Indistinguishability Obfuscation from Compact Functional Encryption

Prabhanjan Ananth[1] and Abhishek Jain[2(✉)]

[1] University of California, Los Angeles, USA
prabhanjan@cs.ucla.edu
[2] Johns Hopkins University, Baltimore, USA
abhishek@cs.jhu.edu

Abstract. The arrival of indistinguishability obfuscation (iO) has transformed the cryptographic landscape by enabling several security goals that were previously beyond our reach. Consequently, one of the pressing goals currently is to construct iO from well-studied standard cryptographic assumptions.

In this work, we make progress in this direction by presenting a reduction from iO to a natural form of public-key functional encryption (FE). Specifically, we construct iO for general functions from any single-key FE scheme for NC^1 that achieves selective, indistinguishability security against sub-exponential time adversaries. Further, the FE scheme should be *compact*, namely, the running time of the encryption algorithm must only be a polynomial in the security parameter and the input message length (and not in the function description size or its output length).

We achieve this result by developing a novel *arity amplification technique* to transform FE for single-ary functions into FE for multi-ary functions (aka multi-input FE). Instantiating our approach with known, non-compact FE schemes, we obtain the first constructions of multi-input FE for constant-ary functions based on standard assumptions.

Finally, as a result of independent interest, we construct a compact FE scheme from randomized encodings for Turing machines and learning with errors assumption.

1 Introduction

The ability to cryptographically obfuscate computer programs holds great prospects for securing the future digital world. While general-purpose program

P. Ananth—Research supported in part from a DARPA/ONR PROCEED award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

A. Jain—The author is partly funded by NSF CNS-1414023. Supported in part by a DARPA Safeware grant and NSF CNS-1414023.

R. Gennaro and M. Robshaw (Eds.): CRYPTO 2015, Part I, LNCS 9215, pp. 308–326, 2015.
DOI: 10.1007/978-3-662-47989-6_15

obfuscation remained an elusive goal for several decades, this changed recently with the seminal work of Garg et al. [26] who gave the first candidate construction of indistinguishability obfuscation [8] (iO) for $\mathsf{P/poly}$. Since then, iO has been used to realize several advanced cryptographic primitives that were previously beyond our reach, including deniable encryption [45], collusion-resistant functional encryption [26], round-optimal multiparty computation [25], and so on. Indeed, by now, iO has been established as a central hub of cryptography.

The tremendous appeal of iO motivates the goal of constructing it from well-studied, standard cryptographic assumptions. However, not much is known in this direction. The security of candidate iO constructions, initiated by [7][1], is proven in the "generic graded encoding model" and lacks a reduction in the standard model. The recent works of Pass et al. [43] and Gentry et al. [33] seek to rectify this situation by constructing iO from various assumptions on multilinear maps [24]. In particular, Pass et al. [43] reduce the security of their construction to an "uber assumption" on multilinear maps while Gentry et al. [33] provide a reduction to the "multilinear subgroup elimination assumption" (stated in their paper) on composite-order multilinear maps [22].

Till date, these remain the only known constructions of general-purpose iO. Further, all of them rely on a common cryptographic primitive, namely, multilinear maps. This is an unsatisfactory situation, especially in light of several recent attacks on multilinear maps [13,21,23,30]. This calls for new constructions of iO from other, more familiar cryptographic primitives.

1.1 This Work

In this work, we make progress in this direction by providing a new construction of iO based on a natural form of functional encryption (FE). Along the way, we also obtain new results on multi-input FE [35] that significantly improve upon the prior results.

I. Indistinguishability Obfuscation from Compact FE. Our main result is a reduction from iO to any public-key functional encryption scheme that satisfies a natural "compactness" requirement on the encryption algorithm. Specifically, we give a construction of iO for $\mathsf{P/poly}$ from any public-key FE scheme for NC^1 that satisfies the following requirements:

- *Security*: It supports *one* key query and achieves selective, indistinguishability security against sub-exponential time adversaries.
- *Compactness*: For any input message x, the running time of the encryption algorithm is polynomial in the security parameter and the size of x. In particular, it does not depend on the circuit description size or the output length of any function f supported by the scheme.[2] We call such an FE scheme *compact*.

[1] See the full version for a comprehensive list of works.
[2] The compactness requirement can be further relaxed.

We stress that we do *not* require function hiding property [1,11] from the underlying FE. Indeed, function-hiding public-key FE already implies iO.

On the use of Sub-exponential Hardness. Our reliance on sub-exponential hardness of the underlying FE scheme is similar in spirit to the use of sub-exponential hardness assumptions in the work of Gentry et al. [33]. Indeed, as discussed in their paper, the use of sub-exponential hardness assumptions "seems" inherent to realizing iO. We note, however, that to the best of our knowledge, no formal proof supporting this intuition is known.

On the Existence of compact FE. While public-key FE is an extremely well-studied notion, somewhat surprisingly, compact FE has remained largely unexplored. Previously, Goldwasser et al. [36] studied the notion of "succinct" FE which, informally speaking, requires that the size of any ciphertext must be independent of the function description size. We note, however, that this notion does not preclude dependence on the function output length. Indeed, [36] focuses on functions with single bit output, and their construction does not achieve our desired compactness property for the case of functions with long output.

Compact FE with simulation-based security is known to be impossible for general functions [2,20]. Concretely, in the case of adaptive simulation security, the impossiblity result holds for a single key and message query. In the selective security case, it holds for a single key and unbounded message queries.[3] However, we stress that for our main result, we only require the underlying compact FE scheme to satisfy *indistinguishability security* in the selective model.

Presently, the only known constructions of compact FE for general functions rely on iO [26,46].[4] In contrast, *non-compact* FE can be based on LWE [36], or even semantically-secure public-key encryption [37,44].

We hope that this work will bring attention to the natural goal of compactness in FE and that it will be realized from standard complexity assumptions in the future. With this view, we believe that the results in this work open new doors to the eventual goal of realizing iO from well-studied cryptographic assumptions, possibly avoiding multilinear maps altogether.

II. A Technique for Arity Amplification. At the heart of our results is a novel technique for *arity amplification* in secret-key multi-input functional encryption (MiFE), a notion introduced by Goldwasser et al. [35]. Specifically, we show how to transform a selectively-secure secret-key MiFE scheme for i-ary functions into another selectively-secure[5] secret-key MiFE scheme for $(i+1)$-ary functions. Interestingly, we achieve this by "knitting together" a secret-key FE scheme for

[3] A related notion of reusable garbled circuits with output-size independence was recently studied by Gentry et al. [31]. They proved an analogous impossibility result for this notion in the case of simulation security.

[4] The compact FE constructions of [26,46], in fact, achieve stronger security than what we require. Specifically, they achieve security against unbounded key queries, while we only require security against a single key query.

[5] Unless stated otherwise, we only consider selectively-secure (Mi)FE schemes in the subsequent discussion.

i-ary functions with a public-key FE scheme for 1-ary functions. In order to prove the security of our transformation, we build on program puncturing techniques that were first introduced by Sahai and Waters [45] in the context of iO and recently developed in the context of secret-key FE by Brakerski and Segev [16] and Komargodski et al. [40].

