Indistinguishability Obfuscation: from Approximate to Exact*

Nir Bitansky[†] Vinod Vaikuntanathan[‡]

Abstract

We show general transformations from subexponentially-secure approximate indistinguishability obfuscation (IO) where the obfuscated circuit agrees with the original circuit on a $1/2+\epsilon$ fraction of inputs on a certain samplable distribution, into exact indistinguishability obfuscation where the obfuscated circuit and the original circuit agree on all inputs. As a step towards our results, which is of independent interest, we also obtain an approximate-to-exact transformation for functional encryption. At the core of our techniques is a method for "fooling" the obfuscator into giving us the correct answer, while preserving the indistinguishability-based security. This is achieved based on various types of secure computation protocols that can be obtained from different standard assumptions.

Put together with the recent results of Canetti, Kalai and Paneth (TCC 2015), Pass and Shelat (Eprint 2015), and Mahmoody, Mohammed and Nemathaji (Eprint 2015), we show how to convert indistinguishability obfuscation schemes in various ideal models into exact obfuscation schemes in the plain model.

Keywords: Functional Encryption, Program Obfuscation, Secure Function Evaluation.

---

*This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

[†]MIT. `E-mail: nirbitan@csail.mit.edu`. Research supported in part by NSF Grants CNS-1350619 and CNS-1414119.

[‡]MIT. `E-mail: vinodv@csail.mit.edu`. Research supported in part by NSF Grants CNS-1350619 and CNS-1414119, Alfred P. Sloan Research Fellowship, Microsoft Faculty Fellowship, the NEC Corporation, and a Steven and Renee Finn Career Development Chair from MIT.

Contents

1 Introduction

Program obfuscation, the science of making programs "unintelligible" while preserving functionality, has been a holy grail in cryptography for over a decade. While the most natural and intuitively appealing notion of obfuscation, namely *virtual-black-box* (VBB) obfuscation [BGI+12], was shown to have strong limitations [BGI+12, GK05, BCC+14], the recent work of Garg, Gentry, Halevi, Raykova, Sahai and Waters [GGH+13b, SW14] opened new doors by demonstrating that the weaker notion of *indistinguishability obfuscation* (IO) is both very useful and potentially achievable. Since then, a veritable flood of applications has made indistinguishability obfuscation virtually "crypto-complete".

On the flip side, the tremendous power of IO also begets its reliance on strong and untested computational assumptions. Indeed, it has been a major cryptographic quest to come up with a construction of IO with a security proof based on well-studied computational assumptions. Garg et al. [GGH+13b] gave us the first *candidate* construction of IO, however the construction came as-is, without a security proof.

We have recently seen several works [PST14, GLSW14, AJ15, BV15] that shed light on how a security proof for IO will look like. Pass, Seth and Telang show security of an IO construction based on a "semantic security" assumption on multi-linear maps [GGH12]; Gentry, Lewko, Sahai and Waters [GLSW14] (following [GLW14]) show security based on the "multilinear subgroup elimination assumption" on multi-linear maps; Ananth and Jain [AJ15] and Bitansky and Vaikuntanathan [BV15] show how to construct IO from any functional encryption scheme. Unfortunately, the first two of these works are based on the mathematical abstraction of multi-linear maps which has had a troubled history so far (with several constructions [GGH13a, CLT13, BWZ14, GGHZ14, GGH15, CLT15] and matching attacks [GGH13a, LS14, CHL+15, CGH+15, HJ15]), and the last two rely on functional encryption with succinct encryption for which the only known constructions, yet again, use multi-linear maps.

Yet another line of work focuses on proving the security of obfuscators in so-called *idealized models*. In a typical idealized model, both the construction and the adversary have access to an oracle that implements a certain functionality; in the random oracle model [BR93], this is a random function; in the generic group model [Sho97], this is the functionality of a group; and the most recent entrant to this club, namely the ideal multilinear map model, is an abstraction of the functionality of multilinear maps. Several works [BR14b, BR14a, BGK+14, AB15, Zim15] along this route prove security of (different) constructions of obfuscation (even in the sense of virtual black-box security) in various ideal multi-linear map models.

An even more recent line of work, initiated by Canetti, Kalai, and Paneth [CKP15], investigates how to transform constructions of obfuscation in idealized models into ones in the plain model, where there are no oracles. Indeed, this may lead to an aesthetically appealing avenue to constructing obfuscation schemes:

1. *Construct an obfuscation scheme in an appropriate idealized model.*
2. *"De-idealize" it: translate the ideal model obfuscation scheme into a scheme in the real world.*

Even if eventual constructions of obfuscation schemes do not initially proceed along these lines, we believe that this two-step process is a conceptually appealing route to eventual, mature, constructions of obfuscation schemes. Indeed, constructions in ideal models, while not immediately deployable, typically give us an abstract, high level, understanding.

Concretely, the work of [CKP15] essentially shows that any obfuscator in the random oracle model can be converted to an obfuscator in the plain model with the same security properties. Pass and Shelat [PS15] and subsequently, Mahmoody, Mohammed and Nematihaji [MMN15] extend this to the generic group and ring models respectively, as well as ideal multilinear maps model with *bounded multi-linearity*.

However, the resulting obfuscators suffer from a major drawback: *they only have approximate correctness*. That is, the plain model obfuscator may err on a polynomially large fraction of inputs (or more generally with some polynomial probability when inputs are taken from a given samplable distribution). Roughly speaking, these results proceed by isolating a list of "heavy oracle queries", that is, queries that arise in the evaluation of the obfuscated circuit on a large fraction of inputs. Once the (polynomially large set of) heavy queries are identified, the result of the oracle queries on this set is published as part of the obfuscated circuit. This approach will inherently miss the queries made by a rare set of inputs, resulting in an incorrect evaluation.

While these transformations already have interesting consequences (regarding the impossibility of VBB in these idealised models), the lack of correctness presents a serious obstacle towards fulfilling the above two-step plan. Indeed, it is far from clear that applications of IO will work when we only have *approximate IO* at our disposal. Certainly, one could go through the applications of IO one-by-one, and attempt to re-derive them from approximate IO, but in the absence of automated theorem provers[1], this seems neither particularly efficient nor aesthetically pleasing.

This motivates us to ask:

Is there an approximate-to-exact transformation for indistinguishability obfuscation?

In other words, we are asking for "one transformation to rule them all", a generic way to compile an approximate obfuscation scheme into a perfectly correct obfuscation scheme, automatically enabling to recover all the applications of IO even given only approximately correct obfuscation.

In this work, we provide exactly such a transformation, under standard additional assumptions. Let us now describe our results in detail.

1.1 Our Results

We say that an obfuscator $\mathsf{ap}\mathcal{O}$ is $(\mathcal{X},\alpha)$-correct for a given input sampler $\mathcal{X}$ and $\alpha \in [0,1]$ (which may depend on the security parameter), if it is correct with probability at least α over inputs sampled by $\mathcal{X}$. Security is defined as in the standard setting of (exact) indistinguishability obfuscation. We shall refer to such an obfuscator as an *approximate indistinguishability obfuscator*.

Our main result is that approximate IO with subexponential security for a certain class of samplers can be converted under standard assumptions into *almost exact IO* where for any circuit, with overwhelming probability over the coins of the obfuscator algorithm the resulting obfuscation is correct on *all* inputs. We present two routes towards this result based on different assumptions and with different parameters.

Theorem 1.1 (informal). *Assuming DDH, there exists an input sampler $\mathcal{X}_1$ and a transformation that for any $\alpha \geq \frac{1}{2} + \lambda^{-O(1)}$, converts any $(\mathcal{X}_1,\alpha)$-correct sub-exponentially secure IO scheme for* $\mathbf{P/poly}$ *into an almost exact IO scheme for* $\mathbf{P/poly}$.

[1] Graduate students do not count.

Theorem 1.2 (informal)**.** *Assuming sub-exponentially-secure puncturable PRFs in* $\mathbf{NC}^1$*, there exists an input sampler* $\mathcal{X}_2$*, polynomial* $\text{poly}_2(\cdot)$*, and a transformation that for any* $\alpha \geq 1 - \frac{1}{\text{poly}_2(\lambda)}$*, converts any* $(\mathcal{X}_2, \alpha)$*-correct sub-exponentially-secure IO scheme for* $\mathbf{P}/\mathbf{poly}$ *into an almost exact IO scheme for* $\mathbf{P}/\mathbf{poly}$*.*

Since the works of [CKP15, PS15, MMN15] apply to any efficient sampler $\mathcal{X}$ and any α that is polynomially bounded away from 1, we obtain the following main corollary

Corollary (Main Theorems + [CKP15, PS15, MMN15])**.** *Assume that there is an indistinguishability obfuscator in either the random oracle model, the ideal generic group/ring model, or ideal multilinear maps model with bounded multi-linearity. Then, there is an (almost) exact obfuscator in the plain model.*

We note that our theorems result in IO that may still output an erroneous obfuscator, but only with some negligible probability over the coins of the obfuscator alone. This is analogous to the setting of correcting decryption errors in plain public key encryption [DNR04], and as far as we know is sufficient in all applications. Nevertheless, we show that under a worst-case complexity assumption typically used in the setting of derandomization, we could transform any such obfuscator to one that is *perfectly correct.*

Theorem 1.3 (informal)**.** *Assuming there is a function in* $\mathbf{Dtime}(2^{O(n)})$ *with nondeterministic circuit complexity* $2^{\Omega(n)}$*, almost exact IO can be transformed into exact IO.*

We also show how to transform approximate functional encryption into exact functional encryption, where approximate FE is defined analogously to approximate IO with respect to a distribution on the message space and decryption errors. Besides being of independent interest, this transformation will also serve as a building block to obtain the second theorem above.

Theorem 1.4 (Informal)**.** *Assuming weak PRFs in* $\mathbf{NC}^1$*, there exists a message sampler* $\mathcal{X}$*, constant* η*, and a transformation that for any* $\alpha \geq 1 - \eta$*, converts any* $(\mathcal{X}, \alpha)$*-correct FE scheme for* $\mathbf{P}/\mathbf{poly}$ *into an almost exact scheme FE scheme for* $\mathbf{P}/\mathbf{poly}$*.*

We now proceed to provide an overview of our techniques.

1.2 Overview of Our Techniques

The starting point of our constructions comes from the notion of random self-reducibility [AFK89]. That is, imagine that you have an error-prone algorithm A that computes a (Boolean) function F correctly on a $1/2 + \varepsilon$ fraction of inputs. Suppose that there is an efficient randomizer $r(\cdot)$ that maps an input x into a random input $r = r(x)$ such that given $F(r)$, one can efficiently recover $F(x)$. Then, we can turn A into a **BPP** algorithm for computing F; namely, $A'(x) = A(r(x))$. The new algorithm computes F correctly for *any* input with high probability over its own random coins. The probability of error can then be made arbitrarily small using standard amplification (i.e., taking majority of $\approx \varepsilon^{-2}$ invocations).

