Multi-Input Inner-Product Functional Encryption from Pairings

Michel Abdalla[1,⋆], Mariana Raykova[2,⋆⋆], and Hoeteck Wee[1,⋆⋆⋆]

[1] ENS and PSL Research University, Paris, France
{michel.abdalla,hoeteck.wee}@ens.fr
https://crypto.di.ens.fr/
[2] Yale University and SRI,
mariana.raykova@yale.edu

Abstract. We present a multi-input functional encryption scheme (MIFE) for the inner product functionality based on the k-Linear assumption in prime-order bilinear groups. Our construction works for any polynomial number of encryption slots and achieves security against unbounded collusion, while relying on standard polynomial hardness assumptions. This is the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions, as well as the first to achieve polynomial security loss for a super-constant number of slots under falsifiable assumptions. Prior works required stronger non-standard assumptions such as indistinguishability obfuscation or multi-linear maps.

1 Introduction

In a functional encryption (FE) scheme [22, 11], an authority can generate restricted decryption keys that allow users to learn specific functions of the encrypted messages and nothing else. That is, each FE decryption key sk_f is associated with a function f and decrypting a ciphertext $\mathsf{Enc}(x)$ with sk_f results in $f(x)$. Multi-input functional encryption (MIFE) introduced by Goldwasser et al. [18] is a generalization of functional encryption to the setting of multi-input functions. A MIFE scheme has several encryption slots and each decryption key sk_f for a multi-input function f decrypts jointly ciphertexts $\mathsf{Enc}(x_1)$, ..., $\mathsf{Enc}(x_n)$ for all slots to obtain $f(x_1, \ldots, x_n)$ without revealing anything more about the encrypted messages. The MIFE functionality provides the capability to encrypt independently messages for different slots. This facilitates scenarios where information, which will be processed jointly during decryption, becomes available at different points of time or is provided by different parties. MIFE has many applications related to computation and data-mining over encrypted data coming from multiple sources, which include examples such as executing search queries over encrypted data, processing encrypted streaming data, non-interactive differentially private data releases, multi-client delegation of computation, order-revealing encryption [18, 10]. The security requirement for FE and MIFE is that the decryption keys are resilient to collusion attacks, namely any group of users holding different decryption keys learns nothing about the underlying messages beyond what each of them could individually learn.

We now have several constructions of MIFE schemes, which can be broadly classified as follows: (i) feasibility results for general circuits [18, 6, 5, 12], and (ii) constructions for specific functionalities, notably comparison, which corresponds to order-revealing encryption [10]. Unfortunately, all of

⋆ CNRS. Supported in part by SAFEcrypto (H2020 ICT-644729).
⋆⋆ Supported by NSF grant CNS-1421102 and DARPA SafeWare W911NF-15-C-0236.
⋆⋆⋆ CNRS and Columbia University. Supported in part by the ERC Project aSCEND (H2020 639554) and NSF Award CNS-1445424.

these constructions rely on indistinguishability obfuscation, single-input FE for circuits, or multilinear maps [16, 15], which we do not know how to instantiate under standard and well-understood cryptographic assumptions.[3]

1.1 Our Contributions

In this work, we present the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions with polynomial security loss, and for any polynomial number of slots and secure against unbounded collusions. The functionality we consider is that of "bounded-norm" multi-input inner product: each function is specified by a collection of n vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$, takes as input n vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and outputs

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

We require that the $\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{y}_1, \ldots, \mathbf{y}_n$ have bounded norm, and inner product is computed over the integers. The functionality is a natural generalization of single-input inner product functionality introduced by Abdalla et. al [1], and studied in [1, 7, 13, 4, 2], and captures several useful computations arising in the context of data-mining. A summary of our results and prior works on single-input inner product is shown in Fig. 1.

Our results. We consider MIFE for the inner product in both the public-key and the private-key setting. Recall that for standard public-key encryption and single-input FE, a public-key scheme is itself also a private-key scheme. However, this is not the case for MIFE because the ideal functionality in the public-key setting inherently leaks more information to the adversary; in particular, given an encryption of an unknown x_1 and a secret key for f for a public-key scheme, the adversary immediately obtains an oracle for $f(x_1, \cdot, \ldots)$.

Going back to inner product, our first observation is that in the public-key setting, we have a general construction of a MIFE scheme from a single-input FE scheme: run n independent copies of the single-input scheme; use the i'th copy to encrypt $\mathbf{x}_i$ in the i'th slot; and the new secret key is the collection of the n secret keys corresponding to each of $\mathbf{y}_1, \ldots, \mathbf{y}_n$. Decryption recovers $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle, \ldots \langle \mathbf{x}_n, \mathbf{y}_n \rangle$ and returns the sum of these values. This means that the adversary also learns each of $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ but that is inherent leakage from the ideal functionality. Concretely, an adversary can always pad an encryption of $\mathbf{x}_i$ in the i'th slot with encryptions of $\mathbf{0}$'s in the remaining $n-1$ slots and then decrypt with a key for $\mathbf{y}_1, \ldots, \mathbf{y}_n$ to learn $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. Security is immediate since the underlying FE guarantees that there is no leakage beyond $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. Combined with prior works, we immediately obtain a MIFE for inner product in the public-key setting under the k-LIN assumption in cyclic groups.

The technical bulk of this work lies in constructing a MIFE scheme for inner product in the private-key setting under the standard k-LIN assumption in *bilinear* groups, which is quite different from the public-key scheme which can be instantiated in pairing-free groups. Here, it is easy to see

[3] In this paper, we refer only to unbounded collusions (i.e. the adversary can request for any number of secret keys). See [21, 20, 19, 12] for results on bounded collusions.

	# inputs	setting	security	assumption
ABDP15 [1]	1	public-key	many-SEL-IND	DDH
ALS15 [4], ABDP16 [2]	1	public-key	many-AD-IND	DDH, k-LIN
[25]	1	public-key	one-SEL-SIM	k-LIN
BSW11 [11]	1	any	many-SEL-SIM	impossible
easy	multi	public-key	many-AD-IND	k-LIN
this work	multi	private-key	many-SEL-IND	k-LIN + pairings

Fig. 1: Summary of constructions from cyclic or bilinear groups. We have 8 security notions xx-yy-zzz where xx $\in$ {one, many} refers to the number of challenge ciphertexts; yy $\in$ {SEL, AD} refers to encryption queries are selectively or adaptively chosen; zzz $\in$ {IND, SIM} refers to indistinguishability vs simulation-based security.

that running n independent copies of a single-input scheme as before is insecure, precisely because it leaks $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. To solve this problem, we simply augment the basic n-copy scheme with some randomization terms to mask the intermediate computations $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ and to "glue" the n keys to prevent mix-and-match attacks. Our final scheme as described in Fig. 10 is extremely simple and achieves good concrete efficiency. On the other hand, the analysis is quite involved. At a high level, the challenges are two-fold: achieving security without any leakage beyond the ideal functionality, and without exponential hardness assumptions. We defer a technical overview to Section 1.2 and proceed instead to describe some potential motivating applications for this work and to position the theoretical contributions of this work in the broader context of MIFE.

Motivating applications. Suppose we have a company which keeps profiles of its employees including the grades for their skills that they received in their last evaluation. This information is considered private and the only person who has access to such information about the employees is their direct manager. When a new project is started at the company, a lead manager is assigned to the project and she has to form a new team. In order to evaluate a possible team configuration, she needs to evaluate the skills of the team as a whole. This can be achieved by obtaining a score for the team that is the weighted sum of the skills of people serving in different positions in the team. The specific weights assigned to various skills are determined by the needs of the particular project.

A MIFE scheme can provide a solution for the above scenario as follows. The grades for the skills of each employee can be represented as an integer vector, which will be encrypted under a MIFE scheme that uses its encryption slots to represent different team positions. When a new project is established, the lead manager is granted a decryption key that assigns weights to each of the skills of different team members. She can use this key to evaluate various combinations of people for the team while learning nothing more about everyone's profile than the total team score.

A similar example is the construction of a complex machine that requires parts from different manufacturers. Each part is rated based on different quality characteristics and price, which are all manufacturer's proprietary information until a contract has been signed. The ultimate goal is to assemble a construction of parts that achieve a reasonable trade-off between quality and price. In order to evaluate different construction configurations, the company wants to compute cumulative score for each configuration that is a weighted sum over the quality rates and price of each of the parts.

Theoretical perspective. The focus of this work is on obtaining constructions for a specific class of functions with good concrete efficiency. Nonetheless, we believe that our results do shed some new insights into general feasibility results for MIFE:

- First, our results are indicative of further qualitative differences between MIFE in the public-key and the private-key settings. Indeed, we already know that the security guarantees are quite different due to additional inherent leakages in the public-key setting. In the case of order-revealing encryption [10], the differences are sufficient to enable positive results in the private-key setting, while completely ruling out any construction in the public-key setting. Our results hint at a different distinction, where the private-key setting seems to require qualitative stronger assumptions than in the public-key setting, namely the use of pairings.
- Next, our results provide the first evidence supporting the intuition that MIFE requires qualitatively stronger assumptions than FE, but not too much stronger. Concretely, for the inner product FE, we have existing positive results under the DDH assumption in pairing-free groups. Prior to this work, it was not clear if we could extend the positive results to MIFE for n-ary inner product under the same assumptions, or if n-ary inner product would already require the same complex assumptions as MIFE for circuits. Our results suggest a rather different picture, namely that going from single-input to multi-input should require no more than an extra level of multi-linearity, even for restricted functionalities. The situation is somewhat different for general circuits, where we now know that going from single-input to multi-input incurs no more than a quantitative loss in the underlying assumptions [5, 12].
- Finally, we presented the first MIFE for a non-trivial functionality that polynomial security loss for a super-constant number of slots under falsifiable assumptions. Recall that indistinguishability obfuscation and generic multi-linear maps are not falsifiable, whereas the constructions based on single-input FE in [5, 8, 12] incur a security loss which is exponential in the number of slots. Indeed, there is a reason why prior works relied on non-falsifiable assumptions or super-polynomial security loss. Suppose an adversary makes Q_0 key queries, and $Q_1, \ldots, Q_n$ ciphertext queries for the n slots. By combining the ciphertexts and keys in different ways, the adversary can learn $Q_0 Q_1 \cdots Q_n$ different decryptions. When n is super-constant, the winning condition in the security game may not be efficiently checkable in polynomial-time, hence the need for either a non-falsifiable assumption or a super-polynomial security loss. To overcome this difficulty, we show that for inner product, we can exploit linearity to succinctly characterize the $Q_0 Q_1 \cdots Q_n$ constraints by roughly $Q_0 \cdot (Q_1 + \cdots Q_n)$ constraints.

1.2 Technical Overview

We now present an overview of our private-key MIFE scheme for inner product, which as noted in the previous section, constitutes the technical bulk of this work.