Starting from a secret key FE scheme for single-ary functions (aka single-input FE) and applying our transformation *iteratively*, we obtain a secret-key MiFE scheme for multi-ary functions. This iterated procedure is sensitive to the efficiency of the underlying single-input FE and yields different end results depending upon whether the underlying FE scheme is compact or not.

More concretely, given a compact *single-key* FE scheme for NC^1, we first convert it into a compact FE scheme for general functions that supports an a priori bounded polynomial number of key queries. This process involves multiple steps, including the key query amplification step of Gorbunov et al. [37] and the generic transformation from [4,31] for boosting the function family from NC^1 to general functions.

Then, instantiating our iterated approach with a sub-exponentially secure *compact* FE scheme that supports (say) q number of key queries, we obtain a secret-key MiFE scheme for polynomial-arity functions that supports q key queries and q message queries. Instantiating this result for the case of $q = 2$ and combining it with the MiFE to iO transformation of Goldwasser et al. [35], we obtain iO for general functions.

III. MiFE for Functions with Small Arity from Standard Assumptions. We also analyze our transformation for the case when the underlying FE scheme is *non-compact*. Recall that in such a scheme, the running time of the encryption algorithm may depend upon the function description size [37,44] or its output length [36].

Bounded-Message Security from Standard Assumptions. Starting with a non-compact FE scheme that supports an a priori bounded polynomial (say) q number of key queries, we obtain a secret-key MiFE scheme for *constant-ary* functions that supports q message and q key queries. Instantiating the underlying FE scheme with [37,44], we obtain the above result based on *semantically secure public-key encryption*.[6] This significantly improves over the state of the art in this area in terms of security assumptions. In particular, prior constructions of such an MiFE scheme either rely upon iO [35] or lack a security proof in the standard model [10].

Unbounded-Message Security from iO. Starting with a non-compact FE scheme that achieves security against unbounded key queries, we obtain a secret-key

[6] At the cost of further decreasing the efficiency of encryption and restricting our attention to a single key query, we can, in fact, obtain this result based on only *one-way functions*. This requires a slight modification in our construction and proof. In particular, to obtain this result, we must replace the underlying public-key FE with a secret-key FE and then leverage the "one-shot" proof technique discussed in Sect. 1.2. We defer the details to the full version of the paper.

MiFE scheme for constant-ary functions that supports unbounded message and key queries.

Presently, known constructions of public-key FE with security against unbounded collusions rely upon iO and one-way functions [26,46] or specific assumptions on composite-order multilinear maps [27]. Then, instantiating the underlying FE scheme in our construction with [26], we obtain a secret-key MiFE scheme for functions with constant arity that supports *unbounded* number of message and key queries based on iO and one-way functions. Previously, such an MiFE scheme [35] was only known based on differing-inputs obfuscation [3,8,14].

On the Optimality of Our Results. It is easy to see that a secret-key MiFE scheme for 2-ary functions that supports a single key query and *unbounded* message queries already implies a secret-key single-input FE scheme that supports *unbounded* key and message queries. This observation is already implicit in [35].

In light of the above, we note that our results on secret-key MiFE with bounded message queries are essentially *optimal*.

IV. Compact FE from Randomized Encodings for Turing Machines. Our final contribution is a construction of a single-key, compact FE scheme from the learning with errors (LWE) assumption and randomized encodings (RE) [6,38] for Turing machines where the size of the encoding only depends on the size of the Turing machine (TM) and not on its running time or the output length. Combining this with our reduction from iO to compact FE, we get a construction of iO for general circuits from sub-exponentially secure RE for Turing machines and LWE.

Randomized encodings for circuits are known to exist from only one-way functions [47]. In contrast, the problem of RE for TMs has received far less attention. Recently, a few works [28,29,42] construct RE for RAM programs from only one-way functions; however, the size of the garbled RAM program in these schemes is proportional to the (worst-case) running time of the underlying RAM program. Even more recently, [9,18,41] give constructions of RE for TMs where the encoding size is independent of the running time of TM. However, all of these results are based on iO.

We hope that our work will bring more attention to this natural goal, and that it can be realized from standard cryptographic assumptions in the future. This result is presented in the full version [5].

1.2 Our Techniques

Main Goal: Arity Amplification. The starting point of our iO construction is the recent work of Goldwasser et al. [35] who showed a transformation from secret-key MiFE to iO. Concretely, [35] proved that secret-key MiFE for $(n+1)$-ary functions that supports a *single* key query and 2 message queries implies iO for all functions with input length n. Very roughly, in order to obfuscate a function f with input length n, their idea is to use the first MiFE ciphertext to

hide the function and use the remaining n positions to encode f's input domain *à la* Yao's garbled circuits [47]. This, coupled with a secret key for the universal circuit yields an indistinguishability obfuscation of f.

Given their result, our goal of constructing general-purpose iO from public-key single-input FE reduces to the task of constructing secret-key MiFE scheme for *polynomial-ary* functions from a public-key FE for *single-ary* functions. To help the presentation, we ignore the succinctness requirements on the underlying FE for now, and revisit it later.

At a first glance, it is not clear at all how to proceed towards realizing the above goal.

Knitting together Two FE Instances. Towards that end, let us first consider a weaker goal of constructing secret-key MiFE for 2-ary functions. Roughly speaking, our main idea is to "knit" together an instance of a *secret-key* single-input FE scheme with an instance of *public-key* single-input FE to obtain a secret-key MiFE for 2-ary functions. Here, the importance of using *both* a secret-key FE and a public-key FE will become clear once we explain our approach.

More concretely, the 2-ary MiFE scheme is constructed as follows:

- The master secret key of the 2-ary scheme consists of a key pair $(\mathsf{pk}, \mathsf{msk})$ of the underlying public-key FE scheme as well as a master secret key msk' of the underlying secret-key FE scheme. Further, a key for a function f is computed as a key K_f of the underlying public-key FE scheme for f.
- In order to encrypt a message m_1 corresponding to the *first* position, we generate (using msk') a function key of the underlying secret-key FE scheme for the following function $\mathcal{G}^{\mathsf{enc}}_{[m_1,K,pk]}$: it contains the message m_1, a key for a pseudorandom function (PRF) K, and the public key pk hardwired in its description. On input a message (m_2, tag), $\mathcal{G}^{\mathsf{enc}}_{[m_1,K,pk]}$ outputs an encryption (using pk) of the combined message $m_1 \| m_2$ w.r.t. the underlying public-key FE. Here, the randomness r for encryption is derived as $r \leftarrow \mathsf{PRF}_K(\mathsf{tag})$.

  A message m_2 corresponding to the *second* position is encrypted (along with a random tag) using the encryption algorithm of the underlying secret-key FE scheme.
- In order to decrypt a pair of ciphertexts (c_1, c_2) using a function key K_f, we first decrypt c_2 using c_1 (recall that c_1 corresponds to a function key of the secret key FE scheme) to produce a new ciphertext $\tilde{c}$ corresponding to the underlying public-key FE scheme. Finally, we decrypt $\tilde{c}$ using K_f to get the desired output.

The correctness of the above construction is easy to verify. A careful reader, however, may immediately notice a security problem. Note that in order to prove security, we must ensure that the first ciphertext hides the message m_1 and the PRF key K. However, this is not necessarily guaranteed by the above construction.