In our setting, F is an arbitrary function, which is likely *not* random self-reducible. Nevertheless, we show how to make the *essence* of this idea work, using various notions of (two-party and multi-party) non-interactive secure function evaluation (SFE) [Yao86, BGW88, Gen09]. Indeed, certain forms of non-interactive SFE (or homomorphic encryption) have been used in several instances in the literature to obtain (sometimes computational) random self-reducibility [AIK06, CKV10, BP12, BGJ+15]. The rough idea is that if we can get the obfuscator to homomorphically evaluate a given function on encryptions for some fixed distribution on inputs, then it must also behave correctly with roughly the same probability on encryptions of any arbitrary input. This, however, should be done with care to ensure that homomorphic evaluation does not harm the security of the obfuscator. We next go into more details on how we carry out this agenda.

Our First Construction. Our first construction uses a two-party non-interactive secure function evaluation protocol with security against malicious senders. For simplicity, let us describe this approach in the language of fully homomorphic encryption (FHE). Let $(\mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a (secret-key) FHE scheme (not necessarily compact). (We assume that the randomness of the key generation algorithm acts as the secret key, and avoid explicitly dealing with the key generation algorithm.)

To exactly obfuscate a circuit C, we use the approximate obfuscator $\mathsf{ap}\mathcal{O}$ to obfuscate the circuit Eval_C which, given as input an encryption of some x, homomorphically computes an encryption of $C(x)$. Assume that $\mathsf{ap}\mathcal{O}(\mathsf{Eval}_C)$ is correct on a $1/2 + \varepsilon$ fraction of encryptions of 0^n. The key observation is that semantic security of the encryption scheme means that $\mathsf{ap}\mathcal{O}(\mathsf{Eval}_C)$ is also correct on a $1/2+\varepsilon-\lambda^{-\omega(1)}$ fraction of encryptions of any x; that is, it outputs $\mathsf{Eval}_C(\mathsf{Enc}(x)) = \mathsf{Enc}(C(x))$. This gives the required randomizer and can be amplified to give us correctness *for every input* x.

The problem with this idea is the security of the final obfuscator. Indeed, $\mathsf{Eval}_C(\mathsf{Enc}(x))$ may reveal information about the circuit C beyond the output $C(x)$. The problem goes even further: since the evaluator in this setting is untrusted, she can try to run the obfuscated circuits with malformed encryptions, potentially making the problem much worse. The solution is to rely on a *maliciously function-hiding* homomorphic encryption scheme. Such an object can be constructed using Yao's garbled circuits combined with an oblivious transfer (OT) protocol secure against malicious receivers (such as the Naor-Pinkas protocol based on the DDH assumption [NP01]). The evaluation procedure, however, is randomized, but can be derandomized with a pseudo-random function.

While the above works perfectly assuming ideal VBB obfuscation, this is not necessarily the case for IO. Nevertheless, we observe that we can use $\mathsf{ap}\mathcal{O}$ to obfuscate this (de)randomized circuit using the machinery of probabilistic IO [CLTV15]. This allows us to show that indistinguishability obfuscation is maintained, but requires going through an exponential number of hybrids requiring sub-exponential security from $\mathsf{ap}\mathcal{O}$ (and some of the other involved primitives).

Our Second Construction. Our second construction goes through the notion of functional encryption (FE). In a (public-key) FE scheme, the owner of a functional secret key FSK_F can "decrypt" a ciphertext $\mathsf{FE.Enc}(\mathsf{MPK}, m)$ to learn $F(m)$, but should learn nothing else about m. In an approximately correct FE scheme, the decryption algorithm could err on encryptions of certain messages m, but should be correct with probability $1 - \varepsilon$ on inputs x drawn from a (sampleable) distribution $\mathcal{X}$.

We show how to transform an approximately correct FE scheme into an exact FE scheme. Here the main advantage over the setting of approximate IO is that we are only concerned with honestly generated encrypted messages and are not concerned with function hiding. In particular, we can relax the assumptions required for the SFE and rely on (a non-interactive) information-theoretic version of the Ben-Or-Goldwasser-Wigderson [BGW88] multi-party computation protocol for $\mathbf{NC}^1$.

This construction also provides an alternative route for the IO transformation. Concretely, we show that starting from approximate IO, we can first apply the transformation of Garg et al. [GGH+13b] to obtain approximate FE. For this to work, we need show how to obtain (exact) NIZKs and public-key encryption directly from approximate IO, which are required for the transformation. Then, in the second step, we apply we apply our exact-to-approximate transformation for FE, and finally invoke a transformation from (exact) FE to IO [AJ15, BV15]. The latter transformation requires that the size of the encryption circuit the FE scheme is relatively succinct. In our case, due to the BGW-based SFE, this size grows exponentially in the depth. Fortunately though, in [BV15], it is shown that this still suffices to obtain IO, assuming also puncturable PRFs in NC^1.

Overall, this leads to a construction of (exact) IO from subexponentially-secure approximate IO and subexponentially-secure puncturable PRFs in $\mathbf{NC}^1$.

From Almost Exact to Exact. We transform any almost exact IO to a perfect one, in two steps. In the first step, we translate almost exact IO to "two-message IO", where the first message consists of a random string R, and the second message is the obfuscated circuit. In this two-message scheme, with overwhelming probability over the string R, the obfuscation is perfectly correct; namely it preserves exact functionality with probability one over the coins of the obfuscator. This is done using the reverse randomization idea from the constructions of Naor's commitments [Nao91], Naor and Dwork's ZAPs [DN07] and decryption-errors elimination by Dwork, Naor, and Reingold [DNR04].

In the second step, we derandomize the choice of the message R, following a similar approach to that of Barak, Ong, and Vadhan [BOV07] previously used to derandomize cryptographic constructions. This is done based on Nisan-Wigderson type psuedo-random generators [NW94], which are in turn known based on appropriate worst-case hardness assumptions [SU01].

2 Preliminaries

The cryptographic definitions in the paper follow the convention of modeling security against non-uniform adversaries. An efficient adversary $\mathcal{A}$ is modeled as a sequence of circuits $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, such that each circuit $\mathcal{A}_\lambda$ is of polynomial size $\lambda^{O(1)}$ with $\lambda^{O(1)}$ input and output bits. We often omit the subscript λ when it is clear from the context.

When we refer to a randomized algorithm $\mathcal{A}$, we typically do not explicitly denote its random coins, and use the notation $s \leftarrow \mathcal{A}$ or $s \leftarrow \mathcal{A}(x)$ if $\mathcal{A}$ has an extra input x. When we want to be explicit regarding the coins, we shall denote $s \leftarrow \mathcal{A}(r)$, or $s \leftarrow \mathcal{A}(x; r)$, respectively.

Whenever we refer to a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}$, we mean that each set $\mathcal{C}_\lambda$ consists of Boolean circuits of size at most $\mathrm{poly}(\lambda)$ for some polynomial $\mathrm{poly}(\cdot)$, defined on the domain $\{0,1\}^{n(\lambda)}$. When referring to inputs $x \in \{0,1\}^{n(\lambda)}$, we often omit λ from the notation.

2.1 Non-Interactive Secure Function Evaluation

We consider two-message secure function evaluation (SFE) protocols. Typically, such a protocol consists of two parties (A, B) and has the following syntax. Party A is given input x, encrypts x and sends the encrypted input to B. B given as additional input a function f, homomorphically evaluates f on the encrypted x, and returns the result to A, who can then decrypt the result $f(x)$. The protocol is required to ensure input-privacy for A and function privacy for B (on top of correctness).

Definition 2.1 (Secure Function Evaluation)**.** *A scheme* $\mathsf{SFE} = (\mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$*, where* $\mathsf{Enc}, \mathsf{Eval}$ *are probabilistic and* Dec *is deterministic, is a two-message secure function evaluation protocol for circuit class* $\mathcal{C} = \{\mathcal{C}_\lambda\}$*, where* $\mathcal{C}_\lambda$ *is defined over* $\{0,1\}^{n(\lambda)}$*, if the following requirements hold:*

- **Correctness:** *for any* $\lambda \in \mathbb{N}$, $C \in \mathcal{C}_\lambda$ *and input* $x \in \{0,1\}^n$ *in the domain of* C *it holds that:*

$$\Pr\left[\mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}}) = C(x) \;\middle|\; \begin{array}{c}(\mathsf{CT}, \mathsf{R}) \leftarrow \mathsf{Enc}(x) \\ \widehat{\mathsf{CT}} \leftarrow \mathsf{Eval}(\mathsf{CT}, C)\end{array}\right] \geq 1 - \mu(\lambda) \;\;,$$

for some negligible $\mu(\cdot)$, *where the probability is over the coin tosses of* Enc *and* Eval.

- **Input Hiding:** *for any polysize distinguisher* $\mathcal{D}$ *there exists a negligible function* $\mu(\cdot)$, *such that for all* $\lambda \in \mathbb{N}$, *and equal size inputs* $x_0, x_1 \in \{0,1\}^n$*:*

$$|\Pr[\mathcal{D}(\mathsf{CT}_0) = 1] - \Pr[\mathcal{D}(\mathsf{CT}_1) = 1]| \leq \mu(\lambda) \;\;,$$

where $\mathsf{CT}_b \leftarrow \mathsf{Enc}(x_b)$.

- **Malicious Function Hiding:** *there exists a (possibly inefficient) function* Ext, *such that for any polysize distinguisher* $\mathcal{D}$ *there exists a negligible function* $\mu(\cdot)$, *such that for all* $\lambda \in \mathbb{N}$, *maliciously chosen* CT^*, *and equal size circuits* $C_0, C_1 \in \mathcal{C}_\lambda$ *that agree on* $x = \mathsf{Ext}(\mathsf{CT})$*:*

$$\left|\Pr[\mathcal{D}(\widehat{\mathsf{CT}}_0) = 1] - \Pr[\mathcal{D}(\widehat{\mathsf{CT}}_1) = 1]\right| \leq \mu(\lambda) \;\;,$$

where $\widehat{\mathsf{CT}}_b \leftarrow \mathsf{Eval}(\mathsf{CT}^*, C_b)$.

We say that the scheme is δ*-function-hiding, for some concrete negligible function* $\delta(\cdot)$, *if for all poly-size distinguishers, the above indistinguishability gap* $\mu(\lambda)$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$.

Remark 2.2 (strong function privacy). For our most basic transformation from approximate IO to exact IO, we will require $2^{-\sigma(\lambda)} \cdot \lambda^{-\omega(1)}$-function-hiding, where $\sigma(\lambda)$ is the size of encryptions in the scheme. Below, we discuss an instantiation, based on the DDH assumption, that has perfect function-hiding, and thus satisfies this requirement.

Distributed Secure Function Evaluation. We will also consider a notion of two-message distributed function evaluation (DSFE). Such a protocol consists of $k+2$ parties $(A, B_1, \ldots, B_k, C)$ and has the following syntax. Party A, given input x, shares x into k shares and sends the shares to $B_1, \ldots, B_k$. The parties $B_1, \ldots, B_k$ given as additional input a function f, homomorphically and non-interactively evaluate f on each share, and send the evaluated shares to C, who can then decrypt and obtain the result $f(x)$.

The protocol is required to ensure that each individual share sent by A in the second message hides all information regarding the input x. We also require that C gains no information on the input, except for the output of the function (formally, we will require an indistinguishability-based guarantee analogous to that of functional encryption.) Furthermore, we will require that correctness holds even if some τ fraction of the parties $B_1, \ldots, B_k$ are faulty.

