsymbol{r}\end{pmatrix} \\ \begin{pmatrix}\boldsymbol{y}_i\\ \boldsymbol{r}\end{pmatrix}\end{pmatrix} + \mathbf{V}_i\mathbf{B}_i\boldsymbol{t}_i$
return $\left(\{[\boldsymbol{d}_i]_2, [\boldsymbol{d}'_i]_2\}_{i\in[n]}, [z]_T\right)$

$\underline{\mathsf{Dec}\left(\left(\{[\boldsymbol{d}_i]_2, [\boldsymbol{d}'_i]_2\}_{i\in[n]}, [z]_T\right), \{[\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1, [\boldsymbol{c}''_i]_1\}_{i\in[n]}\right)}$:

$out \leftarrow \left(\prod_i \left(e\left(\begin{bmatrix}\boldsymbol{c}_i\\ \boldsymbol{c}'_i\end{bmatrix}_1^\top, [\boldsymbol{d}'_i]_2\right) / e\left([\boldsymbol{c}''_i]_1, [\boldsymbol{d}_i]_2\right)\right)\right) / [z]_T$
return discrete log of out

Fig. 18. Many-AD-IND-wFH secure, multi-input FE scheme for the class $\mathcal{F}_n^{m,X,Y}$ (self-contained description).

G_1: we replace the inner encryption of $\boldsymbol{x}_i^{j,0}$ by encryptions of $\boldsymbol{x}_i^{j,1}$, for all $i \in [n]$, $j \in [Q_i]$, using the many-AD-IND security of $\mathcal{MIFE} := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$.

$\mathsf{G}_{1.\rho}$: for the first $\rho - 1$ queries to KeyGen, we replace the inner secret key $\mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^0\| \cdots \|\boldsymbol{y}_n^0\big)$, by $\mathsf{KeyGen}'\big(\mathsf{msk}', \boldsymbol{y}_1^1\| \cdots \|\boldsymbol{y}_n^1\big)$. Note that Game_1 is the same as $\mathrm{Game}_{1.1}$, and $\mathrm{Game}_{1.Q_f+1}$ is the same as $\textbf{many-AD-wFH-IND}_1^{\mathcal{MIFE}}$.

We prove $\mathsf{G}_0 \approx_c \mathsf{G}_1$ in Lemma 10, and $\mathsf{G}_{1.\rho} \approx_c \mathsf{G}_{1.\rho+1}$ for all $\rho \in [Q_f]$ in Lemma 11. □

Lemma 10 (G_0 to G_1). *There exists a PPT adversary $\mathcal{B}_0$ such that*

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\textit{many-AD-IND}}_{\mathcal{MIFE},\mathcal{B}_0}(\lambda).$$

Proof. This proof is very similar to the proof of Lemma 4. We design an adversary $\mathcal{B}_0$ against the many-AD-IND security of the scheme from Figure 24 (which is many-AD-IND-secure by Theorem 6).

Note that the vectors $[\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1$ output by Enc (see Figure 18) exactly correspond to a ciphertext for the MIFE scheme from [AGRW17]. Thus, using the many-AD-IND security of the latter, we can switch encryption of $x_i^{j,0}$ into encryption of $x_i^{j,1}$ for all $i \in [n]$, $j \in [Q_i]$. We now describe how $\mathcal{B}_1$ simulates $\mathcal{A}$'s view.

Simulation of the public key. Given the $\mathsf{mpk} := \mathcal{PG}$, $\mathcal{B}_0$ draws matrices $\mathbf{B}_i \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(2k+m+1)\times(k+1)}$, and forwards the public key $\mathcal{PG}$ to $\mathcal{A}$.

Simulation of the ciphertexts. Consider the case when $\mathcal{A}$ makes a plaintext query $(\boldsymbol{x}_i^{j,0}, \boldsymbol{x}_i^{j,1})$ to $\mathcal{B}_1$. Adversary $\mathcal{B}_0$ forwards the queries to the many-AD-IND challenger, which replies with $([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1)$. $\mathcal{B}_1$ simply computes $[\boldsymbol{c}''_i]_1 := \begin{bmatrix} \boldsymbol{c}_i \\ \boldsymbol{c}'_i \end{bmatrix}_1^\top \cdot \mathbf{V}_i$ and sends $([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1, [\boldsymbol{c}''_i]_1)$ to $\mathcal{A}$.

Simulation of the decryption keys. Upon receiving a query $(\boldsymbol{y}_1^{j,0} \| \ldots \| \boldsymbol{y}_n^{j,0}, \boldsymbol{y}_1^{j,1} \| \ldots \| \boldsymbol{y}_n^{j,1})$, $\mathcal{B}_1$ forwards it to the many-AD-IND challenger. It obtains back $(\{[\mathsf{sk}_i^{\mathsf{in}}]_2\}_{i\in[n]}, [z]_T)$. It then draws $\boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ and computes $[\boldsymbol{d}_i]_2 = [\mathbf{B}_i \boldsymbol{t}_i]_2$ and $[\boldsymbol{d}'_i] := [\mathsf{sk}_i^{\mathsf{in}}]_2 \cdot [\mathbf{V}_i \mathbf{B}_i \boldsymbol{t}_i]_2$. It sends $([\boldsymbol{d}_i]_2, [\boldsymbol{d}'_i]_2, [z]_T)$ back to $\mathcal{A}$.