Our private-key MIFE. We show how to construct such a MIFE scheme starting from a single-input scheme that satisfies some additional structural properties.

- The first idea is to use n independent copies of the single-input scheme as in the public-key setting, using the i'th copy to encrypt $\mathbf{x}_i$, and the new secret key is the collection of the n

secret keys corresponding to each of $\mathbf{y}_1, \ldots, \mathbf{y}_n$. Decryption recovers $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle, \ldots \langle \mathbf{x}_n, \mathbf{y}_n \rangle$ and returns the sum of these values. This satisfies correctness, but is clearly insecure as it leaks each individual $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$.

– The next idea is to randomize each $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$, by using the i'th copy of the single-input to encrypt the "expanded vector" $\mathbf{x}_i \| z_i$, where $z_1, \ldots, z_n$ are fixed random values that are part of the private key. The new secret key is the collection of the n secret keys corresponding to each of $\mathbf{y}_1 \| 1, \ldots, \mathbf{y}_n \| 1$ together with $z_1 + \cdots + z_n$. Decryption recovers $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + z_1, \ldots \langle \mathbf{x}_n, \mathbf{y}_n \rangle + z_n$ and then substracts $z_1 + \cdots + z_n$ from the sum. Observe the collection of $n + 1$ values

$$\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + z_1, \ldots, \langle \mathbf{x}_n, \mathbf{y}_n \rangle + z_n, z_1 + \cdots + z_n$$

reveals exactly $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + \cdots + \langle \mathbf{x}_n, \mathbf{y}_n \rangle$ and no more, provided the random offsets $z_1, \cdots, z_n$ are used only once. However, we need to reuse $z_1, \ldots, z_n$ across multiple secret keys in order to achieve correctness, and this lends the scheme to a mix-and-match attack on the secret keys. The attack already works for $n = 2$ and only requires two key queries: the adversary that obtains secret keys for $(\mathbf{y}_1^1, \mathbf{y}_2^1)$ and $(\mathbf{y}_1^2, \mathbf{y}_2^2)$ can produce a secret key for $(\mathbf{y}_1^1, \mathbf{y}_2^2)$, which leaks additional information beyond what is allowed by the ideal functionality. Note that this attack exploits the fact that key generation is deterministic and uses the same $z_1, \ldots, z_n$ across all of the secret keys.

– To defeat the mix-and-match attack, we modify key generation to be randomized as follows: we pick a fresh random r, and the new secret key is the collection of the n secret keys corresponding to each of $\mathbf{y}_1 \| r, \ldots, \mathbf{y}_n \| r$ together with $(z_1 + \cdots + z_n)r$, all of which are encoded in the exponent. Decryption recovers $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + z_1 r, \ldots \langle \mathbf{x}_n, \mathbf{y}_n \rangle + z_n r$ via a pairing and then substracts $(z_1 + \cdots + z_n)r$ from the sum. Intuitively, security follows from the fact that the interaction between the secret key and the ciphertext generates offsets $(z_1 r, \ldots, z_n r)$ that are both pseudorandom and non-malleable via the DDH assumption. In particular, $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + z_1 r$ (encoded in the exponent) computationally hides $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle$. The construction then extends readily to the k-LIN assumption by replacing each of $r, z_1, \ldots, z_n$ with k-dimensional vectors.

This completes the description of our scheme. Note that our scheme is qualitatively different from the single-input schemes in [1, 4, 2] which admit deterministic key generation. Also, we require pairings for decryption since the secret keys now comprise of group elements.

Overview of security proof. As mentioned earlier, we face two challenges in the security proof: (i) avoiding leakage beyond the ideal functionality, and (ii) avoiding super-polynomial hardness assumptions.

The first challenge already arises in the case where there is a single challenge ciphertext per encryption slot. In fact, the issue already arises for $n = 2$ and when the adversary makes a single key query $\mathbf{y}_1 \| \mathbf{y}_2$. Concretely, to prove indistinguishability-based security, we want to switch encryptions $\mathbf{x}_1^0, \mathbf{x}_2^0$ to encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^1$. Here, the leakage from the ideal functionality imposes the restriction that

$$\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^0, \mathbf{y}_2 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^1, \mathbf{y}_2 \rangle$$

and this is the only restriction we can work with. The natural proof strategy is to introduce an intermediate hybrid that generates encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^0$. However, to move from encryptions

$\mathbf{x}_1^0, \mathbf{x}_2^0$ to this hybrid, we would require that $\langle \mathbf{x}_1^0 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle = \langle \mathbf{x}_1^1 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle$, which implies the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle = \langle \mathbf{x}_2^1, \mathbf{y}_1 \rangle$. (Indeed, the single-input inner product scheme in [7] imposes extranous restrictions to overcome similar difficulties in the fuction-hiding setting.)

We avoid the above issue by using simulation-based security so that we can work with a "computational" variant of the extraneous restriction. Roughly speaking, instead of requiring that $\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle = \langle \mathbf{x}_2^1, \mathbf{y}_1 \rangle$, we would only require that

$$\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle + z_1 r \approx_c \langle \mathbf{x}_2^1, \mathbf{y}_1 \rangle + z_1 r$$

where z_1, r are random as described in the construction. In particular, if we encode the quantities in the exponent, then indistinguishability holds under the DDH assumption (or k-Lin if we replace z_1, r by random k-dimensional vectors). See Lemma 1 and Remark 4 for further details.

Next, we address the second challenge of achieving polynomial security loss. Here, we consider an adversary that makes Q_0 key queries, and $Q_1, \ldots, Q_n$ ciphertext queries for the n slots. As described earlier, we want to describe the leakage from the ideal functionality by roughly $Q_0 \cdot (Q_1 + \cdots + Q_n)$ instead of $Q_0 Q_1 \cdots Q_n$ constraints. For simplicity, we again focus on $Q_0 = 1$. We denote the j'th ciphertext query in the i'th slot by $\mathbf{x}_i^{j,b}$, where b is the challenge bit. By decrypting $\mathbf{x}_1^{2,b}, \mathbf{x}_2^{1,b}, \ldots, \mathbf{x}_n^{1,b}$ and $\mathbf{x}_1^{1,b}, \mathbf{x}_2^{1,b}, \ldots, \mathbf{x}_n^{1,b}$ and substracting the two, the adversary learns $\langle \mathbf{x}_1^{2,b} - \mathbf{x}_1^{1,b}, \mathbf{y}_1 \rangle$ and more generally, $\langle \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \mathbf{y}_i \rangle$. Indeed, these are essentially the only constraints we need to work with, namely:

$$\sum_i \langle \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \sum_i \langle \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$$
$$\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle, j = 2, \ldots, Q_i$$

The security proof then proceed as follows:

- We first switch encryptions of $\mathbf{x}_1^{1,0}, \ldots, \mathbf{x}_n^{1,0}$ to those of $\mathbf{x}_1^{1,1}, \ldots, \mathbf{x}_n^{1,1}$ in a "single shot", and for the remaining ciphertexts, we switch from an encryption of $\mathbf{x}_i^{j,0} = (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,0}$ to that of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$. This basically follows from the setting where there is only a single ciphertext in each slot, for which we can establish simulation-based security.
- Then, we apply a hybrid argument across the slots to switch from encryptions of

  $$(\mathbf{x}_i^{2,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_2,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$$

  to those of

  $$(\mathbf{x}_i^{2,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_2,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}.$$

As described earlier, to carry out the latter hybrid argument, the queries must satisfy the constraint

$$\langle (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle = \langle (\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle \iff \langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$$

where the latter is already imposed by the ideal functionality.

To carry out the security reduction, we will require that the underlying single-input FE satisfies a malleability property, namely given Δ, we can maul an encryption of $\mathbf{x}$ into that of $\mathbf{x} + \Delta$. Note that this does not violate security because given $\langle \mathbf{x}, \mathbf{y} \rangle, \mathbf{y}, \Delta$, we can efficiently compute $\langle \mathbf{x} + \Delta, \mathbf{y} \rangle$.

1.3 Discussion

Beyond inner product? Our constructions and techniques may seem a-priori largely tailored to the inner product functionality and properties of bilinear groups. We clarify here that our high-level approach (which builds upon [24, 9, 25]) may be applicable beyond inner product, namely:

i. start with simulation-based security for a single ciphertext per slot and a single secret key (in our case, this corresponds to introducing the additional $z_1, \ldots, z_n$ to hide the intermediate computation $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$);

ii. achieve simulation-based security for a single ciphertext per slot and multiple secret keys, by injecting additional randomness to the secret keys to prevent mix-and-match attacks (for this, we replaced $z_1, \ldots, z_n$ with $z_1 r, \ldots, z_n r, r$ in the exponent);

iii. "bootstrap" to multiple ciphertexts per slot, where we also showed how to avoid incurring an exponential security loss.

In particular, using simulation-based security for i. and ii. helped us avoid additional leakage beyond what is allowed by the ideal functionality.

Additional related work. Goldwasser et al. [18] showed that both two-input public-key MIFE as well as n-input private-key MIFE for circuits already implies indistinguishability obfuscation for circuits.

There have also been several works that proposed constructions for private-key multi-input functional encryption. The work of Boneh et al. [10] constructs a single-key MIFE in the private key setting, which is based on multilinear maps and is proven secure in the idealized generic multilinear map model. Two other papers explore the question how to construct multi-input functional encryption starting from the single input variant. In their work [5] Ananth and Jain demonstrate how to obtain selectively secure MIFE in the private key setting starting from any general-purpose public key functional encryption. In an independent work, Brakerski et al. [12] reduce the construction of private key MIFE to general-purpose private key (single input) functional encryption. The resulting scheme achieves selective security when the starting private key FE is selectively secure. Additionally in the case when the MIFE takes any constant number of inputs, adaptive security for the private key FE suffices to obtain adaptive security for the MIFE construction as well. The constructions in that work provide also function hiding properties for the MIFE encryption scheme.

While this line of work reduces MIFE to single-input FE for general-purpose constructions, the only known instantiations of construction for public and private key functional encryption with unbounded number of keys require either indistinguishability obfuscation [16] or multilinear maps with non-standard assumptions [17]. We stress that the transformations from single-input to MIFE in [5, 12] are not applicable in the case of inner product since these transformations require that the single-input FE for complex functionalities related to computing a PRF, which is not captured by the simple inner functionality.