We solve this problem by building upon the recent elegant result of Brakerski and Segev [16] who give a generic transformation from any single-input secret-key FE scheme into another secret-key FE scheme that satisfies *function hiding*.

Specifically, instead of using a standard secret-key FE, we will use a function-hiding secret-key FE in the above construction. We then rely upon the function-hiding property of the function key to argue that m_1 and K remain hidden. As we will see later, this technique, when generalized to the MiFE setting, is vital to our overall approach.

We highlight another subtle point in the above construction: suppose that we want the 2-ary MiFE scheme to support $q \geq 2$ message queries. Then, since the function keys of the underlying secret-key FE scheme act as ciphertexts in the 2-ary MiFE scheme, we need the underlying secret-key FE scheme to, in fact, support q key queries. To obtain such an FE scheme, we leverage [37] to transform a single-key FE scheme into a q-key FE scheme. We refer the reader to the full version for more details.

Overview of Proof Strategy. Proving the security of the above construction turns out to be quite non-trivial. Suppose that we wish to prove security for q message queries (for each position), say $\{x_i^0, y_i^0\}_{i=1}^q$, $\{x_i^1, y_i^1\}_{i=1}^q$. Further, for simplicity, let us restrict our attention to a single function key query f. One plausible proof strategy would be to construct a sequence of roughly q hybrids where at any step $i \in [q]$, we switch from (x_i^0, y_i^0) to (x_i^1, y_i^1). However, note that in the case of MiFE, an adversary can compute "cross-terms" from the challenge message pairs. That is, the adversary is allowed to compute (x_i^b, y_i^b) for any $i, j \in [q]$. Indeed, this is why the security definition of MiFE requires that $f(x_i^0, y_j^0) = f(x_i^1, y_j^1)$ for all $i, j \in [q]$. However, note that in the above proof strategy, the adversary might end up computing $f(x_i^1, y_j^0)$ which will allow him to distinguish between two successive hybrids.

A plausible solution to overcome the above problem is to argue indistinguishability in *one shot*. That is, instead of arguing indistinguishability one message-pair at a time, we instead switch all the challenge message pairs corresponding to challenge bit 0 with the ones corresponding to challenge bit 1. Implementing this strategy successfully, however, will require "hardwiring" and "unhardwiring" of the (public-key) encryption of *all* the q^2 message pairs (x_i^b, y_j^b) (each of which corresponds to a different output) in the challenge ciphertexts for the first position that correspond to function keys of the underlying secret-key FE scheme. While this is tolerable for the case of arity 2 (and more generally for constant arity), it quickly becomes prohibitive for large arity. Indeed, for arity $n = \mathsf{poly}(\lambda)$, the number of possible outputs (and therefore the message pair combinations) is exponential.

We solve the above problems by carefully employing a "one-input-at-a-time" strategy where we consider roughly q^2 intermediate hybrids (and q^n in the case of arity n; see below). Very briefly, our proof involves careful hardwiring and unhardwiring of the (public-key) encryption of each of the q^2 message pairs (x_i^b, y_j^b), *one at a time*, in the challenge ciphertexts for the first position that correspond to function keys of the underlying secret-key FE scheme. Furthermore, we crucially ensure that the adversary cannot learn an output of the form $f(x_i^0, y_j^1)$ at any point in the hybrids. In order to implement these ideas, we rely upon *program puncturing techniques* that were originally introduced in the context of $i\mathrm{O}$ [45]

and recently developed in the secret-key FE setting by [16,40]. In particular, as in the work of [40], we rely on function hiding property of the underlying secret-key FE scheme to argue indistinguishability of these core hybrids. We finally note that our proof strategy bears resemblance to the proof methodology in several recent works [9,18,19,32,33,41].

Note that in the above proof strategy, it was crucial that we use a *public-key* FE in our construction. To see this, suppose we were to replace the public-key FE with an instance of a secret-key FE, referred to as $\mathcal{FE}$ (while the other secret-key FE instance used in the construction is referred to as $\mathcal{FE}'$). Note that now, the challenge ciphertexts corresponding to the first position would contain the master secret key (say) msk of $\mathcal{FE}$. Then, in order to execute the aforementioned proof strategy, it would seem that we need to somehow "puncture" msk such that it allows encryption all messages except a select message (say) $x_i^b \| y_j^b$. Furthermore, the punctured msk *should not allow generation of any function keys*, except K_f. However, it is not clear how to realize such a notion of secret-key FE. By using public-key FE, we are able to bypass the above difficulties since by definition, the public key does not need to be hidden.

Climbing the Arity Ladder. The above approach can be generalized to transform a secret-key MiFE scheme for i-ary functions into a secret-key MiFE scheme for $(i+1)$-ary functions. Concretely, this transformation consists of two steps: first, by using ideas from [16], we add function privacy property to the i-ary MiFE scheme. Next, we combine the resultant scheme with a "fresh" instance of a public-key single-input FE scheme to obtain an $(i+1)$-ary MiFE scheme.

In more detail, as in the 2-ary case, the ciphertext corresponding to the first position will consist of a function key of the underlying (function private) c-ary MiFE scheme for the function $\mathcal{G}^{\mathsf{enc}}_{[m_1,K,pk]}$ which is defined similarly to the 2-ary case, except that here it takes as input messages $m_2, \ldots, m_{i+1}$ (along with random tags) and outputs an encryption (using pk) of the combined message $m_1 \| \ldots \| m_{i+1}$ w.r.t. the underlying public-key FE. The ciphertexts corresponding to remaining i positions will correspond to ciphertexts of the underlying c-ary MiFE scheme. The function key for a function f in the $c+1$-ary scheme will correspond to a key K_f for the same function f of the underlying public-key single-input FE scheme.

By applying the above ideas iteratively, we can transform a secret-key single-input FE into a secret-key multi-input FE. Our iterated construction is depicted in Fig. 1. The security of the construction follows along the same lines as discussed above.

The Role of Compactness. Upon "unrolling" our construction of n-ary MiFE scheme, one can observe that it involves n instances of a single-input FE scheme. Specifically, in the n-ary MiFE scheme, each of the ciphertexts corresponding to the first $n-1$ positions corresponds to a function key of (a different instance of) a single-input FE, while the ciphertext corresponding to the nth position corresponds to a ciphertext of a single-input FE scheme. The function key at position $n-1$ computes an encryption corresponding to the function key at

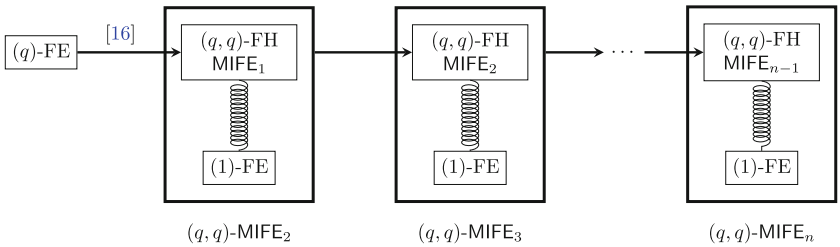

Fig. 1. The Iterated Construction. (q)-FE denotes a single-input public-key FE scheme that supports q key queries. (q_1, q_2)-MIFE_i denotes a secret-key MiFE scheme for i-ary functions that supports q_1 key and q_2 message queries. Finally, FH refers to function hiding.

position $n-2$ which in turn computes an encryption corresponding to the function key at position $n-3$, and so on.