Definition 2.3 (Distributed Secure Function Evaluation)**.** *A scheme* $\mathsf{DSFE} = (\mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$, *where* Enc *is probabilistic and* $\mathsf{Eval}, \mathsf{Dec}$ *are deterministic, is a* (k, τ)*-secure distributed function evaluation protocol for circuit class* $\mathcal{C} = \{\mathcal{C}_\lambda\}$, *where* $\mathcal{C}_\lambda$ *is defined over* $\{0,1\}^n$ *for* $n = n(\lambda)$, $k = k(\lambda)$, *and* $\tau = \tau(\lambda)$, *if the following requirements hold:*

- **Correctness in the presence of faults:** *for any* $\lambda \in \mathbb{N}$, $C \in \mathcal{C}_\lambda$ *and input* $x \in \{0,1\}^n$ *in the domain of* C *and any set* $S \in [k]$ *of size smaller than* τn, *and functions* $\{\mathsf{Err}_i : i \in S\}$ *it holds that:*

$$\Pr\left[\mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}}_1, \ldots, \widehat{\mathsf{CT}}_k) = C(x) \;\middle|\; \begin{array}{c} (\mathsf{CT}_1, \ldots, \mathsf{CT}_k, \mathsf{R}) \leftarrow \mathsf{Enc}(x) \\ \forall i \in [k] \setminus S : \widehat{\mathsf{CT}}_i = \mathsf{Eval}(\mathsf{CT}_i, C) \\ \forall i \in S : \widehat{\mathsf{CT}}_i \leftarrow \mathsf{Err}_i(\mathsf{CT}_i) \end{array}\right] \geq 1 - \mu(\lambda) \ ,$$

for some negligible $\mu(\cdot)$, *where the probability is over the coin-tosses of* Enc.

- **Input Hiding:** *for any polysize distinguisher* $\mathcal{D}$ *there exists a negligible function* $\mu(\cdot)$, *such that for all* $\lambda \in \mathbb{N}$, *and equal size inputs* $x_0, x_1 \in \{0,1\}^n$ *and any* $i \in [k]$:

$$|\Pr[\mathcal{D}(\mathsf{CT}_{0,i}) = 1] - \Pr[\mathcal{D}(\mathsf{CT}_{1,i}) = 1]| \leq \mu(\lambda) \ ,$$

where $\mathsf{CT}_{b,i}$ *denotes the* i*-th ciphertext output by* $\mathsf{Enc}(x_b)$.

- **Residual Input Hiding:** *for any polysize distinguisher* $\mathcal{D}$ *there exists a negligible function* $\mu(\cdot)$, *such that for all* $\lambda \in \mathbb{N}$, *inputs* $x_0, x_1 \in \{0,1\}^n$, *and circuit* $C \in \mathcal{C}_\lambda$ *such that* $C(x_0) = C(x_1)$:

$$\left|\Pr[\mathcal{D}(\mathsf{R}_0, \widehat{\mathsf{CT}}_{0,1}, \ldots, \widehat{\mathsf{CT}}_{0,k}) = 1] - \Pr[\mathcal{D}(\mathsf{R}_1, \widehat{\mathsf{CT}}_{1,1}, \ldots, \widehat{\mathsf{CT}}_{1,k}) = 1]\right| \leq \mu(\lambda) \ ,$$

where for $(b,i) \in \{0,1\} \times [k]$, $\widehat{\mathsf{CT}}_{b,i} = \mathsf{Eval}(\mathsf{CT}_{b,i}, C)$, *and* $(\mathsf{CT}_{b,1}, \ldots, \mathsf{CT}_{b,k}, \mathsf{R}_b) \leftarrow \mathsf{Enc}(x_b)$.

Remark 2.4 (difference from SFE). There are two main differences from SFE. The first is in security, in the above we do not require any type of function-hiding, but require residual input-hiding. The second is the functionality: we allow distributed evaluation (with some resilience to faults). The second difference is not essential, and is considered in order to reduce the underlying computational assumptions. In particular, a (non-distributed) SFE with residual input-hiding implies DSFE with $k = 1, \tau = 0$.

Remark 2.5 (deterministic Eval). Jumping ahead, we remark that we will use distributed SFE in a setting where the encryptor is always honest. Since we are not requiring any privacy against the encryptor, we may assume w.l.o.g that Eval is deterministic. Indeed, we can always sample any required randomness as part of the encryption process and embed it in the shares $\mathsf{CT}_1, \ldots, \mathsf{CT}_k$.

2.1.1 Instantiations

We now mention known instantiations of SFE and DSFE schemes, which we can rely on.

SFE. As mentioned above, for our application, we will require rather strong function-hiding. To instantiate the scheme we may rely on the SFE protocol obtained by using the oblivious transfer protocol of Naor and Pinkas [NP01] that is based on DDH and is secure against unbounded receivers in conjunction with an information-theoretic variant of Yao's garbled circuit [Yao86] for $\mathbf{NC}^1$ [IK02]. The resulting SFE scheme is for classes of circuits in $\mathbf{NC}^1$, which will suffice for our purposes. Alternatively, we can use a strong enough computational variant of Yao based on sub-exponential one-way functions, resulting in a construction for all polynomial-size circuits.

More generally, the Naor-Pinkas OT can be replaced with any OT that had statistical function-hiding. In the CRS model, such two-message protocols exist from other standard assumptions as well [PVW08]. While our main transformation is described using SFE in the plain model, it can be naturally extended to the CRS setting (see Remark 3.4).

DSFE. An information-theoretically secure DSFE scheme for circuit classes in $\mathbf{NC}^1$ can be obtained based on a non-interactive variant of the BGW protocol [BGW88] similar to that used in [GVW12]. For the sake of completeness, we now outline this variant.

Given a class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}$ in $\mathbf{NC}^1$ defined on inputs in $\{0,1\}^{n(\lambda)}$, we can interpret it as a class of arithmetic circuits where any circuit C is defined over inputs $(x_1, \dots, x_n) \in \mathbb{F}^n$, and computes a polynomial of total degree at most $D = 2^d$, where $d = d(\lambda)$ is the maximal depth of any circuit in $\mathcal{C}_\lambda$.

At a high-level, sharing the inputs in the scheme corresponds to encoding them using the Shamir secret-sharing scheme (that is, as random Reed-Solomon polynomials), evaluation corresponds to homomorphic evaluation over the polynomials, and residual input hiding is guaranteed by adding a random zero polynomial to the evaluated shares.

Concretely, the scheme is defined as follows. Fix a field $\mathbb{F}$, such that $|\mathbb{F}| \geq 3D + 1$, and let $k = 3D + 1$. Let $\alpha_1, \dots, \alpha_k$ be k distinct elements in the field.

- $\mathsf{Enc}(x_1, \dots, x_n)$:
  1. sample n random degree-one polynomials $p_1(\cdot), \dots, p_n(\cdot)$, where $p_i(0) = x_i$,
  2. sample a random degree D polynomial $z(\cdot)$ such that $z(0) = 0$.
  3. set $\mathsf{CT}_j = p_1(\alpha_j), \dots, p_n(\alpha_j), z(\alpha_j)$.
  4. output $\mathsf{CT}_1, \dots, \mathsf{CT}_k$.
- $\mathsf{Eval}(\mathsf{CT}_j, C)$:
  1. parse $\mathsf{CT}_j = \pi_1, \dots, \pi_n, \gamma$,
  2. consider the univariate polynomial $E(\cdot) = C(p_1(\cdot), \dots, p_n(\cdot))$ (that has degree at most D), and compute homomorphically $\eta = C(\pi_1, \dots, \pi_n)$. (The result is meant to be $E(\alpha_j)$.)
  3. output $\widehat{\mathsf{CT}}_j = \eta + \gamma$.
- $\mathsf{Dec}(\widehat{\mathsf{CT}}_1, \dots, \widehat{\mathsf{CT}}_k)$:
  1. parse $(\widehat{\mathsf{CT}}_1, \dots, \widehat{\mathsf{CT}}_k)$ as a Reed-Solomon codeword in $\mathbb{F}^k$, and decode a polynomial $\widetilde{E}(\cdot)$,
  2. output $\widetilde{E}(0)$.

We claim that the above scheme is $(k, \frac{1}{3})$-secure. The analysis follows the standard BGW analysis (detailed in [AL11]). Very roughly, to show correctness, note that by the homomorphic properties of the Reed-Solomon code the correct polynomial E is such that $E(0) = C(x_1, \dots, x_n)$, and this also holds for $E(\cdot) + z(\cdot)$. Furthermore, decoding such that $\tilde{E} = E + z$ is guaranteed as long as there are at most D faults. Input-hiding follows from the fact that each individual CT_j is distributed uniformly at random on $\mathbb{F}$. Residual input-hiding follows by the fact that after adding z, the new polynomial $E + z$ is a random polynomial with free coefficient $C(x_1, \dots, x_n)$, and thus can be completely simulated from this value. For more details, see [BGW88, AL11].

Remark 2.6 (complexity of encryption). One measure of interest, when considering our application to correcting errors in functional encryption, will be the complexity of the encryption procedure. Here we note that this size is $O(nk \log D) = O(nD \log D) = n \cdot 2^{O(d)}$; namely, it does not grow with the size of the circuits, but does grow exponentially with the maximal depth d of the circuits. (As will be discussed later on, this will still be good enough in our context, to bootstrap functional encryption to indistinguishability obfuscation, as shown in [AJ15, BV15].)

One point to notice is that the above is not entirely accurate if the output of the circuit C is a large $\ell = \ell(\lambda)$. Indeed, naïvely to guarantee residual input-privacy, we will need to generate ℓ separate polynomials $z_1, \ldots, z_\ell$, meaning the encryption size will grow linearly with ℓ. This can be avoided by shifting the randomness to the evaluation procedure (which will slightly complicate our transformation). Alternatively, this can be avoided assuming the existence of a pseudo-random generator, by adding to the ciphertexts a seed, and having Eval use it to generate the multiple polynomials $z_1, \ldots, z_\ell$.

2.2 Symmetric Encryption

A symmetric encryption scheme Sym consists of a tuple of two PPT algorithms (Sym.Enc, Sym.Dec). The encryption algorithm takes as input a symmetric key $\mathsf{SK} \in \{0,1\}^\lambda$, where λ is the security parameter, and a message $m \in \{0,1\}^*$ of polynomial size in the security parameter, and outputs a ciphertext SCT. The decryption algorithm takes as input (SK, SCT), and outputs the decrypted message m. For this work, we only require one-time security.

Definition 2.7 (One-Time Symmetric Encryption)**.** *A pair of PPT algorithms* (Sym.Enc, Sym.Dec) *is a one-time symmetric encryption scheme for message space* $\{0,1\}^*$ *if it satisfies:*

1. **Correctness:** *For every security parameter* λ *and message* $m \in \{0,1\}^*$*,*

$$\Pr\left[\mathsf{Sym.Dec}(\mathsf{SK},\mathsf{SCT}) = m \;\middle|\; \begin{array}{c} \mathsf{SK} \leftarrow \{0,1\}^\lambda \\ \mathsf{SCT} \leftarrow \mathsf{Sym.Enc}(\mathsf{SK}, m) \end{array}\right] = 1 \ .$$

2. **Indistinguishability:** *for any polysize distinguisher* $\mathcal{D}$ *there exists a negligible function* $\mu(\cdot)$*, such that for all* $\lambda \in \mathbb{N}$*, and any equal size messages* m_0, m_1*,*

$$|\Pr[\mathcal{D}(\mathsf{Sym.Enc}(\mathsf{SK}, m_0)) = 1] - \Pr[\mathcal{D}(\mathsf{Sym.Enc}(\mathsf{SK}, m_1)) = 1]| \le \mu(\lambda) \ ,$$

   where $\mathsf{SK} \leftarrow \{0,1\}^\lambda$.