Open problems. One natural open problem is to eliminate the use of pairings in MIFE for inner product; we think such a result would be quite surprising though. We outline two additional open problems:

– The first is to achieve function privacy, as considered in the setting of single-input inner product functional encryption in [7, 13]. Note that these latter results require pairings. Our first guess is that it would be possible to achieve private-key, function-hiding MIFE for inner product under the k-Linear assumption in bilinear groups.
– The other is to achieve adaptive security. Here, our proof strategy breaks down as the simulator would need to simulate a ciphertext in $\mathbb{G}_1$ given the output of the computation in $\mathbb{G}_2$, which appears to be problematic. This is not an issue in the selective setting, since the simulator could first "obliviously" simulate the ciphertext and then "program" the output of the computation into the secret key. The goal is to achieve adaptive security while still relying only on the k-Linear assumption in bilinear groups and without introducing additional restrictions on the adversary's queries.

2 Preliminaries

Notation. We denote by $s \leftarrow_{\mathrm{R}} S$ the fact that s is picked uniformly at random from a finite set S. By PPT, we denote a probabilistic polynomial-time algorithm. Throughout, we use 1^λ as the security parameter. We use lower case boldface to denote (column) vectors and upper case boldcase to denote matrices.

Cryptographic assumptions We follow the notation and algebraic framework for Diffie-Hellman-like assumptions in [14]. We fix a pairing group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ with $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order q, where q is a prime of $\Theta(\lambda)$ bits. We use the implicit representation notation for group elements: for fixed generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and for a matrix $\mathbf{M}$ over $\mathbb{Z}_q$, we define $[\mathbf{M}]_1 := g_1^{\mathbf{M}}$ and $[\mathbf{M}]_2 := g_2^{\mathbf{M}}$, where exponentiation is carried out component-wise.

The k-Linear Assumption in $\mathbb{G}_1$ (more generally, the Matrix DDH Assumption) specifies an efficiently samplable distribution $\mathcal{D}_k$ over full-rank matrices in $\mathbb{Z}_q^{(k+1)\times k}$, and asserts that

$$([\mathbf{A}]_1, [\mathbf{A}\mathbf{s}]_1) \approx_c ([\mathbf{A}]_1, [\mathbf{u}]_1)$$

where $\mathbf{A} \leftarrow \mathcal{D}_k, \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k, \mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k+1}$. We use $\mathsf{Adv}^{\mathrm{MDDH}}_{\mathbb{G}_1,\mathcal{A}}(\lambda)$ to denote the distinguishing advantage of an adversary $\mathcal{A}$ for the above distributions, and we define $\mathsf{Adv}^{\mathrm{MDDH}}_{\mathbb{G}_2,\mathcal{A}}(\lambda)$ analogously for $\mathbb{G}_2$.

3 Definitions for Multi-Input Functional Encryption

We recall the definitions for multi-input functional encryption from [18]. We focus here on the private-key setting, which allows us to simplify the definitions.

Definition 1 (Multi-input Function Encryption). *Let $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$ 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:*

- $\mathsf{Setup}(1^\lambda, \mathcal{F}_n)$*: on input the security parameter* λ *and a description of* $\mathcal{F}_n \in \mathcal{F}$*, outputs a master public key* mpk[4] *and a master secret key* msk*. All of the remaining algorithms get* mpk *as part of its input.*
- $\mathsf{Encrypt}(\mathsf{msk}, i, x_i)$*: on input the master secret key* msk*,* $i \in [n]$*, and a message* $x_i \in \mathcal{X}_i$*, outputs a ciphertext* ct*. We assume that each ciphertext has an associated index* i*, which denotes what slot this ciphertext can be used for. If* $n = 1$*, we omit the input* i*.*
- $\mathsf{KeyGen}(\mathsf{msk}, f)$*: on input the master secret key* msk *and a function* $f \in \mathcal{F}_n$*, outputs a decryption key* sk_f*.*
- $\mathsf{Decrypt}(\mathsf{sk}_f, f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$*: on input a decryption key* sk_f *for function* f *and* n *ciphertexts, outputs a string* $y \in \mathcal{Y}$*.*

The scheme $\mathcal{MIFE}$ *is correct if for all* $f \in \mathcal{F}$ *and all* $x_i \in \mathcal{X}_i$ *for* $1 \le i \le n$*, we have*

$$\Pr\left[\begin{array}{r} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, n); \\ \mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f); \\ \mathsf{Decrypt}(\mathsf{sk}_f, \mathsf{Encrypt}(\mathsf{msk}, 1, x_1), \ldots, \mathsf{Encrypt}(\mathsf{msk}, n, x_n)) = f(x_1, \ldots, x_n) \end{array}\right] = 1,$$

where the probability is taken over the coins of Setup*,* KeyGen *and* $\mathsf{Encrypt}$*.*

3.1 Security notions

Following [3], we may consider 8 security notions xx-yy-zzz where xx ∈ {one, many} refers to the number of challenge ciphertexts; yy ∈ {SEL, AD} refers to encryption queries are selectively or adaptively chosen; zzz ∈ {IND, SIM} refers to indistinguishability vs simulation-based security. We have the following trivial relations: many ⇒ one, AD ⇒ SEL, and the following standard relations: SIM ⇒ IND, and one-yy-IND ⇒ many-yy-IND, the latter in the public-key setting. Here, we focus on {one,many}-SEL-IND and one-SEL-SIM, which are the notions most relevant to our positive results.

Definition 2 (many-SEL-IND-secure MIFE). *For every stateful adversary* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *defined as*

$$\mathsf{Adv}^{\mathcal{MIFE},IND}_{\mathcal{A},\beta}(\lambda) = |\Pr[\mathbf{IND}^{\mathcal{MIFE}}_{\mathcal{A},1}(1^\lambda) = 1] - \Pr[\mathbf{IND}^{\mathcal{MIFE}}_{\mathcal{A},0}(1^\lambda) = 1]|$$

where the experiment $\mathbf{IND}^{\mathcal{MIFE}}_{\mathcal{A}}(1^\lambda)$ *is defined as follows:*

$\underline{\textit{Experiment}}$ $\mathbf{many\text{-}SEL\text{-}IND}^{\mathcal{MIFE}}_{\mathcal{A},\beta}(1^\lambda)$*:*
$(x_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n)$
$\mathsf{ct}_i^j \leftarrow \mathsf{Encrypt}(\mathsf{msk}, i, x_i^{j,\beta})\ \forall i \in [n], j \in [Q_i]$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}\left((\mathsf{ct}_i^j)_{i\in[n],j\in[Q_i]}\right)$
Output: β'

[4] We note that in the private key setting of MIFE, we can make mpk part of msk, but we allow for a separate master public key for better clarity in our proofs. In constructions where we do not need mpk we omit it.

A private key multi-input functional encryption $\mathcal{MIFE}$ for $\mathcal{F}$ is many-SEL-IND-secure if for every stateful PPT adversary $\mathcal{A}$, which submits admissible challenge ciphertext and decryption key queries that satisfy the following relation:

$$f(x_1^{j_1,0}, \ldots, x_n^{j_n,0}) = f(x_1^{j_1,1}, \ldots, x_n^{j_n,1}) \quad \forall i \in [n], j_i \in [Q_i], \\ \forall f \text{ queries to } \mathsf{KeyGen}(\mathsf{msk}, \cdot),$$

there exists a negligible function $\epsilon(\lambda)$ such that $\mathsf{Adv}_{\mathcal{A}}^{\mathcal{MIFE},IND}(\lambda) < \epsilon(\lambda)$.

Remark 1 (winning condition). Note that the winning condition is in general not efficiently checkable because of the combinatorial explosion in the restriction on the queries.

Next we present the simulation security definition for MIFE in the setting with a single challenge ciphertext per slot.

Definition 3 (one-SEL-SIM-secure MIFE). *A multi-input functional encryption $\mathcal{MIFE}$ for n-ary functions $\mathcal{F}$ is SIM secure if there exists a PPT simulator*[5] $(\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Encrypt}}, \widetilde{\mathsf{KeyGen}})$ *such that for every PPT adversary $\mathcal{A}$, the following two distributions are computationally indistinguishable:*

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

The $\widetilde{\mathsf{KeyGen}}$ *functionality in the above ideal experiment has access to an oracle that provides the value $f(x_1, \ldots, x_n)$ for each query f submitted to* $\widetilde{\mathsf{KeyGen}}$.

Zero vs multiple queries in private-key setting. It is convenient in our proof of security to assume that $Q_1, \ldots, Q_n \geq 1$, that is, there is at least one ciphertext for each encryption slot, which is where the technical bulk of the work lies as we would need to reason about leakage from the ideal functionality. In the setting where some $Q_i = 0$, the ideal functionality leaks nothing, and here, we can easily achieve semantic security for all of the messages being encrypted in the private key MIFE setting, via the following simple generic transformation.

Construction 1 *Let* $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *be a* private key *MIFE construction for n-input functions in the class $\mathcal{F}_n$, which satisfies any* xx-yy-zzz *MIFE security definition when the adversary receives at least one ciphertext for each encryption slot. Let* $(\mathsf{Gen}_{\mathsf{SE}}, \mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$ *be symmetric key encryption. The* private key *MIFE scheme* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *described in Fig. 2 satisfies* xx-yy-zzz *security without any restrictions on the ciphertext challenge sets:*

$\underline{\mathsf{Setup}'(1^\lambda, \mathcal{F}_n)}$:
$\mathsf{msk} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_n)$
$\mathsf{K} \leftarrow \mathsf{Gen}(1^\lambda)$
$k_1, \ldots, k_{n-1} \leftarrow_{\mathrm{R}} \{0,1\}^\lambda$, $k_n = \left(\bigoplus_{i\in[n-1]} k_i\right) \oplus \mathsf{K}$
return $\mathsf{msk}' \leftarrow \left(\mathsf{msk}, \mathsf{K}, \{k_i\}_{i\in[n]}\right)$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:
parse $\mathsf{msk}' = \left(\mathsf{msk}, \mathsf{K}, \{k_i\}_{i\in[n]}\right)$
$\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{msk}, i, \mathbf{x}_i)$
$\mathsf{ct}' \leftarrow \mathsf{Enc}_{\mathsf{SE}}(\mathsf{K}, \mathsf{ct})$
return (k_i, ct')

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, f)}$:
return $\mathsf{KeyGen}(\mathsf{msk}, f)$

$\underline{\mathsf{Decrypt}'(\mathsf{sk}_f, \mathsf{ct}'_1, \ldots, \mathsf{ct}'_n)}$:
parse $\left\{ \mathsf{ct}'_i = (k_i, \mathsf{ct}_i) \right\}_{i\in[n]}$
$\mathsf{K} \leftarrow \bigoplus_{i\in[n]} k_i$
$\left\{ \mathsf{ct}_i \leftarrow \mathsf{Dec}_{\mathsf{SE}}(\mathsf{K}, \mathsf{ct}'_i) \right\}_{i\in[n]}$
return $\mathsf{Decrypt}(\mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$.