With the above view, it is easy to see that the complexity of the above construction becomes prohibitive for $n = \omega(1)$ when it is instantiated with a non-succinct FE scheme. On the other hand, instantiating the construction with a succinct FE scheme allows us to go all the way to $n = \mathsf{poly}(\lambda)$.

We remark that the above discussion is oversimplified. We refer the reader to the technical parts of the paper (and the full version [5]) for more details.

2 Preliminaries

Throughout the paper, we denote the security parameter by λ. We assume that the reader is familiar with basic cryptographic concepts [34].

Given a PPT sampling algorithm A, we use $x \xleftarrow{\$} A$ to denote that x is the output of A when the randomness is sampled from the uniform distribution.

Punctured Pseudorandom Function Families. The works of [12,15,39] constructed a strengthening of PRF families that is commonly known as *punctured* pseudorandom function families. Unlike the standard notion of PRFs, this primitive is accompanied by a puncturing algorithm that takes as input x, a PRF key K and outputs a punctured key K_x that allows one to evaluate the output of PRF on any input other than x. The security guarantee states that the output of PRF on x is indistinguishable from random even if the adversary gets a key punctured on x.

2.1 Indistinguishability Obfuscation

Here we recall the notion of indistinguishability obfuscation (iO) that was defined by Barak et al. [8].

Definition 2.1 (Indistinguishability Obfuscator (iO)). *A uniform PPT algorithm iO is called an indistinguishability obfuscator for a circuit class $\{\mathcal{C}_\lambda\}$,*

where $\mathcal{C}_\lambda$ consists of circuits C of the form $C : \{0,1\}^\lambda \to \{0,1\}$, if the following holds:

- **Completeness:** *For every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$, every input $x \in \{0,1\}^\lambda$ (i.e., it belongs to the input space of C), we have that*

$$\mathsf{Pr}[C'(x) = C(x) \ : \ C' \leftarrow i\mathrm{O}(\lambda, C)] = 1.$$

- **Indistinguishability:** *For any PPT distinguisher D, there exists a negligible function $\mathsf{negl}(\cdot)$ such that the following holds: for all sufficiently large $\lambda \in \mathbb{N}$, for all pairs of circuits $C_0, C_1 \in \mathcal{C}_\lambda$ such that $C_0(x) = C_1(x)$ for all inputs x, we have:*

$$\left|\mathsf{Pr}[D(i\mathrm{O}(\lambda, C_0)) = 1] - \mathsf{Pr}[D(i\mathrm{O}(\lambda, C_1)) = 1]\right| \leq \mathsf{negl}(\lambda)$$

2.2 Public-Key Functional Encryption

Syntax. Let $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ be ensembles where each $\mathcal{X}_\lambda$, $\mathcal{Y}_\lambda$ are sets of size, functions in λ. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be an ensemble where each $\mathcal{F}_\lambda$ is a finite collection of functions. Each function $f \in \mathcal{F}_\lambda$ takes as input a string $x \in \mathcal{X}_\lambda$ and outputs $f(x) \in \mathcal{Y}_\lambda$.

A public-key functional encryption (FE) scheme FE for $\mathcal{F}$ consists of four algorithms ($\mathsf{FE.Setup}$, $\mathsf{FE.KeyGen}$, $\mathsf{FE.Enc}$, $\mathsf{FE.Dec}$):

- **Setup.** $\mathsf{FE.Setup}(1^\lambda)$ is a PPT algorithm that takes as input a security parameter λ and outputs a public key, (master) secret key pair $(\mathsf{FE.pk}, \mathsf{FE.msk})$.
- **Key Generation.** $\mathsf{FE.KeyGen}(\mathsf{FE.msk}, f)$ is a PPT algorithm that takes as input a master secret key $\mathsf{FE.msk}$, a function $f \in \mathcal{F}_\lambda$ and outputs a functional key $\mathsf{FE}.sk_f$.
- **Encryption.** $\mathsf{FE.Enc}(\mathsf{FE.pk}, x)$ is a PPT algorithm that takes as input a public key $\mathsf{FE.pk}$, a message $x \in \mathcal{X}_\lambda$ and outputs a ciphertext ct.
- **Decryption.** $\mathsf{FE.Dec}(\mathsf{FE}.sk_f, \mathsf{ct})$ is a deterministic algorithm that takes as input a functional key $\mathsf{FE}.sk_f$, a ciphertext ct and outputs a string y.

The correctness property guarantees that the output of the decryption on input a functional key of $f \in \mathcal{F}_\lambda$ and a ciphertext of $x \in \mathcal{X}_\lambda$ yields $f(x) \in \mathcal{Y}_\lambda$.

Selective Security. We recall indistinguishability-based selective security for FE. This security notion is modeled as a game between the challenger and the adversary where the adversary can request functional keys and ciphertexts from the challenger. Specifically, the adversary can submit function queries f to the challenger and receive corresponding functional keys. It can also submit a message query of the form (x_0, x_1) and in response, the challenger encrypts message x_b and sends the ciphertext back to the adversary. The adversary wins the game if she can guess b with probability significantly greater than $1/2$ and if $f(x_0) = f(x_1)$ for all function queries f. The only constraint here is that the adversary has to declare the challenge messages at the beginning of the game itself.

The term (q_{key}, μ)-secure FE scheme refers to the setting where the adversary can request up to q_{key} queries and he can succeed in the game with probability at most μ.

Compactness. We now define the notion of *compact* FE that will play a central role in our main result on $i\mathrm{O}$. In a compact FE scheme, the running time of the encryption algorithm only depends on the security parameter and the input message length. In particular, it is independent of the complexity of the function family supported by the FE scheme. Note that a direct consequence of this is that the size of the public key must also be independent of the complexity of the function family.

Definition 2.2 (Compact FE). *Let $p(\cdot)$ be a polynomial. A (q_{key}, μ)-selectively secure public-key FE scheme $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$, defined for an input space $\mathcal{X} = \{\mathcal{X}_\lambda\}$ and function space $\mathcal{F} = \{\mathcal{F}_\lambda\}$ is said to be **compact** if for all $\lambda \in \mathbb{N}$, the size of any public key $\mathsf{FE.pk}$ is $p(\lambda)$, where $(\mathsf{FE.msk}, \mathsf{FE.pk}) \leftarrow \mathsf{FE.Setup}(1^\lambda)$, and the running time of the encryption algorithm $\mathsf{FE.Enc}$, on input 1^λ, $\mathsf{FE.pk}$ and a message $x \in \mathcal{X}_\lambda$, is $p(\lambda, q_{\mathsf{key}}, |x|)$.*

Remark 2.3. We can define the notion of unbounded compact FE in the same manner as above except that we now allow the number of key queries made by the adversary in the security game to be an arbitrary polynomial.

3 Function Private Multi-input Functional Encryption (MiFE)

The concept of multi-input functional encryption was proposed by Goldwasser et al. [35]. Standard FE only allows for computing on a single ciphertext, i.e., it only supports *single-ary* functions. In contrast, multi-input functional encryption (MiFE) allows for (joint) computation over multiple ciphertexts. In other words, it supports *multi-ary* functions.