□

Lemma 11 ($\mathsf{G}_{1.\rho}$ to $\mathsf{G}_{1.\rho+1}$). *For all $\rho \in [Q_f]$, there exists PPT adversary $\mathcal{B}_\rho$ such that:*

$$|\mathsf{Adv}_{\mathsf{G}_{1.\rho}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1.\rho+1}}(\mathcal{A})| \leq (2n+2)\mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_2,\mathcal{B}_\rho}(\lambda) + \tfrac{2(n+k)}{p}.$$

Proof of Lemma 11. We proceed via a series of games H_ρ, $\mathsf{H}_{\rho.\beta}$, $\mathsf{H}'_{\rho.\beta}$, $\mathsf{H}''_{\rho.\beta}$ for $\rho \in [Q_f + 1]$ and $\beta \in \{0, 1\}$ described in Figure 21. Note that H_ρ is $\mathsf{G}_{1.\rho}$. We summarize our sequence of hybrids between $\mathsf{G}_{1,\rho}$ and $\mathsf{G}_{1,\rho+1}$ in Figure 19.

$$\boxed{\mathsf{G}_{1,\rho} \equiv \mathsf{H}_\rho \approx_c \mathsf{H}_{\rho,0} \equiv \mathsf{H}'_{\rho,0} \approx_c \mathsf{H}''_{\rho,0} \equiv \mathsf{H}''_{\rho,1} \approx_c \mathsf{H}'_{\rho,1} \equiv \mathsf{H}_{\rho,1} \approx_c \mathsf{H}_{\rho+1} \equiv \mathsf{G}_{1,\rho+1}}$$

Fig. 19. Summary of the sequence of hybrids in the proof of Lemma 11

$\mathsf{H}_\rho \approx_c \mathsf{H}_{\rho.0}$: we change the distribution of the vectors $[\boldsymbol{c}_i]_1$ computed by $\mathsf{Enc}(i, \cdot, \cdot)$, for all slots $i \in [n]$, using the $\mathcal{D}_k$-MDDH assumption (this is similar to Lemma 7).

Lemma 12 (H_ρ to $\mathsf{H}_{\rho.0}$). *For all $\rho \in [Q_f]$, there exists a PPT adversary $\mathcal{B}_\rho$ such that:*

$$\mathsf{Adv}_{\mathsf{H}_\rho}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}_{\rho.0}}(\mathcal{A}) \leq n \cdot \mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_2,\mathcal{B}_\rho}(\lambda) + \tfrac{n}{p}.$$

Proof of Lemma 10. Here, we switch $([\mathbf{B}_i]_2, [\mathbf{B}_i \boldsymbol{s}_i]_2)$ computed by $\mathsf{Enc}(i, \cdot, \cdot)$ to $([\mathbf{B}_i]_2, [\mathbf{B}_i \boldsymbol{s}_i + \boxed{\boldsymbol{u}_i}]_2)$, where $\mathbf{B}_i \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \mathsf{span}(\mathbf{B}_i)$, and $\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. We do so for all input slots $i \in [n]$, using a hybrid argument. This change is justified by the facts that:

- By the $\mathcal{D}_k$-MDDH assumption, we can switch $([\mathbf{B}_i]_2, [\mathbf{B}_i \boldsymbol{s}_i]_2)$ to $([\mathbf{B}_i]_2, [\mathbf{B}_i \boldsymbol{s}_i + \boxed{\boldsymbol{u}_i}]_2)$, where $\mathbf{B}_i \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$.
- The uniform distribution over $\mathbb{Z}_p^{k+1}$ and $\mathbb{Z}_p^{k+1} \setminus \mathsf{span}(\mathbf{B}_i)$ are $\frac{1}{p}$-close, for all $\mathbf{B}_i$ of rank k.

Combining these facts, we obtain a PPT adversary $\mathcal{B}_\rho$ such that

$$\mathsf{Adv}_{\mathsf{H}_\rho}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}_{\rho.0}}(\mathcal{A}) \leq n \cdot \mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathbb{G}_2,\mathcal{B}_\rho}(\lambda) + \tfrac{n}{p}.$$

□

$\mathsf{H}_{\rho.\beta} \equiv \mathsf{H}'_{\rho.\beta}$, for all $\beta \in \{0,1\}$: here, for all slots $i \in [n]$, we replace the way the vector $[\boldsymbol{d}'_i]_1$ is computed by $\mathsf{KeyGen}(\mathsf{msk},\cdot)$ on its ρ'th query, using an information theoretic argument. Looking ahead, we want to make it possible to simulate the adversary's view only knowing $[\boldsymbol{r}^\rho]_1, [\boldsymbol{z}_i]_1, [\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_T$, and not $[\boldsymbol{r}^\rho]_2$. This is in order to apply $\mathcal{D}_k$-MDDH in $\mathsf{H}'_{\rho.\beta}$ in the next transition.