Fig. 2: Compiler from private-key MIFE with xx-yy-zzz security when $|Q_i| > 0$ for all i to private-key MIFE with xx-yy-zzz security

Proof (sketch). We consider two cases:

- Case 1: there exists some $i \in [n]$ for which $Q_i = 0$. Here, k_i and thus K is perfectly hidden from the adversary. Then, security follows readily from semantic security of $(\mathsf{Gen}_{\mathsf{SE}}, \mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$.
- Case 2: for all i, $Q_i \geq 1$. Here, security follows immediately from that of $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$.

□

3.2 Inner product functionality

Multi-input Inner product. We construct a multi-input functional encryption that supports the class of multi-input bounded-norm inner product functions, which is defined as $\mathcal{F}_n^{m,B} = \{f_{\mathbf{y}_1,\ldots,\mathbf{y}_n} : (\mathbb{Z}^m)^n \to \mathbb{Z}\}$ where

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

We require that the norm of the inner product of any two vector components from function and input $\langle \mathbf{x}, \mathbf{y} \rangle$ is bounded by B. This bound will determine the parameters of the bilinear map groups

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

that we will be using in our constructions; in particular, we will choose a target group that has order $q \gg n \cdot B$. To simplify naming conventions, we will omit "bounded-norm" for the rest of the paper, but we will always refer to a multi-input inner-product functionality with this property.

Remark on leakage. For all i, j, k, the adversary can learn

$$\langle \mathbf{x}_k^i - \mathbf{x}_k^1, \mathbf{y}_k^j \rangle$$

via the ideal functionality. In the IND security game, this means the adversary is restricted to queries satisfying

$$\langle \mathbf{x}_k^{i,0} - \mathbf{x}_k^{1,0}, \mathbf{y}_k^j \rangle = \langle \mathbf{x}_k^{i,1} - \mathbf{x}_k^{1,1}, \mathbf{y}_k^j \rangle \iff \langle \mathbf{x}_k^{i,0} - \mathbf{x}_k^{1,1}, \mathbf{y}_k^j \rangle = \langle \mathbf{x}_k^{i,0} - \mathbf{x}_k^{1,1}, \mathbf{y}_k^j \rangle$$

In the hybrid, we want to avoid additional constraints such as

$$\langle \mathbf{x}_k^{i,0} - \mathbf{x}_k^{1,0}, \mathbf{y}_k^j \rangle = \langle \mathbf{x}_k^{i,0} - \mathbf{x}_k^{1,1}, \mathbf{y}_k^j \rangle = \langle \mathbf{x}_k^{i,1} - \mathbf{x}_k^{1,0}, \mathbf{y}_k^j \rangle = \langle \mathbf{x}_k^{i,1} - \mathbf{x}_k^{1,1}, \mathbf{y}_k^j \rangle$$

4 Private-Key MIFE for Inner Product

In this section, we present our main result, namely a private-key MIFE for inner product that achieves many-SEL-IND security. We use a pairing group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ with $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order q, where q is a prime of $\Theta(\lambda)$ bits. Our construction relies on the k-Linear Assumption in $\mathbb{G}_1$ and in $\mathbb{G}_2$ and is shown in Fig. 10.

We present our construction in two steps:

- First, we show how to construct such a MIFE scheme starting from a single-input one-SEL-SIM scheme that satisfies some additional structural properties.
- Next, we show how to instantiate the underlying single-input scheme (cf. Fig. 11) and we present a self-contained description of the scheme in Fig. 10.

We refer the reader to Section 1.1 for an overview of the construction.

4.1 Multi-Input scheme from single-input scheme

Main construction. We build a private key multi-input FE $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the class $\mathcal{F}_n^{m,B}$, starting from a private key single-input FE $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the class $\mathcal{F}_1^{m+k,B}$. We present our construction in Fig. 3

Correctness. Correctness follows readily from the correctness of the underlying scheme and the equation:

$$\langle \mathbf{x}_1 \| \cdots \| \mathbf{x}_n, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n \rangle = (\sum_{i=1}^{n} \langle \mathbf{x}_i \| \mathbf{z}_i, \mathbf{y}_i \| \mathbf{r} \rangle) - \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$$

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

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{F}_1^{m+k,B}), i = 1, \ldots, n$

$\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k, i = 1, \ldots, n$

$(\mathsf{mpk}, \mathsf{msk}) := \left(\{ \mathsf{mpk}_i \}_{i\in[n]}, \{ \mathsf{msk}_i, \mathbf{z}_i \}_{i\in[n]} \right)$

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:

return $\mathsf{Enc}(\mathsf{msk}_i, \mathbf{x}_i \| \mathbf{z}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$

$\mathbf{d}_i \leftarrow \mathsf{KeyGen}(\mathsf{msk}_i, \mathbf{y}_i \| \mathbf{r}), i = 1, \ldots, n$

$z := \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$

$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \left(\{ [\mathbf{d}_i]_2 \}_{i\in[n]}, [\mathbf{r}]_2, [z]_T \right)$

return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

$\underline{\mathsf{Dec}'((\{ [\mathbf{d}_i]_2 \}_{i\in[n]}, [\mathbf{r}]_2, [z]_T), \mathbf{y}_1 \| \cdots \| \mathbf{y}_n, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)}$:

$[a_i]_T \leftarrow \mathsf{Dec}([\mathbf{d}_i]_2, [\mathbf{y}_i \| \mathbf{r}_i]_2, \mathsf{ct}_i), i = 1, \ldots, n$

return the discrete log of $\left(\prod_{i=1}^n [a_i]_T\right) / [z]_T$

Fig. 3: Multi-input functional encryption scheme $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the class $\mathcal{F}_n^{m,B}$. $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ refers to the single-input functional encryption scheme for the class $\mathcal{F}_1^{m+k,B}$.

Additional requirements. The construction and the analysis requires that $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfy the following structural properties:

- The scheme can be instantiated over $\mathbb{G}_1$, where the ciphertext is a vector $[\mathbf{c}]_1$ over $\mathbb{G}_1$ and the secret key is a vector $\mathbf{d}_i$ over $\mathbb{Z}_q$.
- Enc is linearly homomorphic and public-key; More specifically, we only require that, given $\mathsf{mpk}, \mathsf{Enc}(\mathsf{msk}, \mathbf{x}), \mathbf{x}'$, we can generate a fresh random encryption of $\mathbf{x} + \mathbf{x}'$, i.e. $\mathsf{Enc}(\mathsf{msk}, \mathbf{x} + \mathbf{x}')$. This property is used in the proof of Lemma 2.
- For correctness, Dec should be linear in both of its inputs, so that $\mathsf{Dec}([\mathbf{d}]_2, [\mathbf{y}]_2, [\mathbf{c}]_1) = [\mathsf{Dec}(\mathbf{d}, \mathbf{y}, [\mathbf{c}]_1)]_T \in \mathbb{G}_T$ can be computed using a pairing;
- For an efficient MIFE decryption, Dec must work without any restriction on the norm of the output as long as the output is in the exponent;
- Let $(\widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ be the stateful simulator for the one-SEL-SIM single-input inner-product FE scheme. We require that $\widetilde{\mathsf{KeyGen}}$ takes input $(\mathbf{y}, a)$ and is linear in both of its inputs, so that we can compute $\widetilde{\mathsf{KeyGen}}([\mathbf{y}]_2, [a]_2) = [\widetilde{\mathsf{KeyGen}}(\mathbf{y}, a)]_2$. This property is used in the proof of Lemma 1.

In addition, we require that $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfies both one-SEL-SIM and many-SEL-IND security. In the instantiation in Section 4.2, we rely on a public-key scheme that satisfies one-SEL-SIM security, which implies one-SEL-IND and thus many-SEL-IND security.

Remark 2 (notation). We use subscripts and superscripts for indexing over multiple copies, and never for indexing over positions or exponentiation. Concretely, the j'th ciphertext query in slot i is $\mathbf{x}_i^j$ and the j'th key query is $\mathbf{y}_1^j \| \cdots \| \mathbf{y}_n^j$.

Lemma 1. *Suppose* $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *is one-SEL-SIM-secure and that k-LIN holds in* $\mathbb{G}_2$*. Then,* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *is one-SEL-SIM-secure.*

$\underline{\widetilde{\mathsf{Setup}}'(1^\lambda, \mathcal{F}_n^{m,B})}$:
$\left(\widetilde{\mathsf{mpk}}_i, \widetilde{\mathsf{msk}}_i\right) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \mathcal{F}_1^{m+k,B}), i = 1, \ldots, n$
$(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{msk}}) := \left(\left\{ \widetilde{\mathsf{mpk}}_i \right\}_{i\in[n]}, \left\{ \widetilde{\mathsf{msk}}_i \right\}_{i\in[n]}\right)$
return $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{msk}})$

$\underline{\widetilde{\mathsf{Enc}}'(\widetilde{\mathsf{msk}}, i)}$:
return $\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}_i)$

$\underline{\widetilde{\mathsf{KeyGen}}'(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n, a)}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$
$\tilde{z}_1, \ldots, \tilde{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q$
$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}\left(\widetilde{\mathsf{msk}}_i, [\mathbf{y}_i \| \mathbf{r}]_2, [\tilde{z}_i]_2\right), i = 1, \ldots, n$
$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \left(\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [\tilde{z}_1 + \cdots + \tilde{z}_n - a]_T\right)$
return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

Fig. 4: Simulator for one-SEL-SIM scheme for the multi-input inner product $\mathcal{F}_n^{m,B}$.

Proof. We consider the following sequence of games, starting with the real experiment, and ending with the simulator, which is described in Fig. 4.

Game 0. Real.

Game 1. For each slot $i \in [n]$, compute the master and public keys $(\mathsf{mpk}_i, \mathsf{msk}_i)$ and the challenge ciphertext ct_i using $\widetilde{\mathsf{Setup}}$ and $\widetilde{\mathsf{Enc}}$, respectively. Moreover, compute the key for $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ by running $[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}([\mathbf{y}_i \| \mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \langle \mathbf{z}_i, \mathbf{r} \rangle]_2)$ for a fresh $\mathbf{r}$ and outputting

$$\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [\langle \mathbf{r}, \mathbf{z}_1 + \cdots + \mathbf{z}_n \rangle]_T$$

We have Game 0 $\approx_c$ Game 1 due to the one-SEL-SIM security of $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$. This follows from the description of $\mathbf{Game}_{0,\ell}$ in Fig. 5, by noticing that Game 0 $\equiv \mathbf{Game}_{0,0}$,

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

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \widetilde{\mathsf{Setup}}\left(1^\lambda, \mathcal{F}_1^{m+k,B}\right), i = 1, \ldots, \ell$

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}\left(1^\lambda, \mathcal{F}_1^{m+k,B}\right), i = \ell+1, \ldots, n$

$\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k, i = 1, \ldots, n$

$(\mathsf{mpk}, \mathsf{msk}) := \left(\left\{\, \mathsf{mpk}_i \,\right\}_{i\in[n]}, \left\{\, \mathsf{msk}_i, \mathbf{z}_i \,\right\}_{i\in[n]}\right)$

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:

if $i \le \ell$, then return $\widetilde{\mathsf{Enc}}(\mathsf{msk}_i)$

if $i > \ell$ then return $\mathsf{Enc}\left(\mathsf{msk}_i, \mathbf{x}_i \| \mathbf{z}_i\right)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$

$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}\left(\mathsf{msk}_i, [\mathbf{y}_i \| \mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \langle \mathbf{z}_i, \mathbf{r} \rangle]_2\right), i = 1, \ldots, \ell$

$\mathbf{d}_i \leftarrow \mathsf{KeyGen}\left(\mathsf{msk}_i, \mathbf{y}_i \| \mathbf{r}\right), i = \ell+1, \ldots, n$

$z := \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$

$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \left(\left\{\, [\mathbf{d}_i]_2 \,\right\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\right)$

return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

Fig. 5: Description of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}')$ defining $\mathbf{Game}_{0,\ell}$ in Lemma 1 proof.