Analogous to standard FE, one can consider MiFE in two settings, namely, public-key and secret-key setting.[7] In this work, we will restrict our attention to the *secret-key* setting.

The security notion we are interested in is stronger than the one considered in Goldwasser et al. [35]. We expect the functional keys to hide the function it is associated with. This concept, termed as *function privacy* was previously considered in the single ary private key FE setting by Brakerski-Segev [16]. We extend their notion to the multi-input functional encryption setting as well.

We first present the syntax of a MiFE scheme and later we formalize the function privacy property.

[7] Goldwasser et al. [35] also define a more general notion of MiFE where there is different encryption key for each input position. When the adversary knows all (resp., none of) the encryption keys, then this notion captures the public-key (resp., secret-key) setting.

Syntax. Let $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ be ensembles where each $\mathcal{X}_\lambda$, $\mathcal{Y}_\lambda$ are sets of size, functions in λ. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be an ensemble where each $\mathcal{F}_\lambda$ is a finite collection of n-ary functions. Each function $f \in \mathcal{F}_\lambda$ takes as input strings $x_1, \ldots, x_n$, where each $x_i \in \mathcal{X}_\lambda$, and outputs $f(x_1, \ldots, x_n) \in \mathcal{Y}_\lambda$.

An MiFE scheme MIFE_n for n-ary functions $\mathcal{F}$ consists of four algorithms $(\mathsf{MIFE}_n.\mathsf{Setup}, \mathsf{MIFE}_n.\mathsf{KeyGen}, \mathsf{MIFE}_n.\mathsf{Enc}, \mathsf{MIFE}_n.\mathsf{Dec})$ described below:

- **Setup.** $\mathsf{MIFE}_n.\mathsf{Setup}(1^\lambda)$ is a PPT algorithm that takes as input a security parameter λ and outputs the master secret key $\mathsf{MIFE}_n.\mathsf{msk}$.
- **Key Generation.** $\mathsf{MIFE}_n.\mathsf{KeyGen}(\mathsf{MIFE}_n.\mathsf{msk}, f)$ is a PPT algorithm that takes as input the master secret key $\mathsf{MIFE}_n.\mathsf{msk}$ and a function $f \in \mathcal{F}_\lambda$. It outputs a functional key $\mathsf{MIFE}_n.sk_f$.
- **Encryption.** $\mathsf{MIFE}_n.\mathsf{Enc}(\mathsf{MIFE}_n.\mathsf{msk}, m, i)$ is a PPT algorithm that takes as input the master secret key $\mathsf{MIFE}_n.\mathsf{msk}$, a message $x \in \mathcal{X}_\lambda$ and an index $i \in [n]$. It outputs a ciphertext $\mathsf{MIFE}_n.\mathsf{ct}$.

  Here index i signals to the encryption algorithm that message x corresponds to the i^{th} input of functions $f \in \mathcal{F}_\lambda$.
- **Decryption.** $\mathsf{MIFE}_n.\mathsf{Dec}(\mathsf{MIFE}_n.sk_f, \mathsf{MIFE}_n.\mathsf{ct})$ is a deterministic algorithm that takes as input a functional key $\mathsf{MIFE}_n.sk_f$ and a ciphertext $\mathsf{MIFE}_n.\mathsf{ct}$. It outputs a value $y \in \mathcal{Y}_\lambda$.

Remark 3.1. From now on, we use the phrase "encryption of m in the i^{th} position" to refer to the process of executing $\mathsf{MIFE}_n.\mathsf{Enc}$ on the input $(\mathsf{MIFE}_n.\mathsf{msk}, m, i)$.

Correctness. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, every n-ary function $f \in \mathcal{F}_\lambda$ and input tuple $(x_1, \ldots, x_n) \in \mathcal{X}_\lambda^n$

$$\Pr\left[\begin{array}{c}\mathsf{MIFE}_n.\mathsf{msk} \leftarrow \mathsf{MIFE}_n.\mathsf{Setup}\left(1^\lambda\right) \;;\; \mathsf{MIFE}_n.sk_f \leftarrow \mathsf{MIFE}_n.\mathsf{KeyGen}\left(\mathsf{MIFE}_n.\mathsf{msk}, f\right) \;; \\ \mathsf{MIFE}_n.\mathsf{Dec}\left(\mathsf{MIFE}_n.sk_f, \{\mathsf{MIFE}_n.\mathsf{Enc}\left(\mathsf{MIFE}_n.\mathsf{msk}, x_i, i\right)\}_{i=1}^n\right) \neq f\left(x_1, \ldots, x_n\right)\end{array}\right]$$

is at most $\mathsf{negl}(\lambda)$. In the above expression, the probability is taken over the random coins of all the algorithms.

We present the function privacy definition below. Similar to the single ary setting, we can consider two security notions – selective and adaptive. We first give the selective security definition since this is the definition we are going to consider throughout this paper.

Definition 3.2 (Selective Function Private MiFE). *A secret-key MiFE scheme MIFE_n for n-ary functions $\mathcal{F}$ is $(q_{\mathsf{key}}, q_{\mathsf{msg}}, \mu)$-selective function private if for any PPT adversary $\mathcal{A}$, there exists a function $\mu(\lambda)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ is*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{MIFE}_n} = \left|\Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{MIFE}_n}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{MIFE}_n}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ and $\lambda \in \mathbb{N}$, the experiment $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{MIFE}_n}(1^\lambda, b)$ is defined below:

1. ***Message Queries:*** *$\mathcal{A}$ submits q_{msg} number of queries, $\left\{\left((x^j_{1,0}, x^j_{1,1}), \ldots, (x^j_{n,0}, x^j_{n,1})\right)\right\}_{j\in[q_{\mathsf{msg}}]}$, with $x^j_{i,0} \in \mathcal{X}_\lambda$, to the challenger C.*
2. *C computes $\mathsf{MIFE}_n.\mathsf{msk} \leftarrow \mathsf{MIFE}_n.\mathsf{Setup}(1^\lambda)$. It then computes $\mathsf{MIFE}_n.\mathsf{ct}^j_i \leftarrow \mathsf{MIFE}_n.\mathsf{Enc}(\mathsf{MIFE}_n.\mathsf{msk}, x^j_{i,b})$ for all $i \in [n]$ for all $j \in [q_{\mathsf{msg}}]$. The challenger C then sends $\left\{(\mathsf{MIFE}_n.\mathsf{ct}^j_1, \ldots, \mathsf{MIFE}_n.\mathsf{ct}^j_n)\right\}_{j\in[q_{\mathsf{msg}}]}$ to the adversary $\mathcal{A}$.*
3. ***Function Queries:*** *The following is repeated up to q_{key} times: $\mathcal{A}$ submits a function query $(f_0, f_1) \in \mathcal{F}^2_\lambda$ to C. The challenger C computes $\mathsf{MIFE}_n.sk_f \leftarrow \mathsf{MIFE}_n.\mathsf{KeyGen}(\mathsf{MIFE}_n.\mathsf{msk}, f_b)$ and sends it to $\mathcal{A}$.*
4. *If there exists a function query (f_0, f_1) and a challenge message query $\left((x_{1,0}, \ldots, x_{n,0}), (x_{1,1}, \ldots, x_{n,1})\right)$ such that $f_0(x_{1,0}, \ldots, x_{n,0}) \neq f_1(x_{1,1}, \ldots, x_{n,1})$, then the output of the experiment is set to $\perp$. Otherwise, the output of the experiment is set to b', where b' is the output of $\mathcal{A}$.*

Remark 3.3. When μ is a negligible function in the security parameter, then we omit it from the notation and simply refer to $(q_{\mathsf{key}}, q_{\mathsf{msg}})$-function privacy of MiFE.