Lemma 13 ($\mathsf{H}_{\rho.\beta}$ to $\mathsf{H}'_{\rho.\beta}$, for all $\beta \in \{0,1\}$). *For all $\beta \in \{0,1\}$, $\rho \in [Q_f]$,*

$$\mathsf{Adv}_{\mathsf{H}_{\rho.\beta}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{H}'_{\rho.\beta}}(\mathcal{A}).$$

Proof of Lemma 13. We argue that $\mathsf{H}_{\rho.\beta}$ and $\mathsf{H}'_{\rho.\beta}$ are the same, using the fact that for all $i \in [n]$, $\mathbf{B}_i \in \mathbb{Z}_p^{(k+1)\times k}$, $\boldsymbol{b}_i^\perp \in \mathsf{orth}(\mathbf{B}_i)$, $\mathbf{U}_i \in \mathbb{Z}_p^{(k+m)\times(k+1)}$ and all $\boldsymbol{r}^\rho \in \mathbb{Z}_p^k$, the following distributions are identical:

$$\mathbf{V}_i \text{ and } \mathbf{V}_i - \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \mathbf{0} \\ \boldsymbol{r}^\rho \end{pmatrix} \\ \begin{pmatrix} \mathbf{0} \\ \boldsymbol{r}^\rho \end{pmatrix} \end{pmatrix} (\boldsymbol{b}_i^\perp)^\top,$$

where $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(2k+m+1)\times(k+1)}$. This way, we obtain

$$\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{\rho,b} \\ \mathbf{0} \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{\rho,b} \\ \mathbf{0} \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i \quad \text{and} \quad \boldsymbol{c}''_i := \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{c}'_i \end{pmatrix}^\top \mathbf{V}_i + \langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle \cdot (\boldsymbol{b}_i^\perp)^\top,$$

as in $\mathsf{H}'_{\rho.\beta}$. □

$\mathsf{H}'_{\rho.\beta} \approx_c \mathsf{H}''_{\rho.\beta}$: we replace $\{[\boldsymbol{z}_i]_1, [\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_1\}_{i\in[n]}$ to $\{[\boldsymbol{z}_i]_1, [\widetilde{z}_i]_1\}_{i\in[n]}$, where $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, using the $\mathcal{D}_k$-MDDH assumption.

Lemma 14 (From $\mathsf{H}'_{\rho.\beta}$ to $\mathsf{H}''_{\rho.\beta}$). *For all $\beta \in \{0,1\}$, $\rho \in [Q_f]$, there exists a PPT adversary $\mathcal{B}_{\rho,\beta}$ such that:*

$$\mathsf{Adv}_{\mathsf{H}'_{\rho.\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}''_{\rho.\beta}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathcal{D}_k\text{-}\mathsf{mddh}}_{\mathsf{PGGen},\mathcal{B}_{\rho.\beta}}(\lambda) + \tfrac{k}{p}.$$

Proof. This proof is very similar to the proof of Lemma 7. We design an adversary $\mathcal{B}'_{\rho,\beta}$ against $\mathcal{U}_{n,k}$-MDDH, such that $\mathsf{Adv}_{\mathsf{H}'_{\rho.\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{H}''_{\rho.\beta}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathcal{U}_{n,k}\text{-}\mathsf{mddh}}_{\mathsf{PGGen},\mathbb{G}_1,\mathcal{B}_{\rho,\beta}}(\lambda)$. The lemma then follows from Lemma 1 ($\mathcal{D}_k$-MDDH $\Rightarrow \mathcal{U}_{n,k}$-MDDH).

More precisely, note that $(\boldsymbol{z}_1 \| \cdots \| \boldsymbol{z}_n)^\top$ is uniformly random over $\mathbb{Z}_p^{n\times k}$, which is $\frac{k}{p}$-close to uniformly random over full-rank n times k matrices over $\mathbb{Z}_p$ (assuming $n > k$). Thus, using the $\mathcal{U}_{n,k}$-MDDH assumption, we can switch $\{[\boldsymbol{z}_i]_1, [\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho\rangle]_1\}_{i\in[n]}$ to $\{[\boldsymbol{z}_i]_1, [\widetilde{z}_i]_1\}_{i\in[n]}$, where $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. □