Game 1 $\equiv$ $\mathbf{Game}_{0,n}$, and $\mathbf{Game}_{0,\ell-1} \approx_c \mathbf{Game}_{0,\ell}$ due to the one-SEL-SIM security of $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$.

For the latter part to work, we require $\widetilde{\mathsf{KeyGen}}$ to be linear in both of its inputs $(\mathbf{y}, a)$, so that we can compute $[\widetilde{\mathsf{KeyGen}}(\mathbf{y}, a)]_2$ via $\widetilde{\mathsf{KeyGen}}([\mathbf{y}]_2, [a]_2)$.

Game 2. For each slot $i \in [n]$, continue to compute the master and public keys $(\mathsf{mpk}_i, \mathsf{msk}_i)$ and the challenge ciphertext ct_i using $\widetilde{\mathsf{Setup}}$ and $\widetilde{\mathsf{Enc}}$, respectively. However, compute the key for $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ by running $[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}([\mathbf{y}_i \| \mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \tilde{z}_i]_2)$ for fresh values of $\mathbf{r}$ and $\tilde{z}_1, \ldots, \tilde{z}_n$ and outputting

$$\left\{\, [\mathbf{d}_i]_2 \,\right\}_{i\in[n]}, [\mathbf{r}]_2, [\tilde{z}_1 + \cdots + \tilde{z}_n]_T$$

We have Game 1 $\approx_c$ Game 2 by assuming that k-LIN holds in $\mathbb{G}_2$. This follows from the description of $\mathbf{Game}_{1,\ell}$ in Fig. 6, by first noticing that Game 1 $\equiv$ $\mathbf{Game}_{1,0}$ and Game 2 $\equiv$ $\mathbf{Game}_{1,n}$, and that $\mathbf{Game}_{1,\ell-1} \approx_c \mathbf{Game}_{1,\ell}$ via k-LIN in $\mathbb{G}_2$, which tells us that

$$\left\{\, ([\mathbf{r}^j]_2, [\langle \mathbf{z}_\ell, \mathbf{r}^j \rangle]_2) \,\right\}_{j\in[Q_0]} \approx_c \left\{\, ([[\mathbf{r}^j]_2, [\tilde{z}_\ell^j]_2) \,\right\}_{j\in[Q_0]},$$

where Q_0 is the total number of key queries and $\mathbf{r}^j$ and $\tilde{z}_i^j$ denote the random values used to answer the j-th key query.

In a bit more detail, k-LIN in $\mathbb{G}_2$ implies that (e.g. [14, Lemma 1])

$$\left([\mathbf{B}]_2, [\mathbf{B}\mathbf{s}]_2\right) \approx_c \left([\mathbf{B}]_2, [\mathbf{u}]_2\right)$$

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

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \widetilde{\mathsf{Setup}}\left(1^\lambda, \mathcal{F}_1^{m+k,B}\right), i = 1, \ldots, n$

$\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k, i = \ell+1, \ldots, n$

$\mathsf{msk} := \left\{ \mathsf{msk}_i, \mathbf{z}_i \right\}_{i\in\{1,\ldots,\ell\}}, \left\{ \mathsf{msk}_i \right\}_{i\in\{\ell+1,\ldots,n\}}$

$\mathsf{mpk} := \left\{ \mathsf{mpk}_i \right\}_{i\in[n]}$

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:

return $\widetilde{\mathsf{Enc}}(\mathsf{msk}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$

$\tilde{z}_1, \ldots, \tilde{z}_\ell \leftarrow_{\mathrm{R}} \mathbb{Z}_q$

$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}([\mathbf{y}_i \| \mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \tilde{z}_i]_2), i = 1, \ldots, \ell$

$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}(\mathsf{msk}_i, [\mathbf{y}_i \| \mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \langle \mathbf{z}_i, \mathbf{r} \rangle]_2), i = \ell+1, \ldots, n$

$z := \tilde{z}_1 + \ldots + \tilde{z}_\ell + \langle \mathbf{z}_{\ell+1} + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$

$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \left(\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T \right)$

return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

Fig. 6: Description of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}')$ defining $\mathbf{Game}_{1,\ell}$ in Lemma 1 proof.

where $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_k, \mathbf{B} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{Q_0 \times k}, \mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{Q_0}$. The statement above corresponds to taking $\mathbf{s} = \mathbf{z}_\ell$, $\mathbf{r}^j$ to be the transponse of the j'th row of $\mathbf{B}$, and $\tilde{z}_\ell^j$ to be the j'th entry in $\mathbf{u}$.

Game 3. For each slot $i \in [n]$, continue to compute the master and public keys $(\mathsf{mpk}_i, \mathsf{msk}_i)$ and the challenge ciphertext ct_i using $\widetilde{\mathsf{Setup}}$ and $\widetilde{\mathsf{Enc}}$, respectively. However, compute the key for $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ by running $[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}_i, [\mathbf{y}_i \| \mathbf{r}]_2, [\tilde{z}_i]_2)$ for fresh values of $\mathbf{r}$ and $\tilde{z}_1, \ldots, \tilde{z}_n$ and outputting

$$\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [\tilde{z}_1 + \cdots + \tilde{z}_n - \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle]_2$$

We have Game 2 $\equiv$ Game 3 via the change of variables $\langle \mathbf{x}_i, \mathbf{y}_i^j \rangle + \tilde{z}_i^j \mapsto \tilde{z}_i^j$. Moreover, it is straightforward to notice that Game 3 $\equiv$ Ideal using the simulator in Fig. 4 since $a = \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$.

This completes the proof. □

Remark 3 (decryption capabilities). As a sanity check, we note that the simulated secret key will correctly decrypt a simulated ciphertext. However, unlike schemes proven secure via the standard dual system encryption methodology [23], a simulated secret key will incorrectly decrypt a normal ciphertext. This is not a problem because we are in the private-key setting, so a distinguisher will not be able to generate normal ciphertexts by itself.

Remark 4 (why a naive argument is inadequate). We cannot afford to do a naive hybrid argument across the n slots for the challenge ciphertext as it would introduce extraneous restrictions on

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

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \widetilde{\mathsf{Setup}}\left(1^\lambda, \mathcal{F}_1^{m+k,B}\right), i = 1, \ldots, n$

$(\mathsf{mpk}, \mathsf{msk}) := \left(\left\{\, \mathsf{mpk}_i \,\right\}_{i\in[n]}, \left\{\, \mathsf{msk}_i, \mathbf{z}_i \,\right\}_{i\in[n]}\right)$

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:

return $\widetilde{\mathsf{Enc}}(\mathsf{msk}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \mathbf{y}_1\| \cdots \|\mathbf{y}_n)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$

$\tilde{z}_1, \ldots, \tilde{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q$

$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}([\mathbf{y}_i\|\mathbf{r}]_2, [\langle \mathbf{x}_i, \mathbf{y}_i\rangle + \tilde{z}_i]_2), i = 1, \ldots, n$

$z := \tilde{z}_1 + \ldots + \tilde{z}_n$

$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \left(\left\{\, [\mathbf{d}_i]_2 \,\right\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\right)$

return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

Fig. 7: Description of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}')$ defining $\mathbf{Game}_2$ in Lemma 1 proof.

the adversary's queries. Concretely, suppose we want to use a hybrid argument to switch from encryptions of $\mathbf{x}_1^0, \mathbf{x}_2^0$ in Game 0 to those of $\mathbf{x}_1^1, \mathbf{x}_2^1$ in Game 2 with an intermediate hybrid that using encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^0$ in Game 1. To move from Game 0 to Game 1, the adversary's query $\mathbf{y}_1\|\mathbf{y}_2$ must satisfy $\langle \mathbf{x}_1^0\|\mathbf{x}_2^0, \mathbf{y}_1\|\mathbf{y}_2\rangle = \langle \mathbf{x}_1^1\|\mathbf{x}_2^0, \mathbf{y}_1\|\mathbf{y}_2\rangle$, which implies the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1\rangle = \langle \mathbf{x}_2^1, \mathbf{y}_1\rangle$.

As described in the proof above, we overcome the limitation by using simulation-based security. Note that what essentially happens in the first slot in our proof is as follows (for $k = 1$): we switch from $\mathsf{Enc}(\mathbf{x}_1^0\|z_1)$ to $\mathsf{Enc}(\mathbf{x}_1^1\|z_1)$ while giving out a secret key which contains $[\mathsf{sk}_{\mathbf{y}_1\|r^1}]_2, [r^1]_2$. Observe that

$$\langle \mathbf{x}_1^0\|z_1, \mathbf{y}_1\|r^1\rangle = \langle \mathbf{x}_1^0, \mathbf{y}_1\rangle + z_1 r^1, \quad \langle \mathbf{x}_1^1\|z_1, \mathbf{y}_1\|r^1\rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1\rangle + z_1 r^1$$

may not be equal, since we want to avoid the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1\rangle = \langle \mathbf{x}_2^1, \mathbf{y}_1\rangle$. This means that one-SEL-IND security does not provide any guarantee that the ciphertexts are indistinguishable. However, one-SEL-SIM security does provide such a guarantee, because

$$([\langle \mathbf{x}_1^0, \mathbf{y}_1\rangle + z_1 r^1]_2, [r^1]_2) \approx_c ([\langle \mathbf{x}_1^1, \mathbf{y}_1\rangle + z_1 r^1]_2, [r^1]_2)$$

via the DDH assumption in $\mathbb{G}_2$. Since the outcomes of the decryption are computationally indistinguishable, the output of the simulated ciphertext would also be computationally indistinguishable.

Lemma 2. *Suppose* $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ *is many-SEL-IND-secure and* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *is one-SEL-IND-secure. Then,* $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ *is many-SEL-IND-secure.*

Recall that one-SEL-SIM security implies one-SEL-IND security. Note that we do not change the distribution of the secret keys throughout this proof.