Constructing Function Private MiFE. In the single ary setting, Brakerski-Segev [16] gave a generic transformation that converts any secret key single ary FE into a function private secret key single ary FE. We observe that techniques, similar to those used in Brakerski-Segev, can be adapted to obtain a transformation from any i-ary MiFE into a function private i-ary MiFE in the secret key setting. We defer the technical details to the full version.

4 Our Transformation: From c-ary to (c + 1)-ary MiFE

In this section, we show how to transform a secret-key MiFE scheme for c-ary functions into an MiFE scheme for $(c+1)$-ary functions, for $c \geq 1$.

Our transformation proceeds in two steps:

1. Starting with an MiFE scheme for c-ary functions, we first apply the function privacy transformation (mentioned towards the end of Sect. 3) to obtain a function private MiFE scheme MIFE_c for c-ary functions.
2. Next, we convert MIFE_c into an MiFE scheme MIFE_{c+1} for $c+1$-ary functions. We refer to this step as the *arity amplification* step.

We now describe the arity amplification step. We construct an MiFE scheme for $c+1$-ary functions MIFE_{c+1} with function space $\mathcal{F}^{c+1}$ and message space $\mathcal{X}^{c+1}$.

Notation. We use the following tools in our transformation: (a) A function private MIFE scheme for c-ary functions, denoted as $\mathsf{MIFE}_c = (\mathsf{MIFE}_c.\mathsf{Setup}, \mathsf{MIFE}_c.\mathsf{KeyGen}, \mathsf{MIFE}_c.\mathsf{Enc}, \mathsf{MIFE}_c.\mathsf{Dec})$. Let $\mathcal{F}^{\mathsf{fp},c}$ and $\mathcal{X}^{\mathsf{fp},c}$ be the associated function space and message space, respectively. (b) A public-key FE scheme for single-ary functions, denoted as $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$.

Let $\mathcal{F}^{\mathsf{fe}}$ and $\mathcal{X}^{\mathsf{fe}}$ be the associated function space and message space, respectively. (c) A puncturable pseudorandom function family, denoted as $\mathsf{F} = \mathsf{PRF}_K(\cdot)$.

Setup $\mathsf{MIFE}_{c+1}.\mathsf{Setup}(1^\lambda)$*:* On input a security parameter λ, sample a master secret key $\mathsf{MIFE}_c.\mathsf{msk} \leftarrow \mathsf{MIFE}_c.\mathsf{Setup}(1^\lambda)$ of MIFE_c and a key pair $(\mathsf{FE.pk}, \mathsf{FE.msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ of FE. Output $\mathsf{MIFE}_{c+1}.\mathsf{msk} = (\mathsf{MIFE}_c.\mathsf{msk}, \mathsf{FE.pk}, \mathsf{FE.msk})$.

Key Generation $\mathsf{MIFE}_{c+1}.\mathsf{KeyGen}(\mathsf{MIFE}_{c+1}.\mathsf{msk}, f)$*:* On input master secret key $\mathsf{MIFE}_{c+1}.\mathsf{msk}$ and a function $f \in \mathcal{F}^{c+1}$, parse $\mathsf{MIFE}_{c+1}.\mathsf{msk} = (\mathsf{MIFE}_c.\mathsf{msk}, \mathsf{FE.pk}, \mathsf{FE.msk})$. Sample a functional key $\mathsf{FE}.sk_f \leftarrow \mathsf{FE.KeyGen}(\mathsf{FE.msk}, f)$ for function f. Output $\mathsf{MIFE}_{c+1}.sk_f = \mathsf{FE}.sk_f$.

Encryption $\mathsf{MIFE}_{c+1}.\mathsf{Enc}(\mathsf{MIFE}_{c+1}.\mathsf{msk}, x, i)$*:* On input master secret key $\mathsf{MIFE}_{c+1}.\mathsf{msk}$, message $x \in \mathcal{X}^{c+1}$ and index i, parse $\mathsf{MIFE}_{c+1}.\mathsf{msk} = (\mathsf{MIFE}_c.\mathsf{msk}, \mathsf{FE.pk}, \mathsf{FE.msk})$.

1. If $i = 1$, then draw a PRF key $K \in \{0,1\}^\lambda$ at random. Initialize the index vector $\mathcal{I} = (0, \ldots, 0)$. Compute $\mathsf{MIFE}_c.sk_G \leftarrow \mathsf{MIFE}_c.\mathsf{KeyGen}(\mathsf{MIFE}_c.\mathsf{msk}, G)$ where the circuit $G = \mathsf{GenCT}^{(c)}_{[x,1,K,\mathsf{FE.pk},\mathcal{I}]} \in \mathcal{F}^{\mathsf{fp},c}$ is described in Fig. 2. Output the ciphertext $\mathsf{MIFE}_{c+1}.\mathsf{ct} = \mathsf{MIFE}_c.sk_G$.
2. Else, if $2 \leq i \leq c+1$, then perform the following steps:
   - If the input message x is of the form $(x_1, x_2, 1, \tau, i-1)$ then compute $\mathsf{MIFE}_{c+1}.\mathsf{ct} \leftarrow \mathsf{MIFE}_c.\mathsf{Enc}(\mathsf{MIFE}_c.\mathsf{msk}, (x_1, x_2, 1, \tau, i), i)$
   - Else, choose a tag $\tau \in \{0,1\}^\lambda$ at random. Compute $\mathsf{MIFE}_{c+1}.\mathsf{ct} \leftarrow \mathsf{MIFE}_c.\mathsf{Enc}(\mathsf{MIFE}_c.\mathsf{msk}, (x, x, 1, \tau, i), i)$.

   Output the ciphertext $\mathsf{MIFE}_{c+1}.\mathsf{ct}$.

$\underline{\mathsf{GenCT}^{(c)}_{[x_1, j_1, K, \mathsf{FE.pk}, \mathcal{I}]}}$

Input: $\Big((x_2, x_2', j_2, \tau_2, i_2), \ldots, (x_{c+1}, x_{c+1}', j_{c+1}, \tau_{c+1}, i_{c+1})\Big)$.

- $\forall k \in \{2, \ldots, c+1\}$, if $i_k \neq k$, then output $\perp$.
- $\mathsf{FE.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{FE.pk}, (x_1, \ldots, x_{c+1}); \mathsf{PRF}(K, \tau_2 || \cdots || \tau_{c+1}))$.
- Output FE.ct.

Fig. 2. Description of GenCT^c.