   We further say that Sym *is* δ*-secure, for some concrete negligible function* $\delta(\cdot)$*, if for all polysize distinguishers the above indistinguishability gap* $\mu(\lambda)$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$*.*

A symmetric encryption scheme meeting this definition can be constructed from any pseudo-random generator, and thus any one-way function [HILL99].

2.3 Puncturable Pseudorandom Functions

We consider a simple case of puncturable pseudo-random functions (PRFs) where any PRF may be punctured at a single point. The definition is formulated as in [SW14], and is satisfied by the Goldreich-Goldwasser-Micali PRF construction [GGM86, BW13, KPTZ13, BGI14].

Definition 2.8 (Puncturable PRFs)**.** *Let n, k be polynomially bounded length functions. An efficiently computable family of functions*

$$\mathcal{PRF} = \left\{ \mathsf{PRF}_{\mathsf{K}} : \{0,1\}^* \to \{0,1\}^\lambda \;\middle|\; \mathsf{K} \in \{0,1\}^{k(\lambda)}, \lambda \in \mathbb{N} \right\} \ ,$$

associated with an efficient (probabilistic) key sampler $\mathsf{Gen}_{\mathcal{PRF}}$, is a puncturable PRF *if there exists a poly-time puncturing algorithm* Punc *that takes as input a key* K*, and a point x^*, and outputs a punctured key $\mathsf{K}\{x^*\}$, so that the following conditions are satisfied:*

1. **Functionality is preserved under puncturing:** *For every $x^* \in \{0,1\}^*$,*

$$\Pr_{\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^\lambda)} \left[\forall x \neq x^* : \mathsf{PRF}_{\mathsf{K}}(x) = \mathsf{PRF}_{\mathsf{K}\{x^*\}}(x) \;\middle|\; \mathsf{K}\{x^*\} = \mathsf{Punc}(\mathsf{K}, x^*) \right] = 1 \ .$$

2. **Indistinguishability at punctured points:** *for any polysize distinguisher $\mathcal{D}$ there exists a negligible function $\mu(\cdot)$, such that for all $\lambda \in \mathbb{N}$, and any $x^* \in \{0,1\}^*$,*

$$|\Pr[\mathcal{D}(x^*, \mathsf{K}\{x^*\}, \mathsf{PRF}_{\mathsf{K}}(x^*)) = 1] - \Pr[\mathcal{D}(x^*, \mathsf{K}\{x^*\}, u) = 1]| \leq \mu(\lambda) \ ,$$

   where $\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^\lambda), \mathsf{K}\{x^*\} = \mathsf{Punc}(\mathsf{K}, x^*)$, and $u \leftarrow \{0,1\}^\lambda$.

   We further say that $\mathcal{PRF}$ is δ-secure, for some concrete negligible function $\delta(\cdot)$, if for all polysize distinguishers the above indistinguishability gap $\mu(\lambda)$ is smaller than $\delta(\lambda)^{\Omega(1)}$.

Remark 2.9 (uniform output)*.* For some of our constructions, it will be convenient to assume that the PRF family is *one-universal*; that is, for any fixed x, $\mathsf{PRF}_{\mathsf{K}}(x)$ is distributed uniformly at random (when K is sampled at random). It is not hard to see that such a puncturable PRF can be easily obtained from any puncturable PRF by adding a random string U to the key and XORing U to every output.

3 Correcting Errors in Indistinguishability Obfuscation

In this section, we define approximate IO and show how to transform any approximate IO to (almost) perfectly correct IO, based on SFE.

3.1 Approximate and Exact IO

We start by defining indistinguishability obfuscation (IO) with almost perfect correctness. The definition is formulated as in [BGI+12].

Definition 3.1 (Indistinguishability obfuscation)**.** *A PPT algorithm $\mathcal{O}$ is said to be an* indistinguishability obfuscator *for a class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}$, if it satisfies:*

1. **Almost Perfect Correctness:** *for any security parameter λ and $C \in \mathcal{C}_\lambda$,*

$$\Pr_{\mathcal{O}} \left[\forall x : \mathcal{O}(C, 1^\lambda)(x) = C(x) \right] \geq 1 - 2^{-\lambda} \ .$$

2. **Indistinguishability:** *for any polysize distinguisher $\mathcal{D}$ there exists a negligible function $\mu(\cdot)$, such that for any two circuits $C_0, C_1 \in \mathcal{C}$ that compute the same function and are of the same size:*

$$\left|\Pr[\mathcal{D}(\mathcal{O}(C_0, 1^\lambda)) = 1] - \Pr[\mathcal{D}(\mathcal{O}(C_1, 1^\lambda)) = 1]\right| \le \mu(\lambda) \ ,$$

where the probability is over the coins of $\mathcal{D}$ and $\mathcal{O}$.

We further say that $\mathcal{O}$ is δ-secure, for some concrete negligible function $\delta(\cdot)$, if for all polysize distinguishers the above indistinguishability gap $\mu(\lambda)$ is smaller than $\delta(\lambda)^{\Omega(1)}$.

We now define an approximate notion of correctness that allows the obfuscated circuit to err with some probability over inputs taken from some samplable distribution.

Definition 3.2 (($\alpha, \mathcal{X}$)-correct IO)**.** *For $\alpha(\lambda) \in [0, 1]$ and an ensemble of input samplers $\mathcal{X} = \{\mathcal{X}_\lambda\}$, we say that $\mathcal{O}$ is $(\alpha, \mathcal{X})$-correct if instead of (almost) perfect correctness, it satisfies the following relaxed requirement:*

1. **Approximate Correctness:** *for any security parameter λ, $C \in \mathcal{C}_\lambda$,*

$$\Pr\left[\mathcal{O}(C, 1^\lambda)(x) = C(x) \;\middle|\; x \leftarrow \mathcal{X}_\lambda\right] \ge \alpha(\lambda) \ ,$$

where the probability is also over the coins of $\mathcal{O}$.

3.2 The Transformation

We now describe a transformation from approximately correct IO to (almost) perfectly correct IO and analyze it. The transformation is based on SFE satisfying a strong function-hiding guarantee. We discuss an instantiations based on standard computational assumptions in Section 3.3.

In Section 5, we discuss an alternative transformation through functional encryption based on weaker computational assumptions.

A Worst-Case Approximate Obfuscator. The main step of the transformation is to obtain random self-reducibility; that is,to convert an approximate obfuscator $\mathsf{ap}\mathcal{O}$, which works reasonably well on average for random inputs taken from an appropriate distribution, into a worst-case approximate obfuscator $\mathsf{wc}\mathcal{O}$ that, for any (worst-case) input, works well on average over the random coins of the obfuscator alone. Then, in the second step, we invoke standard "BPP amplification".

Ingredients. In the following, let λ denote a security parameter, let $\varepsilon < 1$ be some constant, $\eta(\lambda) = \lambda^{-\Omega(1)}$ and let $\mathcal{C} = \{\mathcal{C}_\lambda\}$ denote a circuit class. We rely on the following primitives:

- A secure function evaluation scheme SFE for $\mathcal{C}$ that is $2^{-\omega(\sigma(\lambda)+\log\lambda)}$-function-hiding, where $\sigma(\lambda)$ is the length of fresh ciphertexts generated by the encryption algorithm Enc for security parameter λ (and inputs of size $n = n(\lambda)$ in the domain of $\mathcal{C}_\lambda$).

- A $2^{-\tilde{\lambda}^\varepsilon}$-secure puncturable pseudo-random function family $\mathcal{PRF}$, where the security parameter is $\tilde{\lambda} = \omega(\sigma(\lambda) + \log\lambda)^{1/\varepsilon}$.

- A $(\frac{1}{2} + \eta(\lambda), \mathcal{X})$-correct, $2^{-\tilde{\lambda}^\varepsilon}$-secure indistinguishability obfuscator $\mathsf{ap}\mathcal{O}$ for $\overline{\mathcal{C}}$, where the security parameter is $\tilde{\lambda} = \omega(\sigma(\lambda) + \log\lambda)^{1/\varepsilon}$. The sampler class $\mathcal{X}$ depends on SFE and the class $\overline{\mathcal{C}}$ depends on SFE, $\mathcal{PRF}$, and $\mathcal{C}$. Both $\mathcal{X}$ and $\overline{\mathcal{C}}$ are specified below as part of the description of the constructed (exact) obfuscator $\mathcal{O}$.

The Worst-Case Obfuscator $\mathsf{wc}\mathcal{O}$:
Given a circuit $C : \{0,1\}^n \to \{0,1\}$ and security parameter λ, the obfuscator $\mathsf{wc}\mathcal{O}(C, 1^\lambda)$

1. computes a new security parameter $\tilde{\lambda} = \omega(\sigma(\lambda) + \log \lambda)^{1/\varepsilon}$, where $\sigma(\lambda)$ is the length of ciphertexts as defined above,
2. samples a puncturable PRF seed $\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^{\tilde{\lambda}})$,
3. computes the augmented C-evaluation circuit C_{K} defined in Figure 1,
4. outputs an approximate obfuscation $\widetilde{C} \leftarrow \mathsf{ap}\mathcal{O}(C_{\mathsf{K}}, 1^{\tilde{\lambda}})$.

C_{K}

Hardwired: the circuit C and the seed K.
Input: a ciphertext CT.

1. derive randomness $r \leftarrow \mathsf{PRF}_{\mathsf{K}}(\mathsf{CT})$,
2. compute $\widehat{\mathsf{CT}} \leftarrow \mathsf{Eval}(\mathsf{CT}, C; r)$,
3. output $\widehat{\mathsf{CT}}$.

Padding: the circuit is padded to be of size $\ell(|C|, \mathsf{K})$, for some polynomial ℓ specified in the analysis.

Figure 1: The augmented C-evaluation circuit.

We next describe the how the obfuscation $\widetilde{C}$ is evaluated on any input x via a randomized procedure.

Randomized Evaluation:
Given an obfuscation $\widetilde{C}$, an input $x \in \{0,1\}^n$, and security parameter λ:

1. compute $(\mathsf{CT}, \mathsf{R}) \leftarrow \mathsf{Enc}(x)$,
2. compute $\widehat{\mathsf{CT}} = \widetilde{C}(\mathsf{CT})$,
3. output $y = \mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}})$.

The ensemble of samplers $\mathcal{X}$ consists of samplers $\mathcal{X}^{\mathbf{0}}$ that sample encryptions from $\mathsf{Enc}(0^n)$ whereas the class $\overline{\mathcal{C}}$ consists of circuits C_{K} as defined in Figure 1.