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

$(\mathsf{mpk}_i, \mathsf{msk}_i) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \mathcal{F}_1^{m+k,B}), i = 1, \ldots, n$

$(\mathsf{mpk}, \mathsf{msk}) := \left(\{ \mathsf{mpk}_i \}_{i\in[n]}, \{ \mathsf{msk}_i, \mathbf{z}_i \}_{i\in[n]} \right)$

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i)}$:

return $\widetilde{\mathsf{Enc}}(\mathsf{msk}_i)$

$\underline{\mathsf{KeyGen}'(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$

$\tilde{z}_1, \ldots, \tilde{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q$

$[\mathbf{d}_i]_2 \leftarrow \widetilde{\mathsf{KeyGen}}([\mathbf{y}_i \| \mathbf{r}]_2, [\tilde{z}_i]_2), i = 1, \ldots, n$

$z := \tilde{z}_1 + \ldots + \tilde{z}_n - \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$

$\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{ [\mathbf{d}_i]_2 \}_{i\in[n]}, [\mathbf{r}]_2, [z]_T)$

return $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

Fig. 8: Description of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}')$ used to define $\mathbf{Game}_3$ in the proof of Lemma 1.

Game 0. This game is described in Fig. 9 and corresponds to the Experiment $\mathbf{many\text{-}SEL\text{-}IND}_{\mathcal{A},\beta}^{\mathcal{MIFE}}(1^\lambda)$ with $\beta = 0$.

Game 1. This game, which is described in Fig. 9, replaces $\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{j,0}) = \mathsf{Enc}'(\mathsf{msk}, i, \boxed{\mathbf{x}_i^{1,0}} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ with $\mathsf{Enc}(\mathsf{msk}, i, \boxed{\mathbf{x}_i^{1,1}} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ for all $i \in [n], j \in [Q_i]$.

We have Game 0 $\approx_c$ Game 1 via the following properties:

- one-SEL-IND security of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$;
- given $\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}$, we can maul an encryption of $\mathbf{x}_i^{1,\beta}$ under Enc' (corresponding to challenge ciphertext in slot i in the one-SEL-IND security game) into that of $\mathbf{x}_i^{1,\beta} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$ (corresponding to challenge ciphertexts in slots i in Game β).

More precisely, we can build an adversary $\mathcal{D}$ against the one-SEL-IND security of $(\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ such that $|\Pr[\mathbf{Game}_0(\mathcal{A}) = 1] - \Pr[\mathbf{Game}_1(\mathcal{A}) = 1]| \leq \mathsf{Adv}_{\mathcal{A}}^{\mathcal{MIFE}, one-SEL-IND}(\lambda)$.
Adversary $\mathcal{D}$ works as follows. Let $(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}}$ be the challenge vector that $\mathcal{A}$ outputs. $\mathcal{D}$ first outputs the pair of vectors $\mathbf{x}_1^{1,0}\| \ldots \|\mathbf{x}_n^{1,0}$ and $\mathbf{x}_1^{1,1}\| \ldots \|\mathbf{x}_n^{1,1}$ as its selective challenge to get back mpk and the challenge ciphertexts $\mathsf{ct}_1^0\| \ldots \|ct_n^0$ corresponding to $\mathsf{Enc}'(\mathsf{msk}, 1, \mathbf{x}_1^{1,\beta}), \ldots, \mathsf{Enc}'(\mathsf{msk}, n, \mathbf{x}_n^{1,\beta})$.
Next, $\mathcal{D}$ uses the fact that the single-input inner-product scheme is public-key and linearly homomorphic to generate all the remaining ciphertexts ct_i^j for $i \in [n], j \in \{2, \ldots, Q_i\}$ by combining $\mathsf{ct}_i^0 = \mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{1,\beta}) = \mathsf{Enc}(\mathsf{msk}_i, \mathbf{x}_i^{1,\beta} \| \mathbf{z}_i)$ with $\mathsf{Enc}(\mathsf{msk}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \| 0)$ (which can be done using mpk_i). In particular, each ct_i^j will correspond to the $\mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$, which matches the challenge ciphertexts in Game β. $\mathcal{D}$ then provides these values to $\mathcal{A}$ and uses

Experiment **Game** 0:
$(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'\left(1^\lambda, \mathcal{F}_n^{m,B}\right)$
$\forall i \in [n], j \in [Q_i]$
$\quad \mathsf{ct}_i^j \leftarrow \mathsf{Enc}'\left(\mathsf{msk}, i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0}\right)$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}'(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, (\mathsf{ct}_i^j)_{i\in[n],j\in[Q_i]}\right)$
return β'

Experiment **Game** 1:
$(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'\left(1^\lambda, \mathcal{F}_n^{m,B}\right)$
$\forall i \in [n], j \in [Q_i]$
$\quad \mathsf{ct}_i^j \leftarrow \mathsf{Enc}'\left(\mathsf{msk}, i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \boxed{\mathbf{x}_i^{1,1}}\right)$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}'(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, (\mathsf{ct}_i^j)_{i\in[n],j\in[Q_i]}\right)$
return β'

Experiment **Game** 2:
$(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}} \leftarrow \mathcal{A}(1^\lambda, \mathcal{F}_n)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'\left(1^\lambda, \mathcal{F}_n^{m,B}\right)$
$\forall i \in [n], j \in [Q_i]$
$\quad \mathsf{ct}_i^j \leftarrow \mathsf{Enc}'\left(\mathsf{msk}, i, \boxed{\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}} + \mathbf{x}_i^{1,1}\right)$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}'(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, (\mathsf{ct}_i^j)_{i\in[n],j\in[Q_i]}\right)$
return β'

Experiment **Game**$_{1,\ell}$:
$(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}} \leftarrow \mathcal{A}\left(1^\lambda, \mathcal{F}_n\right)$
$(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'\left(1^\lambda, \mathcal{F}_n^{m,B}\right)$
$\forall i \in \{1, \ldots, \ell\}, j \in [Q_i]$
$\quad \mathsf{ct}_i^j \leftarrow \mathsf{Enc}'\left(\mathsf{msk}, i, \boxed{\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}} + \mathbf{x}_i^{1,1}\right)$
$\forall i \in \{\ell+1, \ldots, n\}, j \in [Q_i]$
$\quad \mathsf{ct}_i^j \leftarrow \mathsf{Enc}'\left(\mathsf{msk}, i, \boxed{\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}} + \mathbf{x}_i^{1,1}\right)$
$\beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}'(\mathsf{msk},\cdot)}\left(\mathsf{mpk}, (\mathsf{ct}_i^j)_{i\in[n],j\in[Q_i]}\right)$
return β'

Fig. 9: **Game** 0, **Game** 1, **Game** 2, and **Game**$_{1,\ell}$ for the proof of Lemma 2. The description of Setup', Enc', and KeyGen' are omitted since their simulations do not change throughout the proof.

its KeyGen oracle to simulate the corresponding queries by $\mathcal{A}$. Finally, when $\mathcal{A}$ outputs β', $\mathcal{D}$ returns the same bit as its guess.

From the description above, it follows that

$$|\Pr[\mathbf{Game}_0(\mathcal{A}) = 1] - \Pr[\mathbf{Game}_1(\mathcal{A}) = 1]| \leq \mathsf{Adv}_{\mathcal{A}}^{\mathcal{MIFE},one-SEL-IND}(\lambda)$$

and that the restrictions on the key queries are similar to those imposed by the one-SEL-IND security game, namely that $\sum_{i=1}^n \langle \mathbf{x}_i^{0,0}, \mathbf{y}_i^j \rangle) = \sum_{i=1}^n \langle \mathbf{x}_i^{0,1}, \mathbf{y}_i^j \rangle)$ for every key query $\mathbf{y}_1^j \| \ldots \| \mathbf{y}_n^j$, which follows from the restriction in Section 3.2.

Game 2. This game, which is described in Fig. 9, replaces $\mathsf{Enc}(\mathbf{x}_i^{1,1} + \boxed{(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})} \| \mathbf{z}_i)$ with $\mathsf{Enc}(\mathbf{x}_i^{1,1} + \boxed{(\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1})} \| \mathbf{z}_i) = \mathsf{Enc}(\mathbf{x}_i^{j,1} \| \mathbf{z}_i)$, for all $i \in [n], j \in [Q_i]$. We have Game 1 $\approx_c$ Game 2 via the following properties:

- many-SEL-IND security of $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$;
- the fact that for each key query $\mathbf{y}_1 \| \ldots \| \mathbf{y}_n$ and all $\mathbf{r}$, $\mathbf{z}$, we have

$$\langle \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \| \mathbf{z}, \mathbf{y}_i \| \mathbf{r} \rangle = \langle \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} \| \mathbf{z}, \mathbf{y}_i \| \mathbf{r} \rangle.$$

The latter is equivalent to $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$, which follows from the restriction in Section 3.2.

To be more precise, consider the hybrid game $\mathbf{Game}_{1,\ell}$ in Fig. 9. From its description, it is clear that Game 1 $\equiv \mathbf{Game}_{1,0}$ and Game 2 $\equiv \mathbf{Game}_{1,n}$. Moreover, we can build an adversary $\mathcal{D}$ against the many-SEL-IND security of the single-input inner-product FE scheme $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ such that $|\Pr[\mathbf{Game}_{1,\ell-1}(\mathcal{A}) = 1] - \Pr[\mathbf{Game}_{1,\ell}(\mathcal{A}) = 1]| \leq \mathsf{Adv}_{\mathcal{A}}^{\mathcal{FE},many-SEL-IND}(\lambda)$.

Adversary $\mathcal{D}$ works as follows. Let $(\mathbf{x}_i^{j,b})_{i\in[n],j\in[Q_i],b\in\{0,1\}}$ be the challenge vector that $\mathcal{A}$ outputs. $\mathcal{D}$ first generates the public and secret keys $(\mathsf{msk}_i, \mathsf{mpk}_i)$ for $i \in [n], i \neq \ell$, as well as the values $\mathbf{z}_i$ for $i \in [n]$. Then, it computes the values $\bar{\mathbf{x}}^{j,b} = \mathbf{x}_\ell^{1,1} + \mathbf{x}_\ell^{j,b} - \mathbf{x}_\ell^{1,b} \| \mathbf{z}_\ell$ for $j \in [Q_\ell], b \in \{0,1\}$ and outputs them as its selective challenge to get back mpk_i and $\mathsf{ct}_\ell^1 \| \ldots \| ct_\ell^{Q_\ell}$, where ct_ℓ^j corresponds to $\mathsf{Enc}(\mathsf{msk}_\ell, \mathbf{x}_\ell^{1,1} + \mathbf{x}_\ell^{j,\beta} - \mathbf{x}_\ell^{1,\beta} \| \mathbf{z}_\ell)$. The remaining ciphertexts ct_i^j are generated honestly. For $i < \ell, j \in \{1, \ldots, Q_i\}$, $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{msk}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1} \| \mathbf{z}_i) = \mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$. For $i > \ell, j \in \{1, \ldots, Q_i\}$, $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{msk}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1} \| \mathbf{z}_i) = \mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1}) = \mathsf{Enc}'(\mathsf{msk}, i, \mathbf{x}_i^{j,1})$.