$\mathsf{H}''_{\rho.0} \equiv \mathsf{H}''_{\rho.1}$: we go the selective variant of $\mathsf{H}''_{\rho.0}$, called $\mathsf{H}^{\star}_{\rho.0}$, via complexity leveraging (i.e. guessing the challenge), then use the following change of variables:

$$\mathbf{V}_i \to \mathbf{V}_i + \begin{pmatrix} -\mathbf{U}_i^{\top} \begin{pmatrix} \boldsymbol{y}_i^{\rho,1} - \boldsymbol{y}_i^{\rho,0} \\ \mathbf{0} \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{\rho,1} - \boldsymbol{y}_i^{\rho,0} \\ \mathbf{0} \end{pmatrix} \end{pmatrix} (\boldsymbol{b}_i^{\perp})^{\top},$$

and

$\widetilde{z}_i \to \widetilde{z}_i - \langle \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,1} - \boldsymbol{y}_i^{\rho,0} \rangle$ to switch to $\mathsf{H}^{\star}_{\rho.1}$. We use the fact that $\sum_i \langle \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,1} - \boldsymbol{y}_i^{\rho,0} \rangle = 0$, and that for all $i \in [n]$, $j \in [Q_i]$, we have: $\langle \boldsymbol{x}_i^{j,1} - \boldsymbol{x}_i^{1,1}, \boldsymbol{y}_i^{\rho,1} - \boldsymbol{y}_i^{\rho,0} \rangle = 0$, by definition of the security game. We switch back to the adaptive variant, $\mathsf{H}''_{\rho.1}$, "unguessing", which incurs an exponential security loss, multiplied by zero.

Summing up, we have $\mathsf{G}_{1.\rho} \approx_c \mathsf{G}_{1.\rho+1}$ (see Figure 19 for a summary of the hybrids).

□

G_0, $\boxed{\mathsf{G}_1, \boxed{\mathsf{G}_{1.\rho}}}$ for $\rho \in [Q_f + 1]$:

$\mathcal{PG} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^{\lambda}), \mathbf{A}_1, \mathbf{B}_1, \ldots, \mathbf{A}_n, \mathbf{B}_n \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{U}_1, \ldots, \mathbf{U}_n, \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(k+m)\times(k+1)}$, $\mathbf{V}_1, \ldots, \mathbf{V}_n, \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(2k+m+1)\times(k+1)}$
$\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\mathsf{mpk} := \mathcal{PG}$
$\mathsf{msk} := \{\mathbf{A}_i, \mathbf{B}_i, \mathbf{U}_i, \mathbf{V}_i, \boldsymbol{z}_i\}_{i\in[n]}$
$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot),\mathsf{Enc}(\cdot,\cdot,\cdot)}(\mathsf{mpk})$
Output: α.

$\mathsf{Enc}\big(i, (\boldsymbol{x}_i^{j,b})_{b\in\{0,1\}}\big)$:
$\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{c}_i := \mathbf{A}_i \boldsymbol{s}_i$
$\boldsymbol{c}'_i := \begin{pmatrix} \boldsymbol{x}_i^{j,0} \\ \boldsymbol{z}_i \end{pmatrix} + \mathbf{U}_i \mathbf{A}_i \boldsymbol{s}_i$, $\boxed{\boldsymbol{c}'_i := \begin{pmatrix} \boldsymbol{x}_i^{j,1} \\ \boldsymbol{z}_i \end{pmatrix} + \mathbf{U}_i \mathbf{A}_i \boldsymbol{s}_i}$
$\boldsymbol{c}''_i := \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{c}'_i \end{pmatrix}^{\top} \mathbf{V}_i$
Return $\mathsf{ct}_i := ([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1, [\boldsymbol{c}''_i]_1)$

$\mathsf{KeyGen}\big(\mathsf{msk}, (\boldsymbol{y}_1^{j,b} \| \cdots \| \boldsymbol{y}_n^{j,b})_{b\in\{0,1\}}\big)$:
$\boldsymbol{r}^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}^j \rangle$
$\forall i \in [n] : \boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{d}_i := \mathbf{B}_i \boldsymbol{t}_i$
$\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^{\top} \begin{pmatrix} \boldsymbol{y}_i^{j,0} \\ \boldsymbol{r}^j \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{j,0} \\ \boldsymbol{r}^j \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \mathbf{B}_i \boldsymbol{t}_i$, $\boxed{\text{If } j < \rho\text{: } \boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^{\top} \begin{pmatrix} \boldsymbol{y}_i^{j,1} \\ \boldsymbol{r}^j \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{j,1} \\ \boldsymbol{r}^j \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \mathbf{B}_i \boldsymbol{t}_i}$
return $\big(\{[\boldsymbol{d}_i]_2, [\boldsymbol{d}'_i]_2\}_{i\in[n]}, [z]_T\big)$

Fig. 20. G_0, $\mathsf{G}_{1.\rho}$ for $\rho \in [Q_f + 1]$, for the proof of . In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame.