Decryption $\mathsf{MIFE}_{c+1}.\mathsf{Dec}(\mathsf{MIFE}_{c+1}.sk_f, \mathsf{MIFE}_{c+1}.\mathsf{ct}_1, \ldots, \mathsf{MIFE}_{c+1}.\mathsf{ct}_{c+1})$*:* On input $(\mathsf{MIFE}_{c+1}.sk_f, \mathsf{MIFE}_{c+1}.\mathsf{ct}_1, \ldots, \mathsf{MIFE}_{c+1}.\mathsf{ct}_{c+1})$, perform the following steps:

1. Parse: (a) $\mathsf{MIFE}_{c+1}.sk_f = \mathsf{FE}.sk_f$, (b) $\mathsf{MIFE}_{c+1}.\mathsf{ct}_1 = \mathsf{MIFE}_c.sk_G$, and (c) $\mathsf{MIFE}_{c+1}.\mathsf{ct}_i = \mathsf{MIFE}_c.\mathsf{ct}_{i-1}$ for all $i \neq 1$, where $\mathsf{MIFE}_c.\mathsf{ct}_{i-1}$ denotes the ciphertext corresponding to $(i-1)^{th}$ position in MIFE_c.
2. Next, compute $\mathsf{FE}.\mathsf{ct}^* \leftarrow \mathsf{MIFE}_c.\mathsf{Dec}(\mathsf{MIFE}_c.sk_G, \mathsf{MIFE}_c.\mathsf{ct}_1, \ldots, \mathsf{MIFE}_c.\mathsf{ct}_c)$.
3. Finally, compute $y \leftarrow \mathsf{FE}.\mathsf{Dec}(\mathsf{FE}.sk_f, \mathsf{FE}.\mathsf{ct}^*)$. Output y.

This completes the description of the scheme.

Correctness. We now argue the correctness of MIFE_{c+1}. Let $\mathsf{MIFE}_c.sk$ be a valid functional key for the function $\mathsf{GenCT}[x_1, j_1, K, \mathsf{FE}.\mathsf{pk}, \mathcal{I}]$ w.r.t. MIFE_c. For $i \in [c]$, let $\mathsf{MIFE}_c.\mathsf{ct}_i$ be a valid encryption of x_{i+1} w.r.t. MIFE_c. By the correctness of $\mathsf{MIFE}_c.\mathsf{Enc}$, we have that the output of $\mathsf{MIFE}_c.\mathsf{Dec}(\mathsf{MIFE}_c.sk_{\mathsf{GenCT}}, \mathsf{MIFE}_c.\mathsf{ct}_1, \ldots, \mathsf{MIFE}_c.\mathsf{ct}_c)$ is $\mathsf{FE}.\mathsf{ct}^*$, where $\mathsf{FE}.\mathsf{ct}^*$ is a valid encryption of $(x_1, \ldots, x_{c+1})$ w.r.t. FE. Further, from the correctness of FE, it follows that the output of $\mathsf{FE}.\mathsf{Dec}(\mathsf{FE}.sk_f, \mathsf{FE}.\mathsf{ct}^*)$ is $f(x_1, \ldots, x_{c+1})$, where $\mathsf{FE}.sk_f$ is a valid functional key of f w.r.t. FE. The proof of security can be found in the full version [5].

5 Multi-input FE from Single-input FE

In Sect. 4, we gave a general transformation from a secret-key MiFE scheme for c-ary functions to another secret-key MiFE scheme for $c+1$-ary functions. Using this transformation, we now give a construction of a secret-key MiFE scheme for functions with $n = \mathsf{poly}(\lambda)$ arity. Later we will use this construction to obtain our main result on $i\mathrm{O}$. We will also consider different instantiations of this construction which yield new results on constant-ary MiFE from standard assumptions.

We construct a n-ary (q,q)-secure MiFE scheme MIFE_n. To obtain this construction we start with a q-secure[8] public-key FE scheme. This implies a single-ary (q,q)-secure secret-key MiFE scheme, MIFE_1.

(Iterated) Construction of MIFE_n (Informal description):
Repeat the following two steps for $c = 1, \ldots, n$:

1. **Function Privacy Transformation:** Using the MiFE function-privacy transformation (mentioned towards the end of Sect. 3), convert the (q,q)-secure MiFE scheme MIFE_c, obtained in the previous iteration, into a function-private (q,q)-secure MiFE scheme $\mathsf{MIFE}_c^{\mathsf{fp}}$, also supporting c-arity functions.
2. **Arity Amplification:** The function-private c-ary (q,q)-secure MiFE scheme $\mathsf{MIFE}_c^{\mathsf{fp}}$ obtained in the previous step is then transformed into a $c+1$-ary (q,q)-secure MiFE scheme, using the transformation presented in Sect. 4. In this step, we additionally use a q-secure public-key FE scheme FE.

[8] Throughout this section, we only consider selectively secure public-key FE and secret key MiFE schemes. For simplicity of notation, we omit the use of the word "selective" in the rest of this section and assume that it is implicit.

The efficiency properties of the underlying public-key FE scheme determines the value of n that we can achieve in the above construction. Consequently, we consider two different instantiations of the underlying public-key FE scheme that yield different results. We discuss these instantiations next.

iO from Compact FE. We start by stating our main result for secret-key MiFE for polynomial-arity functions.

Theorem 5.1. *For all $n = \mathrm{poly}(\lambda)$, the above proposed scheme MIFE_n is (q, q)-secure for any polynomial q, assuming that FE is $\left(1, \frac{1}{(64q)^{(n+1)^2} \cdot 2^\lambda}\right)$-selectively secure compact public-key FE scheme.*

The core non triviality in the above theorem is to argue that the size of parameters does not grow exponentially with the number of iterations. At a high level, this is because the compactness of FE ensures that the growth of the parameters at the i^{th} level (i-ary MiFE) depends only on the security parameter, level i and the message length. The actual calculations can be found in the appropriate section in the full version.

We now invoke a theorem by [35] that shows how to obtain iO for functions of input length n from a n-ary MiFE for a specific function family. We thus have the following main theorem.

Theorem 5.2. *Assuming the $(2, \frac{1}{(128)^{n^2} 2^\lambda})$-security of compact public-key selectively secure FE public key FE scheme for polynomial time computable functions, the scheme $i\mathrm{O}$ is an indistinguishability obfuscation scheme for P/poly.*

Constant ary MiFE. If we restrict our attention to just constant ary MiFE then we can obtain a construction based on public key encryption encryption schemes. We state the result formally below.

Theorem 5.3. *For any constant n, the above proposed scheme MIFE_n is (q, q)-selectively secure assuming that FE is a q-selectively secure (not necessarily compact) public-key FE scheme.*

Combining Theorem 5.3 with [37,44], we obtain the following result.

Corollary 5.4. *For any polynomial $q = q(\lambda)$, there exists a (q, q)-selectively secure secret-key MiFE scheme for constant-arity functions, assuming the existence of semantically-secure public-key encryption.*

The reason why we can only achieve constant arity is because the growth of parameters in this case could be exponential. If we start from any public key FE scheme, the size of the parameters at each level grows proportional to the size of the parameters at the next level. This stems from the fact that the FE scheme that we start off with could be such that the encryption complexity might depend on the function complexity. And hence, the number of iterations that can be performed is just a constant. A detailed explanation is provided in the full version.