Proposition 3.1. $\mathsf{wc}\mathcal{O}$ *satisfies:*

1. **Worst-Case Approximate Correctness:** *for any λ, $C \in \mathcal{C}_\lambda$, $x \in \{0,1\}^n$,*
$$\Pr\left[\mathcal{O}(C, 1^\lambda)(x) = C(x)\right] \geq \frac{1}{2} + \eta(\lambda) - \lambda^{-\omega(1)} \ ,$$
where the probability is over the coins of $\mathsf{ap}\mathcal{O}$.
2. **Indistinguishability:** *as in Definition 3.1.*

The intuition behind the proof is outlined in the introduction. We now turn to the actual proof.

Proof. We first prove that the new obfuscator is worst-case approximately correct, and then prove that it is secure.

Correctness. For any $\lambda, n = n(\lambda)$, input $x \in \{0,1\}^n$, let us denote $\mathcal{X}^x := \mathsf{Enc}(x)$ a sampler for encryptions of x. Then, by the input-hiding guarantee of SFE, and the approximate correctness of $\mathsf{ap}\mathcal{O}$, we claim that the approximate obfuscation is correct on encryptions of an arbitrary $x \in \{0,1\}^n$ as on encryptions of 0^n. That is, there exists a negligible $\mu(\lambda)$ such that

$$\Pr\left[\widetilde{C}(\mathsf{CT}) = C_{\mathsf{K}}(\mathsf{CT}) \;\middle|\; \mathsf{CT} \leftarrow \mathcal{X}^x\right] \geq$$
$$\Pr\left[\widetilde{C}(\mathsf{CT}) = C_{\mathsf{K}}(\mathsf{CT}) \;\middle|\; \mathsf{CT} \leftarrow \mathcal{X}^{\mathbf{0}}\right] - \mu(\lambda) \geq$$
$$\frac{1}{2} + \eta(\lambda) - \mu(\lambda) \ ,$$

where in both of the above $\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^{\tilde{\lambda}})$, $\widetilde{C} \leftarrow \mathsf{ap}\mathcal{O}(C_{\mathsf{K}}, 1^{\tilde{\lambda}})$.

It now follows that decryption is correct with probability noticeably larger than half. Concretely,

$$\Pr\left[\mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}}) = C(x) \;\middle|\; \begin{array}{c} \mathsf{CT}, \mathsf{R} \leftarrow \mathsf{Enc}(x) \\ \widehat{\mathsf{CT}} = \widetilde{C}(\mathsf{CT}) \end{array}\right] \geq$$
$$\Pr\left[\mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}}) = C(x) \;\middle|\; \begin{array}{c} \mathsf{CT}, \mathsf{R} \leftarrow \mathsf{Enc}(x) \\ \widehat{\mathsf{CT}} = C_{\mathsf{K}}(\mathsf{CT}) \end{array}\right] \cdot \Pr\left[\widetilde{C}(\mathsf{CT}) = C_{\mathsf{K}}(\mathsf{CT}) \;\middle|\; \mathsf{CT} \leftarrow \mathcal{X}^x\right] =$$
$$\Pr\left[\mathsf{Dec}(\mathsf{R}, \widehat{\mathsf{CT}}) = C(x) \;\middle|\; \begin{array}{c} \mathsf{CT}, \mathsf{R} \leftarrow \mathsf{Enc}(x) \\ r = \mathsf{PRF}_{\mathsf{K}}(\mathsf{CT}) \\ \widehat{\mathsf{CT}} = \mathsf{Eval}(\mathsf{CT}, C; r) \end{array}\right] \cdot \Pr\left[\widetilde{C}(\mathsf{CT}) = C_{\mathsf{K}}(\mathsf{CT}) \;\middle|\; \mathsf{CT} \leftarrow \mathcal{X}^x\right] \geq$$
$$(1 - \mu'(\lambda)) \cdot \left(\frac{1}{2} + \eta(\lambda) - \mu(\lambda)\right) \ ,$$

where in all of the above $\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^{\tilde{\lambda}})$, $\widetilde{C} \leftarrow \mathsf{ap}\mathcal{O}(C_{\mathsf{K}}, 1^{\tilde{\lambda}})$, and $\mu'(\cdot)$ is some negligible function (corresponding to the negligible decryption error of SFE). In the last step, we relied on the fact that for any fixed CT, $\mathsf{PRF}_{\mathsf{K}}(\mathsf{CT})$ is distributed uniformly at random (Remark 2.9), and the (almost) perfect correctness of SFE.

This completes the proof of correctness.

Security Analysis. Consistently with the notation above, for $\mathsf{K} \leftarrow \mathsf{Gen}_{\mathcal{PRF}}(1^{\tilde{\lambda}})$, and a circuit $C \in \mathcal{C}_\lambda$, we denote by $\widetilde{C} \leftarrow \mathsf{ap}\mathcal{O}(C_{\mathsf{K}}, 1^{\tilde{\lambda}})$ the corresponding approximate obfuscation of the (derandomized) evaluation circuit. We show that for any polysize distinguisher there exists a neglgible $\mu(\cdot)$, such that for any $C_0, C_1 \in \mathcal{C}_\lambda$ that compute the same function it holds that

$$\left|\Pr[\mathcal{D}(\widetilde{C}_0) = 1] - \Pr[\mathcal{D}(\widetilde{C}_1) = 1]\right| \leq \mu(\lambda) \ .$$

Roguhly, the above follows from the fact that the output of the two underlying obfuscated circuits on any point $\mathsf{CT} \in \{0,1\}^{\sigma(\lambda)}$ is indistinguishable even given C_0, C_1. Indeed, because the circuits C_0, C_1 compute the same function, by the function-hiding of SFE, for any ciphertext $\mathsf{CT} \in \{0,1\}^{\sigma(\lambda)}$, the evaluated ciphers $\mathsf{Eval}(\mathsf{CT}, C_0)$ and $\mathsf{Eval}(\mathsf{CT}, C_1)$ are indistinguishable. Canetti, Lin, Tessaro, and

Vaikuntanathan [CLTV15] show that (sub-exponential) IO in conjunction with (sub-exponential) puncuturable PRFs are sufficient in this setting, which they formalize by *probabilistic IO* notion. For the sake of completeness, we next sketch the argument.

We consider a sequence of $2^\sigma + 1$ hybrids $\{\mathcal{H}_{\mathsf{CT}}\}_{\mathsf{CT} \in \{0,\dots,2^\sigma\}}$, where we naturally identify integers in $[2^\sigma]$ with strings in $\{0,1\}^\sigma$. In $\mathcal{H}_{\mathsf{CT}}$, we obfuscate a circuit $\mathbb{C}_{\mathsf{CT}}(\mathsf{CT}')$ that computes $C_{0,\mathsf{K}}$ for all $\mathsf{CT}' > \mathsf{CT}$ and $C_{1,\mathsf{K}}$ for all $\mathsf{CT}' \leq \mathsf{CT}$; the circuit is padded to size ℓ (as in Figure 1).

We first note that $\mathbb{C}_0$ computes the same function as $C_{0,\mathsf{K}}$ and that $\mathbb{C}_{2^\sigma}$ computes the same function as $C_{1,\mathsf{K}}$, and thus by the IO security,

$$\left|\Pr\left[\mathcal{D}(\widetilde{C}_0) = 1\right] - \Pr\left[\mathcal{D}(\mathcal{H}_0) = 1\right]\right| \leq 2^{-\tilde{\lambda}^\varepsilon} \;,$$
$$\left|\Pr\left[\mathcal{D}(\mathcal{H}_{2^\sigma}) = 1\right] - \Pr\left[\mathcal{D}(\widetilde{C}_1) = 1\right]\right| \leq 2^{-\tilde{\lambda}^\varepsilon} \;.$$

We show that for any $\mathsf{CT} \in [2^\sigma]$,

$$\left|\Pr\left[\mathcal{D}(\mathcal{H}_{\mathsf{CT}-1}) = 1\right] - \Pr\left[\mathcal{D}(\mathcal{H}_{\mathsf{CT}}) = 1\right]\right| \leq O(2^{-\tilde{\lambda}^\varepsilon}) \;.$$

This follows a standard puncturing argument with respect to the point CT, consisting of:

- puncturing PRF_K at CT, and hardwiring $C_{0,\mathsf{K}}(\mathsf{CT}) = \mathsf{Eval}(\mathsf{CT}, C_0; \mathsf{PRF}_\mathsf{K}(\mathsf{CT}))$, which relies on IO security,
- replacing $\mathsf{PRF}_\mathsf{K}(\mathsf{CT})$ with true randomness, which relies on pseudorandomness at punctured points,
- replacing $\mathsf{Eval}(\mathsf{CT}, C_0)$ with $\mathsf{Eval}(\mathsf{CT}, C_1)$, which relies on function hiding.
- reversing the above steps.

Each of the steps induces a loss of $2^{-\tilde{\lambda}^\varepsilon} = 2^{-\omega(\sigma(\lambda)+\log\lambda)}$ in the indistinguishability gap.

This completes the security analysis.

□

The (Almost) Exact Obfuscator $\mathcal{O}$: to obtain an (almost) exact obfuscator $\mathcal{O}$ from the worst-case approximate obfuscator we apply standard "BPP amplification". Such a transformation is given in [KMN+14, Appendix B]. For the sake of completeness, we sketch it here.

Obfuscation: Given a circuit $C : \{0,1\}^n \to \{0,1\}$ and security parameter λ, the obfuscator $\mathcal{O}(C, 1^\lambda)$ outputs $N = \frac{\omega(n+\lambda)}{\eta^2(\lambda)}$ obfuscations $\widetilde{C}_1, \dots, \widetilde{C}_N$, where $\widetilde{C}_i \leftarrow \mathsf{wc}\mathcal{O}(C, 1^\lambda)$, and N random strings $r_1, \dots, r_N$, where $r_i \leftarrow \{0,1\}^\lambda$.

Evaluation: Given an obfuscation $\left\{\widetilde{C}_i, r_i \;\middle|\; i \in [N]\right\}$, input $x \in \{0,1\}^n$, and security parameter λ:

1. For $i \in [N]$, invoke the randomized evaluation procedure for $\widetilde{C}_i$, for input x, using randomness r_i, store the result y_i.
2. Output $y = \mathsf{majority}(y_1, \dots, y_N)$.

Remark 3.3 (deterministic evaluator)*.* Publishing the random strings r_i is done to match the usual obfuscation syntax where the evaluation is deterministic. We may also let the evaluator sample this randomness.

Proposition 3.2. *$\mathcal{O}$ is an (almost) perfectly correct indistinguishability obfuscator.*

Proof sketch. We show correctness and security.

Correctness. By a Chernoff bound, for large enough λ, and any $x \in \{0,1\}^n$, the probability that the majority value y among all decrypted $y_1, \ldots, y_N$ is incorrect is bounded by

$$\Pr\left[y \neq C(x)\right] \leq 2^{-\Omega(N \cdot \eta^2(\lambda))} \leq 2^{-n+\lambda} \ .$$

The required correctness follows by a union bound over all inputs in $\{0,1\}^n$.