H_ρ, $\boxed{\mathsf{H}_{\rho.\beta}, \boxed{\mathsf{H}'_{\rho.\beta}, \mathsf{H}''_{\rho.\beta}}}$ for $\rho \in [Q_f + 1]$ and $\beta \in \{0,1\}$:

$\mathcal{PG} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda), \mathbf{A}_1, \mathbf{B}_1, \ldots, \mathbf{A}_n, \mathbf{B}_n \leftarrow_{\mathrm{R}} \mathcal{D}_k,\ \mathbf{U}_1, \ldots, \mathbf{U}_n, \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(k+m)\times(k+1)},\ \mathbf{V}_1, \ldots, \mathbf{V}_n, \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(2k+m+1)\times(k+1)}$
$\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\boxed{\forall i \in [n] : \boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \mathsf{span}(\mathbf{B}_i),\ \boldsymbol{b}_i^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B}_i) \text{ s.t. } \langle \boldsymbol{u}_i, \boldsymbol{b}_i^\perp \rangle = 1}$
$\widetilde{z}_1, \ldots, \widetilde{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p$
$\mathsf{mpk} := \mathcal{PG}$
$\mathsf{msk} := \{\mathbf{A}_i, \mathbf{B}_i, \mathbf{U}_i, \mathbf{V}_i, \boldsymbol{z}_i\}_{i\in[n]}$
$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot),\mathsf{Enc}(\cdot,\cdot,\cdot)}(\mathsf{mpk})$
Output: α.

$\mathsf{Enc}\big(i, (\boldsymbol{x}_i^{j,b})_{b\in\{0,1\}}\big)$:
$\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ \boldsymbol{c}_i := \mathbf{A}_i \boldsymbol{s}_i$
$\boldsymbol{c}'_i := \begin{pmatrix} \boldsymbol{x}_i^{j,1} \\ \boldsymbol{z}_i \end{pmatrix} + \mathbf{U}_i \mathbf{A}_i \boldsymbol{s}_i$
$\boldsymbol{c}''_i := \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{c}'_i \end{pmatrix}^\top \mathbf{V}_i - \boxed{\langle \boldsymbol{z}_i, \boldsymbol{r}^\rho \rangle \cdot (\boldsymbol{b}_i^\perp)^\top}$, $\boldsymbol{c}''_i := \begin{pmatrix} \boldsymbol{c}_i \\ \boldsymbol{c}'_i \end{pmatrix}^\top \mathbf{V}_i - \widetilde{z}_i \cdot (\boldsymbol{b}_i^\perp)^\top$
Return $\mathsf{ct}_i^j := ([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1, [\boldsymbol{c}''_i]_1)$

$\mathsf{KeyGen}\big(\mathsf{msk}, (\boldsymbol{y}_1^{j,b} \| \cdots \| \boldsymbol{y}_n^{j,b})_{b\in\{0,1\}}\big)$:

If $j < \rho$:
$\boldsymbol{r}^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}^j \rangle$
$\forall i \in [n] : \boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ \boldsymbol{d}_i := \mathbf{B}_i \boldsymbol{t}_i$
$\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{j,1} \\ \boldsymbol{r}^j \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{j,1} \\ \boldsymbol{r}^j \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i$

If $j > \rho$:
$\boldsymbol{r}^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}^j \rangle$
$\forall i \in [n] : \boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ \boldsymbol{d}_i := \mathbf{B}_i \boldsymbol{t}_i$
$\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{j,0} \\ \boldsymbol{r}^j \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{j,0} \\ \boldsymbol{r}^j \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i$

If $j = \rho$:
$\boldsymbol{r}^\rho \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}^\rho \rangle$, $z := \sum_{i\in[n]} \widetilde{z}_i$
$\forall i \in [n] : \boldsymbol{t}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k,\ \boldsymbol{d}_i := \mathbf{B}_i \boldsymbol{t}_i + \boxed{\boldsymbol{u}_i}$
$\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{\rho,0} \\ \boldsymbol{r}^\rho \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{\rho,0} \\ \boldsymbol{r}^\rho \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i$, $\boxed{\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{\rho,\beta} \\ \boldsymbol{r}^\rho \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{\rho,\beta} \\ \boldsymbol{r}^\rho \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i}$, $\boldsymbol{d}'_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix} \boldsymbol{y}_i^{\rho,\beta} \\ \boldsymbol{0} \end{pmatrix} \\ \begin{pmatrix} \boldsymbol{y}_i^{\rho,\beta} \\ \boldsymbol{0} \end{pmatrix} \end{pmatrix} + \mathbf{V}_i \boldsymbol{d}_i$
Return $\big(\{[\boldsymbol{d}_i]_2, [\boldsymbol{d}'_i]_2\}_{i\in[n]}, [z]_T\big)$

Fig. 21. H_ρ, $\mathsf{H}_{\rho.\beta}$, $\mathsf{H}'_{\rho.\beta}$, $\mathsf{H}''_{\rho.\beta}$ for $\rho \in [Q_f + 1]$ and $\beta \in \{0,1\}$, for the proof of Lemma 11. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

Acknowledgments. Michel Abdalla was supported in part by SAFEcrypto (H2020 ICT-644729) and by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement 780108 (FENTEC). Dario Fiore was partially supported by the Spanish Ministry of Economy under project references TIN2015-70713-R (DEDETIS), RTC-2016-4930-7 (DataMantium), and by

the Madrid Regional Government under project N-Greens (ref. S2013/ICE-2731). Romain Gay was partially supported by a Google PhD Fellowship in Privacy and Security and by the ERC Project aSCEND (H2020 639554). Bogdan Ursu was partially supported by ANR-14-CE28-0003 (Project EnBiD) and by the ERC Project PREP-CRYPTO (H2020 724307).