Next, $\mathcal{D}$ provides these values to $\mathcal{A}$ and answers key queries by $\mathcal{A}$ using its KeyGen oracle to compute the terms associated with slot ℓ. Finally, when $\mathcal{A}$ outputs β', $\mathcal{D}$ returns the same bit as its guess.

From the description above, it follows that $|\Pr[\mathbf{Game}_{1,\ell-1}(\mathcal{A}) = 1] - \Pr[\mathbf{Game}_{1,\ell}(\mathcal{A}) = 1]| \leq \mathsf{Adv}_{\mathcal{A}}^{\mathcal{FE},many-SEL-IND}(\lambda)$ since $\mathcal{D}$ simulates $\mathbf{Game}_{1,\ell-1}(\mathcal{A})$ when $\beta = 0$ and $\mathbf{Game}_{1,\ell}(\mathcal{A})$ when $\beta = 1$. Moreover, the restrictions on the key queries imposed by the many-SEL-IND security of the underlying single-input scheme implies that, for every key query $\mathbf{y}_1 \| \ldots \| \mathbf{y}_n$,

$$\langle \mathbf{x}_\ell^{1,1} + \mathbf{x}_\ell^{j,0} - \mathbf{x}_\ell^{1,0} \| \mathbf{z}_\ell, \mathbf{y}_\ell \| \mathbf{r} \rangle = \langle \mathbf{x}_\ell^{1,1} + \mathbf{x}_\ell^{j,1} - \mathbf{x}_\ell^{1,1} \| \mathbf{z}_\ell, \mathbf{y}_\ell \| \mathbf{r} \rangle.$$

This is in turn equivalent to $\langle \mathbf{x}_\ell^{j,0} - \mathbf{x}_\ell^{1,0}, \mathbf{y}_\ell \rangle = \langle \mathbf{x}_\ell^{j,1} - \mathbf{x}_\ell^{1,1}, \mathbf{y}_\ell \rangle$, which follows from the restriction in Section 3.2.

4.2 Putting everything together

In Figure 10 we spell out the details of the scheme in the previous section with a concrete instantiation of the underlying single-input inner-product scheme, which is provided for completeness in Figure 11.

References

[1] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Simple functional encryption schemes for inner products. In J. Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Heidelberg, Mar. / Apr. 2015.

[2] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011, 2016. http://eprint.iacr.org/2016/011.

[3] S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 500–518. Springer, Heidelberg, Aug. 2013.

[4] S. Agrawal, B. Libert, and D. Stehle. Fully secure functional encryption for inner products, from standard assumptions. Cryptology ePrint Archive, Report 2015/608, 2015. http://eprint.iacr.org/2015/608.

[5] P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 308–326. Springer, Heidelberg, Aug. 2015.

$\underline{\mathsf{Setup}(\mathbb{G}, \mathcal{F}_n^{m,B})}$:

$\mathbf{A}_1, \ldots, \mathbf{A}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{(k+1)\times k}$
$\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times(k+1)}$
$\mathbf{V}_1, \ldots, \mathbf{V}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k\times(k+1)}$
$\mathbf{z}_1, \ldots, \mathbf{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k}$
$\mathsf{mpk} := \left\{ [\mathbf{A}_i]_1, [\mathbf{W}_i\mathbf{A}_i]_1 \right\}_{i\in[n]}$
$\mathsf{msk} := \left\{ \mathbf{W}_i, \mathbf{V}_i, \mathbf{z}_i \right\}_{i\in[n]}$
return $(\mathsf{mpk}, \mathsf{msk})$

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

pick $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$;
$([\mathbf{c}'_i]_1, [\mathbf{c}''_i]_1, [\mathbf{c}_i]_1) := ([\mathbf{x}_i + \mathbf{W}_i\mathbf{A}_i\mathbf{s}_i]_1, [\mathbf{z}_i + \mathbf{V}_i\mathbf{A}_i\mathbf{s}_i]_1, [\mathbf{A}_i\mathbf{s}_i]_1)$
return $([\mathbf{c}'_i]_1, [\mathbf{c}''_i]_1, [\mathbf{c}_i]_1)$

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

pick $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$
$\mathbf{d}_i := \mathbf{W}_i^\top \mathbf{y}_i + \mathbf{V}_i^\top \mathbf{r}$
$z := \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$
return $\left(\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T \right)$

$\underline{\mathsf{Dec}\left(\left(\left\{ [\mathbf{d}_i]_2 \right\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T \right), \mathbf{y}_1 \| \cdots \| \mathbf{y}_n, \left\{ [\mathbf{c}'_i]_1, [\mathbf{c}''_i]_1, [\mathbf{c}_i]_1 \right\}_{i\in[n]} \right)}$:

$\mathbf{out} \leftarrow \left(\sum_i e([\mathbf{c}'_i]_1, [\mathbf{y}_i]_2) \cdot e([\mathbf{c}''_i]_1, [\mathbf{r}]_2) / e([\mathbf{c}_i]_1, [\mathbf{d}_i]_2) \right) / [z]_T$
return discrete log of $\mathbf{out}$

Fig. 10: Our private-key MIFE scheme for the class $\mathcal{F}_n^{m,B}$ (self-contained description). The scheme is many-SEL-IND-secure under the k-Linear Assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$. We use $e([\mathbf{X}]_1, [\mathbf{Y}]_2)$ to denote $[\mathbf{X}^\top\mathbf{Y}]_T$.

[6] S. Badrinarayanan, D. Gupta, A. Jain, and A. Sahai. Multi-input functional encryption for unbounded arity functions. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 27–51. Springer, Heidelberg, Nov. / Dec. 2015.

[7] A. Bishop, A. Jain, and L. Kowalczyk. Function-hiding inner product encryption. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Heidelberg, Nov. / Dec. 2015.

[8] N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In V. Guruswami, editor, *56th FOCS*, pages 171–190. IEEE Computer Society Press, Oct. 2015.

[9] O. Blazy, E. Kiltz, and J. Pan. (Hierarchical) identity-based encryption from affine message authentication. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 408–425. Springer, Heidelberg, Aug. 2014.

[10] D. Boneh, K. Lewi, M. Raykova, A. Sahai, M. Zhandry, and J. Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 563–594. Springer, Heidelberg, Apr. 2015.

[11] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, Mar. 2011.

$\underline{\mathsf{Setup}(\mathbb{G}, \mathcal{F}_1^{m,B})}$:

$\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{(k+1)\times k}, \mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times(k+1)}$

$\mathsf{mpk} := ([\mathbf{A}], [\mathbf{WA}]), \mathsf{msk} := \mathbf{W}$;

return $(\mathsf{mpk}, \mathsf{msk})$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y} \in \mathbb{Z}_q^m)}$:

return $\mathsf{sk}_{\mathbf{y}} := \mathbf{W}^\top \mathbf{y} \in \mathbb{Z}_q^{k+1}$

$\underline{\mathsf{Enc}(\mathsf{msk}, \mathbf{x} \in \mathbb{Z}_q^m)}$:

pick $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$;

return $([\mathbf{c}'], [\mathbf{c}]) := ([\mathbf{x} + \mathbf{WAs}], [\mathbf{As}])$

$\underline{\mathsf{Dec}(\mathsf{sk}_{\mathbf{y}}, \mathbf{y}, ([\mathbf{c}'], [\mathbf{c}]))}$:

return discrete log of $[\mathbf{c}'^\top \mathbf{y} - \mathbf{c}^\top \mathsf{sk}_{\mathbf{y}}]$

Fig. 11: A one-SEL-SIM scheme for single-input inner product $\mathcal{F}_1^{m,B}$ from [25]. Note that the scheme is public-key since Enc only needs to know mpk and not msk. Under the k-Linear Assumption, this scheme is one-SEL-SIM-secure and thus many-SEL-IND-secure.

[12] Z. Brakerski, I. Komargodski, and G. Segev. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. In *EUROCRYPT 2016*, LNCS. Springer, Heidelberg, 2016.

[13] P. Datta, R. Dutta, and S. Mukhopadhyay. Functional encryption for inner product with full function privacy. In C.-M. Cheng, K.-M. Chung, G. Persiano, and B.-Y. Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 164–195. Springer, Heidelberg, Mar. 2016.

[14] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, Aug. 2013.

[15] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, Heidelberg, May 2013.

[16] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, Oct. 2013.

[17] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Functional encryption without obfuscation. In E. Kushilevitz and T. Malkin, editors, *TCC 2016-A, Part II*, volume 9563 of *LNCS*, pages 480–511. Springer, Heidelberg, Jan. 2016.

[18] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou. Multi-input functional encryption. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Heidelberg, May 2014.

[19] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *45th ACM STOC*, pages 555–564. ACM Press, June 2013.

[20] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 162–179. Springer, Heidelberg, Aug. 2012.

[21] A. Sahai and H. Seyalioglu. Worry-free encryption: functional encryption with public keys. In E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, editors, *ACM CCS 10*, pages 463–472. ACM Press, Oct. 2010.

[22] A. Sahai and B. R. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005.

[23] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, Aug. 2009.

[24] H. Wee. Dual system encryption via predicate encodings. In Y. Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 616–637. Springer, Heidelberg, Feb. 2014.

[25] H. Wee. Simulation-based inner product functional encryption. Private communication, 2016.

A One-SEL-SIM Scheme for Single-Input Inner Product

We include a brief description of the analysis and simulator from [25] since the manuscript is not publicly available. The scheme is in fact essentially the same as those in [4, 2], extended explicitly to the k-Lin assumption. The analysis also starts out the same way, by using the k-Lin assumption to replace $[\mathbf{As}]$ in the challenge ciphertext with $[\mathbf{c}]$, so that the ciphertext is given by $([\mathbf{x}^* + \mathbf{Wc}], [\mathbf{c}])$, where $\mathbf{x}^*$ denotes the selective challenge. Next, we perform a change of variables $\widetilde{\mathbf{W}} = \mathbf{W} + \frac{1}{\langle \mathbf{c}, \mathbf{a}^\perp \rangle} \mathbf{x}^* (\mathbf{a}^\perp)^\top$, so that

$$\begin{aligned} \mathbf{WA} &= \widetilde{\mathbf{W}}\mathbf{A} \\ \mathbf{x}^* + \mathbf{Wc} &= \widetilde{\mathbf{W}}\mathbf{c} \\ \mathbf{W}^\top \mathbf{y} &= \widetilde{\mathbf{W}}^\top \mathbf{y} - \frac{\langle \mathbf{x}^*, \mathbf{y} \rangle}{\langle \mathbf{c}, \mathbf{a}^\perp \rangle} \mathbf{a}^\perp \end{aligned}$$

which coincides precisely with the output of the simulator.