Security. The obfuscation consists of N independent copies of worst-case obfuscations $\widetilde{C}_i \leftarrow \mathsf{wc}\mathcal{O}(C)$, where $\mathsf{wc}\mathcal{O}$ satisfies indistinguishability. Security thus follows by a standard hybrid argument. □

Remark 3.4 (SFE in the CRS model)*.* The above construction can be naturally extended to rely also on non-interactive SFE schemes in the CRS model (rather than the plain model). Indeed, the CRS can be generated by the (honest) obfuscator.

3.3 Instantiating the Scheme

As discussed in Section 2.1.1, we can instantiate the SFE based on the (polynomial) DDH assumption and sub-exponential one-way functions. Sub-exponential one-way functions are also needed here in order to obtain sub-exponentially-secure puncturable PRFs.

We can thus state the following theorem

Theorem 3.5. *Assuming sub-exponentially secure approximate IO for* $\mathbf{P}/\mathbf{poly}$*, (polynomial) DDH, and sub-exponentially-secure one-way functions, there exists (almost) perfectly correct IO for* $P/\mathbf{poly}$*.*

Alternative instantiations of the above under more computational assumptions [PVW08] can be obtained when extending the scheme to SFE in the CRS model.

4 Correcting Errors in Functional Encryption

In this section, we define approximate FE and show how to transform any approximate FE to (almost) perfectly correct FE, based on DSFE. For the sake of concreteness, we focus on the public-key setting. We also focus on selective-security, which can be generically boosted to adaptive security [ABSV14].

4.1 Approximate and Exact FE

We recall the definition of public-key functional encryption (FE) with selective indistinguishability-based security [BSW12, O'N10], and extend the definition to the case of approximate correctness.

A public-key functional encryption (FE) scheme FE, for a function class $\mathcal{F} = \{\mathcal{F}_\lambda\}$ (represented by boolean circuits) and message space $\mathcal{M} = \left\{\{0,1\}^{n(\lambda)} : \lambda \in \mathbb{N}\right\}$, consists of four PPT algorithms $(\mathsf{FE.Setup}, \mathsf{FE.Gen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ with the following syntax:

- $\mathsf{FE.Setup}(1^\lambda)$: Takes as input a security parameter λ in unary and outputs a (master) public key and a secret key $(\mathsf{MPK}, \mathsf{MSK})$.
- $\mathsf{FE.Gen}(\mathsf{MSK}, f)$: Takes as input a secret key MSK, a function $f \in \mathcal{F}_\lambda$ and outputs a functional key FSK_f.
- $\mathsf{FE.Enc}(\mathsf{MPK}, M)$: Takes as input a public key MPK, a message $M \in \{0,1\}^{n(\lambda)}$ and outputs an encryption of M.
- $\mathsf{FE.Dec}(\mathsf{FSK}_f, \mathsf{FCT})$: Takes as input a functional key FSK_f, a ciphertext FCT and outputs $\widehat{M}$.

We next recall the required security properties as well the common (almost) perfect correctness requirement.

Definition 4.1 (Selectively-secure public-key FE)**.** *A tuple of PPT algorithms* $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Gen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *is a selectively-secure public-key functional encryption scheme, for function class* $\mathcal{F} = \{\mathcal{F}_\lambda\}$*, and message space* $\mathcal{M} = \left\{\{0,1\}^{n(\lambda)} : \lambda \in \mathbb{N}\right\}$*, if it satisfies:*

1. **Almost Perfect Correctness:** *for every* $\lambda \in \mathbb{N}$*, message* $M \in \{0,1\}^{n(\lambda)}$*, and function* $f \in \mathcal{F}_\lambda$*,*

$$\Pr\left[f(M) \leftarrow \mathsf{FE.Dec}(\mathsf{FSK}_f, \mathsf{FCT}) \;\middle|\; \begin{array}{c} (\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{FE.Setup}(1^\lambda) \\ \mathsf{FSK}_f \leftarrow \mathsf{FE.Gen}(\mathsf{MSK}, f) \\ \mathsf{FCT} \leftarrow \mathsf{FE.Enc}(\mathsf{MPK}, M) \end{array} \right] \geq 1 - 2^{-\lambda}.$$

2. **Selective-security:** *for any polysize adversary* $\mathcal{A}$*, there exists a negligible function* $\mu(\lambda)$ *such that for any* $\lambda \in \mathbb{N}$*, it holds that*

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

where for each $b \in \{0,1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

   (a) *The adversary submits the challenge message-pair* $M_0, M_1 \in \{0,1\}^{n(\lambda)}$ *to the challenger.*

   (b) *The challenger executes* $\mathsf{FE.Setup}(1^\lambda)$ *to obtain* $(\mathsf{MPK}, \mathsf{MSK})$*. It then executes* $\mathsf{FE.Enc}(\mathsf{MPK}, M_b)$ *to obtain* FCT*. The challenger sends* $(\mathsf{MPK}, \mathsf{FCT})$ *to the adversary.*

   (c) *The adversary submits function queries to the challenger. For any submitted function query* $f \in \mathcal{F}_\lambda$*, if* $f(M_0) = f(M_1)$*, the challenger generates and sends* $\mathsf{FSK}_f \leftarrow \mathsf{FE.Gen}(\mathsf{MSK}, f)$*. In any other case, the challenger aborts.*

   (d) *The output of the experiment is the output of* $\mathcal{A}$*.*

We further say that FE *is* δ*-secure, for some concrete negligible function* $\delta(\cdot)$*, if for all polysize adversaries the above indistinguishability gap* $\mu(\lambda)$ *is smaller than* $\delta(\lambda)^{\Omega(1)}$*.*

We now define an approximate notion of correctness that allows decryption to error with some probability over encryption of messages taken from some given distribution.

Definition 4.2 ($(\alpha, \mathcal{X})$-correct FE)**.** *For* $\alpha(\lambda) \in [0,1]$ *and an ensemble of samplers* $\mathcal{X} = \{\mathcal{X}_\lambda\}$ *with support* $\mathcal{M} = \left\{\{0,1\}^{n(\lambda)} : \lambda \in \mathbb{N}\right\}$*, we say that* FE *is* $(\alpha, \mathcal{X})$*-correct if instead of (almost) perfect correctness, it satisfies the following relaxed requirement:*

1. **Approximate Correctness:** *for every* $\lambda \in \mathbb{N}$, *and function* $f \in \mathcal{F}_\lambda$,

$$\Pr\left[f(M) \leftarrow \mathsf{apFE.Dec}(\mathsf{apFSK}_f, \mathsf{FCT}) \;\middle|\; \begin{array}{c} (\mathsf{apMPK}, \mathsf{apMSK}) \leftarrow \mathsf{apFE.Setup}(1^\lambda) \\ \mathsf{apFSK}_f \leftarrow \mathsf{apFE.Gen}(\mathsf{apMSK}, f) \\ M \leftarrow \mathcal{X}_\lambda \\ \mathsf{apFCT} \leftarrow \mathsf{apFE.Enc}(\mathsf{apMPK}, M) \end{array} \right] \geq \alpha(\lambda).$$

4.2 The Transformation

We now describe the transformation from approximately correct FE to (almost) perfectly correct FE and analyze it. The transformation is based on DSFE. We discuss instantiations in Section 4.3.

A Worst-Case Approximate FE. As in the case of obfuscation, the main step of the FE transformation is to obtain random self-reducibility; that is,to convert an approximate FE scheme apFE, which works reasonably well on average for random messages taken from some appropriate distribution, into a worst-case approximate scheme wcFE that, for any (worst-case) message, works well on average over the random coins of the obfuscator alone. Then, in the second step, we invoke standard "BPP amplification".

Ingredients. In the following, let λ denote a security parameter, and let $\mathcal{F} = \{\mathcal{F}_\lambda\}$ denote a function class. Consider functions $k(\lambda) \in \mathbb{N}$, and $\rho(\lambda), \eta(\lambda) \in [0,1]$ such that $\eta = \frac{1}{2} - \sqrt{\rho} \in [\frac{1}{\lambda^{O(1)}}, \frac{1}{2} - \frac{1}{\lambda^{O(1)}}]$. We rely on the following primitives:

- A $(k, \sqrt{\rho})$-secure distributed function evaluation scheme DSFE for $\mathcal{C}$. We shall further assume that when encrypting an input, the shares $\mathsf{CT}_1, \ldots, \mathsf{CT}_k$ all have the same marginal distribution (i.e., $\mathsf{CT}_i \equiv \mathsf{CT}_j$).[2]
- A $(1 - \rho, \mathcal{X})$-correct (single-key, selectively-secure) functional encryption scheme apFE = (apFE.Setup, apFE.Gen, apFE.Enc, apFE.Dec) for $\overline{\mathcal{C}}$. The sampler class $\mathcal{X}$ depends on DSFE and the class $\overline{\mathcal{F}}$ depends on DSFE, and $\mathcal{F}$. Both $\mathcal{X}$ and $\overline{\mathcal{F}}$ are specified below as part of the description of the constructed (exact) scheme FE.

The Worst-Case Scheme wcFE: The scheme wcFE, for function class $\mathcal{F} = \{\mathcal{F}_\lambda\}$ and message space $\mathcal{M} = \left\{\{0,1\}^{n(\lambda)} : \lambda \in \mathbb{N}\right\}$, consists of the algorithms (wcFE.Setup, wcFE.Gen, wcFE.Enc, wcFE.Dec) defined as follows:

- $\mathsf{wcFE.Setup}(1^\lambda)$: generate $(\mathsf{apMPK}, \mathsf{apMSK}) \leftarrow \mathsf{apFE.Setup}(1^\lambda)$. The public key MPK and secret key MSK are accordingly set to be the apMPK and apMSK.
- $\mathsf{wcFE.Gen}(\mathsf{wcMSK}, f)$: sample $\mathsf{SCT} \leftarrow \mathsf{Sym.Enc}(\mathsf{SK}, 0^{\ell \times k})$, where $\ell = \ell(\lambda)$ is a polynomial specified in the security analysis, and $\mathsf{SK} \leftarrow \{0,1\}^\lambda$. Consider the augmented f-evaluation function f_{SCT} as defined in Figure 2. Generate $\mathsf{apFSK}_{\mathsf{SCT}} \leftarrow \mathsf{apFE.Gen}(\mathsf{apMSK}, f_{\mathsf{SCT}})$. The functional key wcFSK_f will consists of the functional key $\mathsf{apFSK}_{\mathsf{SCT}}$.
- $\mathsf{wcFE.Enc}(\mathsf{wcMPK}, M)$:
  1. Compute $(\mathsf{CT}_1, \ldots, \mathsf{CT}_k, \mathsf{R}) \leftarrow \mathsf{DSFE.Enc}(M)$,

[2] This is just to simplify the construction and is satisfied the instantiation discussed in Section 2. In Remark 4.3, we explain how this assumption can be removed (at the cost of complicating the construction).

2. For $j \in [k]$
   - let $\mathsf{ap}M_j = (\mathtt{norm}, \mathsf{CT}_j, \bot, \bot)$
   - generate $\mathsf{apFCT}_j \leftarrow \mathsf{apFE.Enc}(\mathsf{apMPK}, \mathsf{ap}M_j)$.

Output $\mathsf{wcFCT} = \{\mathsf{apFCT}_1, \ldots, \mathsf{apFCT}_k, \mathsf{R}\}$.