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. (Pages 2 and 3.)

ABDP16. Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011, 2016. http://eprint.iacr.org/2016/011. (Pages 2 and 3.)

AGRW17. Michel Abdalla, Romain Gay, Mariana Raykova, and Hoeteck Wee. Multi-input inner-product functional encryption from pairings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 601–626. Springer, Heidelberg, April / May 2017. (Pages 1, 2, 3, 4, 6, 7, 14, 16, 18, 19, 22, 23, 25, 30, 31, 32, 39, and 40.)

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. (Page 1.)

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. (Pages 2, 3, 13, 14, 18, 19, 20, 21, 23, and 30.)

BGJS15. Saikrishna Badrinarayanan, Divya Gupta, Abhishek Jain, and Amit Sahai. Multi-input functional encryption for unbounded arity functions. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 27–51. Springer, Heidelberg, November / December 2015. (Page 1.)

BKS16. Zvika Brakerski, Ilan Komargodski, and Gil Segev. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 852–880. Springer, Heidelberg, May 2016. (Page 1.)

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. (Page 1.)

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. (Pages 4 and 5.)

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. (Pages 2, 9, and 10.)

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. (Pages 1 and 5.)

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. (Pages 3, 4, 22, and 23.)

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. (Pages 4, 8, 24, 30, and 39.)

O'N10. Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/2010/556. (Page 1.)

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. (Page 1.)

Wee17. Hoeteck Wee. Attribute-hiding predicate encryption in bilinear groups, revisited. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 206–233. Springer, Heidelberg, November 2017. (Pages 3 and 23.)

A One-SEL-SIM security of the one-time MIFE

Here we give the definition of one-SEL-SIM security for a multi-input functional encryption scheme. Then we show that the one-time MIFE scheme $\mathcal{MIFE}^{\mathsf{ot}}$ described in Figure 1 satisfies this security notion, which implies the weaker one-SEL-IND security notion presented in Definition 3.

Definition 12 (one-SEL-SIM-secure FE). *A multi-input functional encryption $\mathcal{MIFE}$ for function $\mathcal{F}_n$ is one-SEL-SIM-secure if there exist PPT simulator algorithms*[15] $(\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ *such that for every PPT (stateful) adversary $\mathcal{A}$ and every $\lambda \in \mathbb{N}$, the following two distributions are computationally indistinguishable:*

Experiment $\mathbf{REAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(\mathbf{1}^\lambda, \mathcal{A})$:	*Experiment* $\mathbf{IDEAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(\mathbf{1}^\lambda, \mathcal{A})$:
$\{x_i\}_{i\in[n]} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$	$\{x_i\}_{i\in[n]} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n)$	$(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{msk}}) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \mathcal{F}_n)$
For all $i \in [n]$, $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{msk}, i, x_i)$	*For all* $i \in [n]$, $\mathsf{ct}_i \leftarrow \widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}, i)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{mpk}, \{\mathsf{ct}_i\}_{i\in[n]})$	$\alpha \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\widetilde{\mathsf{mpk}}, \{\mathsf{ct}_i\}_{i\in[n]})$
Output: α	***Output:*** α

The oracle $\mathcal{O}(\cdot)$ in the ideal experiment above is given access to another oracle that, given $f \in \mathcal{F}_n$, returns $f(x_1, \ldots, x_n)$, and then $\mathcal{O}(\cdot)$ returns $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, f, f(x_1, \ldots, x_n))$.

For every stateful adversary $\mathcal{A}$, we define its advantage as

$$\mathsf{Adv}^{\mathit{one\text{-}SEL\text{-}SIM}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = \left|\Pr\left[\mathbf{REAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(\mathbf{1}^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathbf{IDEAL}^{\mathcal{MIFE}}_{\mathbf{SEL}}(\mathbf{1}^\lambda, \mathcal{A}) = 1\right]\right|,$$

and we require that for every PPT $\mathcal{A}$, there exists a negligible function negl *such that for all* $\lambda \in \mathbb{N}$, $\mathsf{Adv}^{\mathit{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

Lemma 15. *The one-time MIFE described in Figure 1 is one-SEL-SIM secure. Namely, for any adversary $\mathcal{A}$,* $\mathsf{Adv}^{\mathit{one\text{-}SEL\text{-}SIM}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{A}}(\lambda) = 0$.