$\widetilde{\mathsf{Setup}}(\mathbb{G})$:

$\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{(k+1)\times k}$, $\widetilde{\mathbf{W}} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times(k+1)}$

compute $\mathbf{a}^\perp \in \mathbb{Z}_q^{k+1} \setminus \{\mathbf{0}\}$ s.t. $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}$

$\widetilde{\mathsf{mpk}} := ([\mathbf{A}], [\widetilde{\mathbf{W}}\mathbf{A}])$, $\widetilde{\mathsf{msk}} := (\mathbf{A}, \widetilde{\mathbf{W}})$;

return $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{msk}})$

$\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \mathbf{y} \in \mathbb{Z}_q^m, a \in \mathbb{Z}_q)$:

return $\mathsf{sk}_{\mathbf{y}} := \widetilde{\mathbf{W}}^\top \mathbf{y} - \frac{a}{\langle \mathbf{c}, \mathbf{a}^\perp \rangle} \mathbf{a}^\perp \in \mathbb{Z}_q^{k+1}$

$\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}})$:

pick $\mathbf{c} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k+1}$;

return $([\widetilde{\mathbf{W}}\mathbf{c}], [\mathbf{c}])$

Fig. 12: Simulator for one-SEL-SIM scheme for single-input inner product $\mathcal{F}_1^{m,B}$ from [25].

B Additional Definitions

In this section we provide many-AD-IND security definition for MIFE in the public key setting where the adversary holds encryptions keys for a subset of the MIFE encryption slots. We first define the MIFE functionality that allows public encryption keys for some slots

Definition 4 (Multi-input Function Encryption). *Let $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$ 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:*

- $\mathsf{Setup}(1^\lambda, \mathcal{F}_n)$*: on input the security parameter* λ *and a description of* $\mathcal{F}_n \in \mathcal{F}$ *outputs* n *encryption keys* $\mathsf{ek}_1, \ldots, \mathsf{ek}_n$ *and a master secret key* msk.
- $\mathsf{Encrypt}(\mathsf{ek}_i, i, m)$*: on input* $\mathsf{ek}_i \in \{\mathsf{ek}_1, \ldots, \mathsf{ek}_n\}$ *and a message* $m \in \mathcal{X}_i$ *outputs a ciphertext* ct *for the* i*-th MIFE slot. We assume that each ciphertext has an associated index* i*, which denotes what slot this ciphertext can be used for.*
- $\mathsf{KeyGen}(\mathsf{msk}, f)$*: on input the master secret key* msk *and a function* $f \in \mathcal{F}_n$ *outputs a decryption key* sk_f *that takes inputs encrypted under* $\mathsf{ek}_1, \ldots \mathsf{ek}_n$.
- $\mathsf{Decrypt}(\mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$*: on input a decryption key for function* f *and* sk_f *ciphertexts outputs a string* $y \in \mathcal{Y}$.

The scheme $\mathcal{MIFE}$ *is correct if for all* $f \in \mathcal{F}$ *and all* $x_i \in \mathcal{X}_i$ *for* $1 \leq i \leq n$*, we have*

$$\Pr\left[\begin{array}{r} (\mathsf{ek}_1, \ldots, \mathsf{ek}_n, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, n); \\ \mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f); \\ \mathsf{Decrypt}(\mathsf{sk}_f, \mathsf{Encrypt}(\mathsf{ek}_1, x_1), \ldots, \mathsf{Encrypt}(\mathsf{ek}_n, x_n)) = f(x_1, \ldots, x_n) \end{array}\right] = 1,$$

where the probability is taken over the coins of Setup, KeyGen *and* $\mathsf{Encrypt}$.

For out definition we will use the notion of I*-compatibility*, which is defined next.

Definition 5 (I-Compatibility). *Let* $\mathcal{F}_n = \{f\}$ *be a family of* n*-ary functions and let* $N = \{1, \ldots, n\}$, $I \subset N$. *The pairs of input vectors* $\mathbf{x}_0 = \left(\{x_1^{j,0}\}_{j=1}^{q_1}, \ldots, \{x_n^{j,0}\}_{j=1}^{q_n}\right)$ *and* $\mathbf{x}_1 = \left(\{x_1^{j,1}\}_{j=1}^{q_1}, \ldots, \{x_n^{j,1}\}_{j=1}^{q_n}\right)$ *are* I*-compatible if they satisfy the following property:*

For every $f \in \{f\}$*, every* $I' = \{i_1, \ldots, i_t\} \subseteq I \cup \emptyset$*, every* $j_1 \in [q_1], \ldots, j_{n-t} \in [q_n]$*, for every* $x'_{i_1} \in \mathcal{X}_{i_1}, \ldots, x'_{i_t} \in \mathcal{X}_{i_t}$ *the following two distributions are indistinguishable:*

$$f\left(\pi(x_{i_1}^{j_1,0}, \ldots, x_{i_{n-t}}^{j_{n-t},0}, x'_{i_1}, \ldots, x'_{i_t})\right) = f\left(\pi(x_{i_1}^{j_1,1}, \ldots, x_{i_{n-t}}^{j_{n-t},1}, x'_{i_1}, \ldots, x'_{i_t})\right),$$

where $\pi(y_{i_1}, \ldots, y_{i_n})$ *denotes a permutation of the values* $y_{i_1}, \ldots, y_{i_n}$ *such that the value* y_{i_j} *is mapped to the* l*'th location of* y_{i_j} *is the* l*'th input (out of* n *input) to* f.

Using the above notions of I-compatibility the indistinguishability security for MIFE is defined as follows.

Definition 6 (many-AD-IND-secure MIFE). *A multi-input functional encryption* $\mathcal{MIFE}$ *for* n*-ary functions* $\mathcal{F}$ *is* (t, q)*-IND secure if for every PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$*, the advantage of* $\mathcal{A}$ *defined as*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathcal{MIFE},IND} = |\Pr[\mathbf{IND}_{\mathcal{A},1}^{\mathcal{MIFE}}(1^\lambda) = 1] - \Pr[\mathbf{IND}_{\mathcal{A},0}^{\mathcal{MIFE}}(1^\lambda) = 1]|$$

is negligible and the experiment $\mathbf{IND}_{\mathcal{A},\beta}^{\mathcal{MIFE}}(1^\lambda)$ *is defined as follows:*

$\underline{\textit{Experiment}}\ \mathbf{IND}^{\mathcal{MIFE}}_{\mathcal{A},\beta}(1^\lambda)$:
$(I, \mathsf{st}_0) \leftarrow \mathcal{A}_0(1^\lambda)$ *where* $|I| = t$
$(\{\mathsf{ek}_i\}_{i\in[n]}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathbf{X}^0, \mathbf{X}^1, \mathsf{st}_1) \leftarrow \mathcal{A}_1^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{st}_0, \{\mathsf{ek}_i\}_{i\in I})$ *for* $\left\{\mathbf{X}^d = \{x_1^{j,d}, \ldots, x_n^{j,d}\}_{j=1}^q\right\}_{d=1,2}$
$\mathsf{ct}_{i,j} \leftarrow \mathsf{Encrypt}(\mathsf{ek}_i, x_i^{j,\beta})\ \forall i \in [n], j \in [q]$
$\beta' \leftarrow \mathcal{A}_2^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{st}_1, \mathbf{ct})$
Output: β'

In the above experiment we require that all queries for $\{f\}$ for decryption functions made by $\mathcal{A}_1$ and $\mathcal{A}_2$ must be I-compatible with $\mathbf{X}^0$ and $\mathbf{X}^1$ according to Definition 5.

We call the MIFE scheme public key many-AD-IND-secure when $t = n$, i.e., the adversary obtain the encryption keys for all slots.

Since our paper focuses on multi-input functional encryption for inner product functions, we note that the I-compatibility condition in this setting becomes as follows. Each function f_i is defined by a vector $f = (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$. Then, if $\overline{I}$ is the set of the inputs for which the adversary does not receive encryption keys, we require that for every $j_1, j_1' \in [q_1], \ldots, j_{n-t}, j_{n-t}' \in [q_n]$ and every $J \subseteq [n]$ such that $\overline{I} \subseteq J$:

$$\sum_{i\in J} \langle \mathbf{x}_i^{j_i,0}, \mathbf{y}_i \rangle = \sum_{i\in J} \langle \mathbf{x}_i^{j_i',1}, \mathbf{y}_i \rangle.$$

This restriction is due to the fact that the adversary can always encrypt the all 0's vectors in the slots for which it holds encryption keys. Thus, if the above equality does not hold, the adversary will be able to distinguish the challenges trivially.

We note that in the case of public key MIFE, the above restriction amounts to

$$\langle \mathbf{x}_i^{j_i,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j_i',1}, \mathbf{y}_i \rangle \quad \forall i \in [n],$$

since $\overline{I} = \emptyset$ and the sets J include every singleton set.

C Extensions

C.1 Public-Key MIFE for Inner Product

This is easy, as discussed in the introduction. Run n independent copies of the single-input scheme; use the i'th copy to encrypt $\mathbf{x}_i$ in the i'th slot; and the new secret key is the collection of the n secret keys corresponding to each of $\mathbf{y}_1, \ldots, \mathbf{y}_n$. Decryption recovers $\langle \mathbf{x}_1, \mathbf{y}_1 \rangle, \ldots \langle \mathbf{x}_n, \mathbf{y}_n \rangle$ and returns the sum of these values. This means that the adversary also learns each of $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ but that is inherent leakage from the ideal functionality. Concretely, an adversary can always pad an encryption of $\mathbf{x}_i$ in the i'th slot with encryptions of $\mathbf{0}$'s in the remaining $n-1$ slots and then decrypt with a key for $\mathbf{y}_1, \ldots, \mathbf{y}_n$ to learn $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. Security is immediate since the underlying FE guarantees that there is no leakage beyond $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$.

C.2 Impossibility for many-SEL-SIM.

The impossibility result for simulation-based IBE in [11, 3] applies to this setting, even for private-key schemes with "selective" security in the single-input setting for one-dimensional inner product over bits. Suppose the secret key has ℓ bits. The adversary starts by requesting for $\ell+\lambda$ encryptions of random bits. At this point, the simulator needs to simulate these $\ell+\lambda$ ciphertexts without having learned anything from the ideal functionality. The adversary then requests for a secret key for 1, and the simulator needs to produce an ℓ-bit secret key that decrypts the simulated ciphertext to an arbitrary sequence of $\ell+\lambda$ bits. This is impossible even for a computationally unbounded adversary.