- $\mathsf{wcFE.Dec}(\mathsf{wcFSK}_f, \mathsf{wcFCT})$:

  1. Parse $\mathsf{wcFSK}_f = \mathsf{apFSK}_{\mathsf{SCT}}$ and $\mathsf{wcFCT} = (\mathsf{apFCT}_1, \ldots, \mathsf{apFCT}_k, \mathsf{R})$.
  2. for $j \in [k]$, compute $\widehat{\mathsf{CT}}_j \leftarrow \mathsf{apFE.Dec}(\mathsf{apFSK}_{\mathsf{SCT}}, \mathsf{apFCT}_j)$.
  3. output $y = \mathsf{DSFE.Dec}(\mathsf{R}, \widehat{\mathsf{CT}}_1, \ldots, \widehat{\mathsf{CT}}_k)$.

f_{SCT}

Hardwired: a symmetric key ciphertext SCT.
Input $\mathsf{ap}M = (b, \mathsf{CT}, \mathsf{SK}, j)$**:**

- a flag bit b,
- a DSFE ciphertext CT_j,
- a symmetric encryption key SK.
- index $j \in [k]$.

1. If $b = \mathtt{norm}$ (normal mode of operation, ignoring inputs SK, j),
   - compute $\widehat{\mathsf{CT}} = \mathsf{Eval}(\mathsf{CT}, f)$.
2. If $b = \mathtt{trap}$ (trapdoor mode of operation, ignoring inputs CT, K),
   - compute $(\widehat{\mathsf{CT}}_1, \ldots, \widehat{\mathsf{CT}}_k) = \mathsf{Sym.Dec}(\mathsf{SK}, \mathsf{SCT})$,
   - let $\widehat{\mathsf{CT}} := \widehat{\mathsf{CT}}_j$.
3. Output $\widehat{\mathsf{CT}}$.

Figure 2: The augmented f-evaluation circuit.

The ensemble of samplers $\mathcal{X}$ consists of samplers $\mathcal{X}^{\mathbf{0}}$ that sample FE plaintexts of the form $\mathsf{ap}M = (\mathtt{norm}, \mathsf{CT}, \bot, \bot)$ where CT is the first of k ciphertext components sampled from $\mathsf{DSFE.Enc}(0^n)$, i.e. it is a share of a zero-encryption in the underlying DSFE scheme. The class $\overline{\mathcal{F}}$ consists of circuits f_{SCT} as in Figure 2.

Proposition 4.1. wcFE *satisfies:*

1. **Worst-Case Approximate Correctness:** *for every $\lambda \in \mathbb{N}$, function $f \in \mathcal{F}_\lambda$, and message $M \in \{0,1\}^n$,*

$$\Pr\left[f(M) \leftarrow \mathsf{wcFE.Dec}(\mathsf{wcFSK}_f, \mathsf{wcFCT}) \;\middle|\; \begin{array}{c} (\mathsf{wcMPK}, \mathsf{wcMSK}) \leftarrow \mathsf{wcFE.Setup}(1^\lambda) \\ \mathsf{wcFSK}_f \leftarrow \mathsf{wcFE.Gen}(\mathsf{wcMSK}, f) \\ \mathsf{wcFCT} \leftarrow \mathsf{wcFE.Enc}(\mathsf{wcMPK}, M) \end{array} \right] \geq \frac{1}{2} + \eta - \lambda^{-\omega(1)}.$$

2. **Selective security:** *as in Definition 4.1.*

We now turn to the proof.

Proof. We first prove that the new obfuscator has worst-case approximate correctness, and then prove that it is selectively secure.

Correctness. For any $\lambda, n = n(\lambda)$, message $M \in \{0,1\}^n$, let us denote $\mathcal{X}^M$ a sampler for FE plaintexts of the form $\mathsf{ap}M = (\mathtt{norm}, \mathsf{CT}, \bot, \bot)$ that is defined just like $\mathcal{X}^{\mathbf{0}}$ except that CT is a share of an encryption of M sampled from $\mathsf{DSFE.Enc}(M)$ in the underlying DSFE scheme, rather than a share of an encryption of 0^n.

Then, by the input-hiding guarantee of SFE, and the approximate correctness of apFE, we claim that, for any function $f \in \mathcal{F}$ and corresponding f_{SCT}, decryption in apFE is correct on encryptions of an arbitrary $M \in \{0,1\}^n$ as on encryptions of 0^n. That is, there exists a negligible $\mu(\lambda)$ such that

$$\begin{aligned}
&\Pr\left[\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}}, \mathsf{apFCT}) = f_{\mathsf{SCT}}(\mathsf{ap}M) \;\middle|\; \mathsf{ap}M \leftarrow \mathcal{X}^M\right] \geq \\
&\Pr\left[\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}}, \mathsf{apFCT}) = f_{\mathsf{SCT}}(\mathsf{ap}M) \;\middle|\; \mathsf{ap}M \leftarrow \mathcal{X}^{\mathbf{0}}\right] - \mu \geq \\
&1 - \rho - \mu \ ,
\end{aligned}$$

where in both $(\mathsf{apMPK}, \mathsf{apMSK}) \leftarrow \mathsf{apFE.Setup}(1^\lambda)$,$\mathsf{apFSK}_{f_{\mathsf{SCT}}} \leftarrow \mathsf{apFE.Gen}(\mathsf{apMSK}, f_{\mathsf{SCT}})$, $\mathsf{apFCT} \leftarrow \mathsf{apFE.Enc}(\mathsf{apMPK}, \mathsf{ap}M)$, all defined above in the construction of the exact scheme, and $\mathsf{ap}M = (\mathtt{norm}, \mathsf{CT}, \bot, \bot)$.

We now consider alternative samplers $\left\{\mathcal{X}_j^M \;\middle|\; j \in [k]\right\}$ that sample $\mathsf{ap}M_j$ just as in the canonical $\mathcal{X}^M$, except that CT is sampled as the the jth share of a DSFE encryption of M (rather than the first). Note that by our assumption that the shares $\mathsf{CT}_1, \dots, \mathsf{CT}_k \leftarrow \mathsf{DSFE.Enc}(M)$ have the same marginal distribution, the samplers $\mathcal{X}^M, \mathcal{X}_1^M, \dots, \mathcal{X}_k^M$ all sample from the same distribution. In particular, they satisfy the above statement regarding the probability of correct decryption, satisfied by $\mathcal{X}^M$.

We shall denote by $\mathcal{X}_j^M|\mathsf{CT}_j$ the corresponding sampler conditioned on $\mathsf{CT} = \mathsf{CT}_j$ for some fixed CT_j. We now consider the joint sampler $(\mathsf{ap}M_1, \dots, \mathsf{ap}M_k) \leftarrow \mathcal{X}_{[k]}^M$ where first shares $(\mathsf{CT}_1, \dots, \mathsf{CT}_k)$ are sampled from $\mathsf{DSFE.Enc}(M)$, and then each $\mathsf{ap}M_j$ is sampled from $\mathcal{X}_j|\mathsf{CT}_j$. Note that this sampler corresponds to the way that encryption is done in our actual scheme wcFE defined above.

Noting that the marginal distribution of each $\mathsf{ap}M_j$ sampled accordingly to $\mathcal{X}_{[k]}^M$ is the same as $\mathcal{X}_j^M$, it follows that the expected number of successful decryptions for a sample from $\mathcal{X}_{[k]}^M$ can be lower bounded as follows

$$\begin{aligned}
&\mathbb{E}\left[|\{j \mid \mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}}, \mathsf{apFCT}_j) = f_{\mathsf{SCT}}(\mathsf{ap}M_j)\}| \;\middle|\; (\mathsf{ap}M_1, \dots, \mathsf{ap}M_k) \leftarrow \mathcal{X}_{[k]}^M\right] = \\
&k \cdot \Pr\left[\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}}, \mathsf{apFCT}_j) = f_{\mathsf{SCT}}(\mathsf{ap}M_j) \;\middle|\; \mathsf{ap}M_j \leftarrow \mathcal{X}^M\right] \geq \\
&k \cdot (1 - \rho - \mu) \ ,
\end{aligned}$$

where $(\mathsf{apMPK}, \mathsf{apMSK}) \leftarrow \mathsf{apFE.Setup}(1^\lambda)$, $\mathsf{apFSK}_{f_{\mathsf{SCT}}} \leftarrow \mathsf{apFE.Gen}(\mathsf{apMSK}, f_{\mathsf{SCT}})$, $\mathsf{apFCT}_j \leftarrow \mathsf{apFE.Enc}(\mathsf{apMPK}, \mathsf{ap}M_j)$.

It follows by averaging that with probability at least $1 - \sqrt{\rho} - \frac{2\mu}{\sqrt{\rho}}$ the number of successful decryptions as defined above is larger than $k \cdot (1 - \sqrt{\rho})$. In particular, (for large enough λ) the fraction of faults is below the threshold $\sqrt{\rho}$ allowing to reconstruct $f_{\mathsf{SCT}}(\mathsf{ap}M)$.

Going to our actual encryption scheme wcFE, we now claim that decryption is correct with probability noticeably larger than half. Concretely,

$$\Pr\left[\mathsf{DSFE.Dec}(\mathsf{R},\widehat{\mathsf{CT}}_1,\ldots,\widehat{\mathsf{CT}}_k)=f(M)\;\middle|\; \begin{array}{c}\mathsf{CT}_1,\ldots,\mathsf{CT}_k,\mathsf{R}\leftarrow\mathsf{DSFE.Enc}(M)\\ \widehat{\mathsf{CT}}_j=\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}},\mathsf{apFCT}_j)\end{array}\right]\geq$$

$$\Pr\left[\mathsf{DSFE.Dec}(\mathsf{R},\widehat{\mathsf{CT}}_1,\ldots,\widehat{\mathsf{CT}}_k)=f(M)\;\middle|\; \begin{array}{c}\mathsf{CT}_1,\ldots,\mathsf{CT}_k,\mathsf{R}\leftarrow\mathsf{DSFE.Enc}(M)\\ \left|\left\{\widehat{\mathsf{CT}}_j=f_{\mathsf{SCT}}(\mathsf{ap}M_j)\right\}\right|\geq\sqrt{\rho}\cdot k\end{array}\right]\cdot$$

$$\Pr\left[\left|\{\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}},\mathsf{apFCT}_j)=f_{\mathsf{SCT}}(\mathsf{ap}M_j)\}\right|\geq\sqrt{\rho}\cdot k \mid \mathsf{CT}_1,\ldots,\mathsf{CT}_k,\mathsf{R}\leftarrow\mathsf{DSFE.Enc}(M)\right]=$$

$$\Pr\left[\mathsf{DSFE.Dec}(\mathsf{R},\widehat{\mathsf{CT}}_1,\ldots,\widehat{\mathsf{CT}}_k)=f(M)\;\middle|\; \begin{array}{c}\mathsf{CT}_1,\ldots,\mathsf{CT}_k,\mathsf{R}\leftarrow\mathsf{DSFE.Enc}(M)\\ \left|\left\{\widehat{\mathsf{CT}}_j=\mathsf{DSFE.Eval}(\mathsf{CT}_j,f)\right\}\right|\geq\sqrt{\rho}\cdot k\end{array}\right]\cdot$$

$$\Pr\left[\left|\{\mathsf{apFE.Dec}(\mathsf{apFSK}_{f_{\mathsf{SCT}}},\mathsf{apFCT}_j)=f_{\mathsf{SCT}}(\mathsf{ap}M_j)\}\right|\geq\sqrt{\rho}\cdot k \;\middle|\; \mathsf{CT}_1,\ldots,\mathsf{CT}_k\leftarrow\mathcal{X}^M_{[k]}\right]\geq$$

$$\left(1-\mu'\right)\cdot\left(1-\sqrt{\rho}-\frac{2\mu}{\sqrt{\rho}}\right)\geq\frac{1}{2}+\eta-\lambda^{-\omega(1)}\ ,$$

where in all of the above $(\mathsf{apMPK},\mathsf{apMSK})\leftarrow\mathsf{apFE.Setup}(1^\lambda)$, $\mathsf{apFSK}_{f_{\mathsf{SCT}}}\leftarrow\mathsf{apFE.Gen}(\mathsf{apMSK},f_{\mathsf{SCT}})$, $\mathsf{ap}M_j=(\mathtt{norm},\mathsf{CT}_j,\perp,\perp)$, $\mathsf{apFCT}_j\leftarrow\mathsf{apFE.Enc}(\mathsf{apMPK},\mathsf{ap}M_j)$, and $\mu'(\cdot)$ is some negligible function (corresponding to the negligible decryption error of DSFE).