Proof. Let us define the simulator algorithms as follows: $\widetilde{\mathsf{Setup}} = \mathsf{Setup}^{\mathsf{ot}}$, $\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}, i) = \boldsymbol{u}_i$, and $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \boldsymbol{y}, aux) \to \mathsf{sk}_{\boldsymbol{y}} := z$, where $z := \sum_{i\in[n]} \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle - aux \bmod L$.

Next, we use the fact that for all $\{\boldsymbol{x}_i \in \mathbb{Z}_L^m\}_{i\in[\ell]}$, the following distributions are identical: $\{\boldsymbol{u}_i \bmod L\}_{i\in[n]}$ and $\{\boldsymbol{u}_i - \boldsymbol{x}_i \bmod L\}_{i\in[n]}$, with $\boldsymbol{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_L^m$. Note that the independence of the $\boldsymbol{x}_i$ from the $\boldsymbol{u}_i$ is only true in the selective security game. Therefore, using this fact we can rewrite the experiment $\mathbf{REAL}^{\mathcal{MIFE}^{\mathsf{ot}}}_{SEL}(1^\lambda, \mathcal{B})$ as $\mathbf{HYB}(1^\lambda, \mathcal{B})$ in Figure 22. This hybrid is also identical to the experiment $\mathbf{IDEAL}^{\mathcal{MIFE}^{\mathsf{ot}}}_{SEL}(1^\lambda, \mathcal{B})$ when executed with our simulator algorithms. In particular, observe that the oracle $\mathcal{O}_H$ corresponds to the oracle $\mathcal{O}(\cdot)$ which returns $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \boldsymbol{y}, \langle \boldsymbol{x}, \boldsymbol{y} \rangle)$ for every queried $\boldsymbol{y}$. Thus, we obtain: $\mathsf{Adv}^{\mathit{one\text{-}SEL\text{-}SIM}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}}(\lambda) = 0$. □

[15] That is, $\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}}$ correspond respectively to the simulated $\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}$.

$\underline{\mathbf{HYB}(1^\lambda, \mathcal{B})}$:	
$\{x_i\}_{i\in[n]} \leftarrow \mathcal{B}(1^\lambda, \mathcal{F}^m_{L,n})$	
For all $i \in [n]$,	
$\quad \boldsymbol{u}_i \leftarrow_R \mathbb{Z}_L^m$	
$\quad \mathsf{ct}_i \leftarrow \boldsymbol{u}_i \bmod L$	
$\alpha \leftarrow \mathcal{B}^{\mathcal{O}_H(\cdot)}(\{\mathsf{ct}_i\}_{i\in[\ell]})$	
Output α	
	$\underline{\mathcal{O}_H(\boldsymbol{y})}$:
	$z := \sum_{i\in[n]} \langle \boldsymbol{u}_i, \boldsymbol{y}_i \rangle - \langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod L$
	Return z

Fig. 22. Experiment for the proof of Lemma 15.

B From Weak to Full Function-Hiding

In [LV16], the authors propose a simple transformation for turning a weak function-hiding FE scheme into a full-fledged function hiding one. In this section, we show that the same transformation is applicable in the multi-input case. For brevity, we use $\boldsymbol{x}_i \| \mathbf{0}$ to denote that $\boldsymbol{x}_i \in \mathbb{Z}_p^m$ is padded with m trailing zero. The new scheme consists in using the original $\mathcal{MIFE}$ scheme to encrypt $\boldsymbol{x}'_i = \boldsymbol{x}_i \| \mathbf{0}$ instead of $\boldsymbol{x}_i$, for every slot $i \in [n]$. Likewise, instead of generating keys for $\boldsymbol{y} = \boldsymbol{y}_1 \| \dots \| \boldsymbol{y}_n$, the keys will be generated for $\boldsymbol{y}' = (\boldsymbol{y}_1 \| \mathbf{0}) \| \dots \| (\boldsymbol{y}_n \| \mathbf{0})$. This does not change the result of the decryption, since:

$$\sum_i \langle \boldsymbol{x}_i \| \mathbf{0}, \boldsymbol{y}_i \| \mathbf{0} \rangle = \sum_i \langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle.$$

Security is justified via a hybrid argument over Games 0 to 3. For every $i \in \{0, 1, 2\}$, the advantage of an adversary $\mathcal{A}$ distinguishing between Game_i and Game_{i+1} is negligible based on the weak function-hiding security property. The transitions are more precisely described in Figure 23.