References

1. Agrawal, S., Agrawal, S., Badrinarayanan, S., Kumarasubramanian, A., Prabhakaran, M., Sahai, A.: Function private functional encryption and property preserving encryption: new definitions and positive results. IACR Cryptology ePrint Archive 2013/744 (2013)
2. Agrawal, S., Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption: new perspectives and lower bounds. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II [17]. LNCS, vol. 8043, pp. 500–518. Springer, Heidelberg (2013). http://dx.doi.org/10.1007/978-3-642-40084-1_28
3. Ananth, P., Boneh, D., Garg, S., Sahai, A., Zhandry, M.: Differing-inputs obfuscation and applications. IACR Cryptology ePrint Archive 2013/689 (2013). http://eprint.iacr.org/2013/689
4. Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: The trojan method in functional encryption: from selective to adaptive security, generically. IACR Cryptology ePrint Archive 2014/917 (2014). http://eprint.iacr.org/2014/917
5. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. Technical report, Cryptology ePrint Archive, report 2015/173 (2015)
6. Applebaum, B., Ishai, Y., Kushilevitz, E.: Computationally private randomizing polynomials and their applications. Comput. Complexity **15**(2), 115–162 (2006)
7. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 221–238. Springer, Heidelberg (2014)
8. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (Im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). http://dx.doi.org/10.1007/3-540-44647-8_1
9. Bitansky, N., Garg, S., Lin, H., Pass, R., Telang, S.: Succinct randomized encodings and their applications. In: STOC (2015)
10. Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 563–594. Springer, Heidelberg (2015)
11. Boneh, D., Raghunathan, A., Segev, G.: Function-private identity-based encryption: hiding the function in functional encryption. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 461–478. Springer, Heidelberg (2013)
12. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (2013)
13. Boneh, D., Wu, D.J., Zimmerman, J.: Immunizing multilinear maps against zeroizing attacks. IACR Cryptology ePrint Archive 2014/930 (2014). http://eprint.iacr.org/2014/930
14. Boyle, E., Chung, K.-M., Pass, R.: On extractability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 52–73. Springer, Heidelberg (2014). http://dx.doi.org/10.1007/978-3-642-54242-8_3
15. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014)

16. Brakerski, Z., Segev, G.: Function-private functional encryption in the private-key setting. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 306–324. Springer, Heidelberg (2015)
17. Canetti, R., Garay, J.A. (eds.): CRYPTO 2013, Part II. LNCS, vol. 8043. Springer, Heidelberg (2013). http://dx.doi.org/10.1007/978-3-642-40084-1
18. Canetti, R., Holmgren, J., Jain, A., Vaikuntanathan, V.: Indistinguishability obfuscation of iterated circuits and RAM programs. In: STOC (2015)
19. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (2015)
20. De Caro, A., Iovino, V., Jain, A., O'Neill, A., Paneth, O., Persiano, G.: On the achievability of simulation-based security for functional encryption. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II [17]. LNCS, vol. 8043, pp. 519–535. Springer, Heidelberg (2013). http://dx.doi.org/10.1007/978-3-642-40084-1_29
21. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 3–12. Springer, Heidelberg (2015)
22. Coron, J.-S., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 476–493. Springer, Heidelberg (2013)
23. Coron, J., Lepoint, T., Tibouchi, M.: Cryptanalysis of two candidate fixes of multilinear maps over the integers. IACR Cryptology ePrint Archive 2014/975 (2014). http://eprint.iacr.org/2014/975
24. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013). http://dx.doi.org/10.1007/978-3-642-38348-9_1
25. Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 74–94. Springer, Heidelberg (2014)
26. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26–29 October 2013, Berkeley, CA, USA, pp. 40–49. IEEE Computer Society (2013). http://doi.ieeecomputersociety.org/10.1109/FOCS.2013.13
27. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Fully secure functional encryption without obfuscation. IACR Cryptology ePrint Archive 2014/666 (2014). http://eprint.iacr.org/2014/666
28. Garg, S., Lu, S., Ostrovsky, R., Scafuro, A.: Garbled RAM from one-way functions. In: STOC (2015)
29. Gentry, C., Halevi, S., Lu, S., Ostrovsky, R., Raykova, M., Wichs, D.: Garbled RAM revisited. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 405–422. Springer, Heidelberg (2014)
30. Gentry, C., Halevi, S., Maji, H.K., Sahai, A.: Zeroizing without zeroes: cryptanalyzing multilinear maps without encodings of zero. IACR Cryptology ePrint Archive 2014/929 (2014). http://eprint.iacr.org/2014/929
31. Gentry, C., Halevi, S., Raykova, M., Wichs, D.: Outsourcing private RAM computation. IACR Cryptology ePrint Archive 2014/148 (2014). http://eprint.iacr.org/2014/148
32. Gentry, C., Lewko, A., Waters, B.: Witness encryption from instance independent assumptions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 426–443. Springer, Heidelberg (2014)

33. Gentry, C., Lewko, A.B., Sahai, A., Waters, B.: Indistinguishability obfuscation from the multilinear subgroup elimination assumption. IACR Cryptology ePrint Archive 2014/309 (2014). http://eprint.iacr.org/2014/309
34. Goldreich, O.: Foundations of Cryptography: Volume 2, Basic Applications, vol. 2. Cambridge University Press, Cambridge (2009)
35. Goldwasser, S., Gordon, S.D., Goyal, V., Jain, A., Katz, J., Liu, F.-H., Sahai, A., Shi, E., Zhou, H.-S.: Multi-input functional encryption. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 578–602. Springer, Heidelberg (2014). http://dx.doi.org/10.1007/978-3-642-55220-5_32
36. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) Symposium on Theory of Computing Conference, STOC 2013, 1–4 June 2013, Palo Alto, CA, USA, pp. 555–564. ACM (2013). http://doi.acm.org/10.1145/2488608.2488678
37. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions via multi-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 162–179. Springer, Heidelberg (2012)
38. Ishai, Y., Kushilevitz, E.: Randomizing polynomials: a new representation with applications to round-efficient secure computation. In: 41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12–14 November 2000, Redondo Beach, California, USA, pp. 294–304 (2000)
39. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, pp. 669–684. ACM (2013)
40. Komargodski, I., Segev, G., Yogev, E.: Functional encryption for randomized functionalities in the private-key setting from minimal assumptions. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 352–377. Springer, Heidelberg (2015)
41. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. In: STOC (2015)
42. Lu, S., Ostrovsky, R.: How to garble RAM programs? In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 719–734. Springer, Heidelberg (2013)
43. Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 500–517. Springer, Heidelberg (2014)
44. Sahai, A., Seyalioglu, H.: Worry-free encryption: functional encryption with public keys. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, 4–8 October 2010, Chicago, Illinois, USA, pp. 463–472 (2010)
45. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) Symposium on Theory of Computing, STOC 2014, 31 May–03 June 2014, New York, NY, USA, pp. 475–484. ACM (2014). http://doi.acm.org/10.1145/2591796.2591825
46. Waters, B.: A punctured programming approach to adaptively secure functional encryption. Cryptology ePrint Archive, report 2014/588 (2014)
47. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: FOCS, pp. 162–167 (1986)