Game	$\boldsymbol{x}'_i$	$\boldsymbol{y}'_i$	justification/remark
0	$\boldsymbol{x}_i^0 \| \mathbf{0}$	$\boldsymbol{y}_i^0 \| \mathbf{0}$	many-zzz-FH_0 security game, zzz $\in$ {SEL, AD}
1	$\boldsymbol{x}_i^0 \| \boldsymbol{x}_i^1$	$\mathbf{0} \| \boldsymbol{y}_i^1$	weak function-hiding of the underlying scheme
2	$\boldsymbol{x}_i^1 \| \boldsymbol{x}_i^1$	$\boldsymbol{y}_i^1 \| \mathbf{0}$	weak function-hiding of the underlying scheme
3	$\boldsymbol{x}_i^1 \| \mathbf{0}$	$\boldsymbol{y}_i^1 \| \mathbf{0}$	weak function-hiding of the underlying scheme many-zzz-FH_1 security game, zzz $\in$ {SEL, AD}

Fig. 23. Sequence of games for transforming a weak multi-input function-hiding inner-product encryption scheme into a full function-hiding one.

Notice that for every $i \in \{0, 1, 2\}$, Game_i can be argued computationally indistinguishable from Game_{i+1} based only on the weak function-hiding property. For example, for Game_0 and Game_1, $\langle \boldsymbol{x}'^0, \boldsymbol{y}'^0 \rangle = \langle \boldsymbol{x}'^1, \boldsymbol{y}'^0 \rangle = \langle \boldsymbol{x}'^1, \boldsymbol{y}'^1 \rangle$, which implies that Game_0 and Game_1 are computationally indistinguishable only due to the weak function-hiding of the underlying scheme. Applying the same argument for $i \in \{1, 2\}$, we get that the scheme using paddings is fully function-hiding.

C Appendix - Adaptive (Non-Function-Hiding) Multi-Input Scheme

In this section we recall the adaptively-secure multi-input encryption scheme from [AGRW17], where it was proven to be many-AD-IND-secure. This ensures that encryptions of $\boldsymbol{x}_i^{j,0}$ are indistinguishable

from encryptions of $\boldsymbol{x}_i^{j,1}$, for all slots $i \in [n]$, and queries $j \in [Q_i]$ (in the presence of the constraints from Definition 2, which avoid trivial attacks). Nevertheless, the scheme is not function hiding, since the $\boldsymbol{y}$ values are encoded directly in the exponent (in $\mathbb{Z}_p$). In order to prove the many-AD-FH security of our new scheme (see Figure 18), we will need to use the following result, proven in [AGRW17]:

Theorem 6 (many-AD-IND security). *Suppose the $\mathcal{D}_k$-MDDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$. Then, the multi-input FE in Figure 24 is one-AD-IND-secure.*

$\underline{\mathsf{Setup}(1^\lambda, \mathcal{F}_n^{m,X,Y})}$:
$\mathcal{PG} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$
$\mathbf{A}_1 \ldots, \mathbf{A}_n \leftarrow_{\mathrm{R}} \mathcal{D}_k$
$\mathbf{U}_1, \ldots, \mathbf{U}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{(k+m)\times(k+1)}$
$\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\mathsf{mpk} := \mathcal{PG}$
$\mathsf{msk} := \{\mathbf{A}_i, \mathbf{U}_i, \boldsymbol{z}_i\}_{i\in[n]}$
return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Dec}\Big(\big(\{[\mathsf{sk}_i]_2\}_{i\in[n]}, [z]_T\big), \{[\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1\}_{i\in[n]}\Big)}$:
$out \leftarrow \left(\prod_i e\left(\begin{bmatrix}\boldsymbol{c}_i\\ \boldsymbol{c}'_i\end{bmatrix}_1^\top, [\mathsf{sk}_i]_2\right)\right) / [z]_T$
return discrete log of out

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{y}_1 \| \cdots \| \boldsymbol{y}_n \in (\mathbb{Z}_p^m)^n)}$:
$\boldsymbol{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $z := \langle \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n, \boldsymbol{r}\rangle$
$\forall i \in [n]$:
$\mathsf{sk}_i := \begin{pmatrix} -\mathbf{U}_i^\top \begin{pmatrix}\boldsymbol{y}_i\\ \boldsymbol{r}\end{pmatrix} \\ \begin{pmatrix}\boldsymbol{y}_i\\ \boldsymbol{r}\end{pmatrix}\end{pmatrix}$
return $\big(\{[\mathsf{sk}_i]_2\}_{i\in[n]}, [z]_T\big)$

$\underline{\mathsf{Enc}(\mathsf{msk}, i, \boldsymbol{x}_i \in \mathbb{Z}_p^m)}$:
$\boldsymbol{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\boldsymbol{c}_i := \mathbf{A}_i \boldsymbol{s}_i$
$\boldsymbol{c}'_i := \begin{pmatrix}\boldsymbol{x}_i\\ \boldsymbol{z}_i\end{pmatrix} + \mathbf{U}_i \mathbf{A}_i \boldsymbol{s}_i$
return $([\boldsymbol{c}_i]_1, [\boldsymbol{c}'_i]_1)$

Fig. 24. Many-AD-IND secure, private-key, multi-input FE scheme for the class $\mathcal{F}_n^{m,X,Y}$ (self-contained description from [AGRW17]).