This completes the proof of correctness.

Security Analysis. We prove the selective security of wcFE in a sequence of hybrids, showing that any adversary $\mathcal{A}$ cannot tell the case that the challenge is an encryption of M_0 from the case that the challenge is an encryption of M_1, for the corresponding (M_0,M_1) of his choice.

$\mathcal{H}_1$**:** this corresponds to the usual security game where the challenge is an encryption of M_0.

$\mathcal{H}_2$**:** here, when generating a key FSK_f for a function f, and accordingly generating an (approximate) key $\mathsf{apFSK}_{f_{\mathsf{SCT}}}$ for the function SCT, the symmetric ciphertext SCT is not an encryption of $0^{\ell\times k}$ as in the previous hybrid, but rather an encryption of the DSFE evaluation corresponding to the challenge ciphertext. Concretely, consider the generation of the challenge ciphertext FCT^*:

- $\mathsf{FE.Enc}(\mathsf{MPK},M_0)$:
  1. Compute $(\mathsf{CT}^*_1,\ldots,\mathsf{CT}^*_k,\mathsf{R}^*)\leftarrow\mathsf{DSFE.Enc}(M_0)$,
  2. For $j\in[k]$
     - let $\mathsf{ap}M^*_j=(\mathtt{norm},\mathsf{CT}^*_j,\perp,\perp)$
     - generate $\mathsf{apFCT}^*_j\leftarrow\mathsf{apFE.Enc}(\mathsf{apMPK},\mathsf{ap}M^*_j)$.

  Output $\mathsf{FCT}^*=(\mathsf{apFCT}^*_1,\ldots,\mathsf{apFCT}^*_k,\mathsf{R}^*)$.

Then SCT will now encrypt $\widehat{\mathsf{CT}}^*_{f,1},\ldots,\widehat{\mathsf{CT}}^*_{f,k}$, where $\widehat{\mathsf{CT}}^*_{f,j}=\mathsf{DSFE.Eval}(\mathsf{CT}^*_j,f)$.

Indistinguishability from the previous hybrid follows from the semantic-security of the symmetric encryption. (Note that at this point, a corresponding symmetric secret key SK is not present, in all encryptions the symmetric-key slot is set to $\perp$.)

$\mathcal{H}_3$**:** here, we change the generation of the challenge ciphertext so to invoke the trapdoor mode rather than the normal mode. Concretely, for each $j\in[k]$, we generate $\mathsf{ap}M^*_j=(\mathtt{trap},\perp,\perp,\mathsf{SK},j)$, where SK is the symmetric key corresponding SCT.

Indistinguishability from the previous hybrid follows from the security of the underlying scheme apFE. Indeed, at this point, for every function f for which a key $\mathsf{apFSK}_{f_{\mathsf{SCT}}}$ was generated,

$$f_{\mathsf{SCT}}(\mathtt{trap}, \bot, \bot, \mathsf{SK}, j) = f_{\mathsf{SCT}}(\mathtt{norm}, \mathsf{CT}_j, \bot, \bot) = \widehat{\mathsf{CT}}^*_{f,j}$$

.

$\mathcal{H}_4$**:** here, we change how the evaluations $\widehat{\mathsf{CT}}^*_{f,j}$ are generated. Recall that in the previous hybrid $\widehat{\mathsf{CT}}^*_{f,j} = \mathsf{DSFE.Eval}(\mathsf{CT}^*_j, f)$, where CT^*_j was generated as part of $(\mathsf{CT}^*_1, \ldots, \mathsf{CT}^*_k, \mathsf{R}^*) \leftarrow \mathsf{DSFE.Enc}(M_0)$. Now, instead of encrypting M_0 in the latter we encrypt M_1.

Indistinguishability now follows from the residual input privacy of the underlying DSFE, since $f(M_0) = f(M_1)$. (Recall, that this is guaranteed also in the presence of R^*, provided that $\mathsf{CT}^*_1, \ldots, \mathsf{CT}^*_k$ are absent from the adversary's view, which is indeed the case in this hybrid.)

$\mathcal{H}_5$-$\mathcal{H}_8$**:** symmetrically follow the above hybrids in reverse order, until the usual security game where M_1 is encrypted in the challenge.

This completes the security analysis. □

Remark 4.3 (removing the assumption on equally-distributed shares)*.* In the above construction we have assumed that the DSFE shares $\mathsf{CT}_1, \ldots, \mathsf{CT}_k$ have the same marginal distribution (for which we have also exhibited an instantiation in Section 2.1.1). To remove this assumption, we could have an instance of an approximate FE scheme apFE_i for each i with respect to the corresponding distribution on CT_i (whereas in the construction above we relied on one instance of an approximate FE defined with respect to the marginal distribution which was the same for all shares).

The (Almost) Exact Scheme FE: to obtain an (almost) exact scheme from the worst-case approximate scheme we again apply standard "BPP amplification". Namely, we consider N parallel copies of the scheme for a large enough N.

Formally, the scheme FE, for function class $\mathcal{F} = \{\mathcal{F}_\lambda\}$ and message space $\mathcal{M} = \left\{\{0,1\}^{n(\lambda)} : \lambda \in \mathbb{N}\right\}$, consists of the algorithms ($\mathsf{FE.Setup}$, $\mathsf{FE.Gen}$, $\mathsf{FE.Enc}$, $\mathsf{FE.Dec}$) defined as follows:

- $\mathsf{FE.Setup}(1^\lambda)$: let $N = \frac{\omega(n+\lambda)}{\eta^2}$. For $i \in [N]$, generate $(\mathsf{wcMPK}_i, \mathsf{wcMSK}_i) \leftarrow \mathsf{wcFE.Setup}(1^\lambda)$. The public key MPK and secret key MSK are accordingly set to be all of the public keys $\{\mathsf{wcMPK}_i\}_{i\in[N]}$ and secret keys $\{\mathsf{wcMSK}_i\}_{i\in[N]}$.
- $\mathsf{FE.Gen}(\mathsf{MSK}, f)$: For $i \in [N]$, generate $\mathsf{wcFSK}_{f,i} \leftarrow \mathsf{wcFE.Gen}(\mathsf{wcMSK}_i, f)$. The functional key FSK_f will consists of the functional keys $\{\mathsf{wcFSK}_{f,i}\}_{i\in[N]}$.
- $\mathsf{FE.Enc}(\mathsf{MPK}, M)$: For $i \in [N]$, compute $\mathsf{wcFCT}_i \leftarrow \mathsf{wcFE.Enc}(\mathsf{wcMPK}_i, M)$. The ciphertext FCT consists of the ciphertexts $(\mathsf{wcFCT}_1, \ldots, \mathsf{wcFCT}_N)$.
- $\mathsf{FE.Dec}(\mathsf{FSK}_f, \mathsf{FCT})$:
  1. Parse $\mathsf{FSK}_f = \{\mathsf{wcFSK}_{f,i}\}_{i\in[N]}$ and $\mathsf{FCT} = \{\mathsf{wcFCT}_i\}_{i\in[N]}$.
  2. For $i \in [N]$, compute $y_i = \mathsf{wcFE.Dec}(\mathsf{wcFSK}_{f,i}, \mathsf{wcFCT}_i)$.
  3. Output $y = \mathsf{majority}(y_1, \ldots, y_N)$.

Proposition 4.2. FE *is an (almost) perfectly correct selectively-secure functional encryption scheme.*

Proof sketch. We show correctness and security.

Correctness. By a Chernoff bound, for large enough λ, and message $M \in \{0,1\}^n$, the probability that the majority value y among all decrypted $y_1, \ldots, y_N$ is incorrect is bounded by

$$\Pr\left[y \neq f(M)\right] \leq 2^{-\Omega(N \cdot \eta^2(\lambda))} \leq 2^{-n+\lambda} \ .$$

The required correctness follows by a union bound over all messages in $\{0,1\}^n$.

Security. The scheme consists of N independent copies of the worst-case scheme that is selectively secure. Security thus follows by a standard hybrid argument. □

4.3 Instantiating the Scheme

As discussed in Section 2.1.1, we can instantiate the DSFE based an information-theoretic variant of BGW for $\mathbf{NC}^1$, resulting in an FE scheme for $\mathbf{NC}^1$. The scheme can then be generically bootstrapped to $\mathbf{P/poly}$ assuming weak PRFs in $\mathbf{NC}^1$ [ABSV14].

We can thus state the following theorem

Theorem 4.4. *Assuming approximate FE for* $\mathbf{P/poly}$ *and weak PRFs in* $\mathbf{NC}^1$*, there exists (almost) perfectly correct FE for* $\mathbf{P/poly}$*.*

5 An Alternative Transformation for IO based on FE

Recall that the transformation from (subexponential) approximate IO to (almost) exact IO described in Section 3.2 required SFE with function hiding against malicious receivers, and was instantiated based on (polynomial) DDH and subexponential one-way functions. In this section, we show an alternative transformation based on any subexponential puncturable PRF in $\mathbf{NC}^1$. The transformation is based on a combination of the FE transformation from Section 4 and known results from the literature.

The high-level idea consists of three basic steps:

1. Start with a (subexponentially-secure) approximate IO and implement directly (subexponentially-secure) approximate FE with compact ciphertexts by following a construction from the exact IO setting [GGH+13b].

2. Apply the transformation from approximate FE to obtain exact FE with compact ciphertexts, based on weaker assumptions.

3. Apply a transformation from exact FE to (exact) IO [AJ15, BV15].

Fulfilling this high-level plan requires some care though. The transformation of Garg et al. [GGH+13b] from IO to FE naturally extends to the the approximate setting, but relies on additional assumptions: (exact) public-key encryption and (exact, or rather complete) NIZKs. While these primitives are known based on exact IO [SW14, BP15], they do not work in the approximate setting. Nevertheless, we show how these constructions can be extended to imply the exact versions of the primitives (from approximate IO). A second issue that should be addressed is how the approximate FE to exact FE transformation affects the complexity of FE encryption. Indeed, the transformations of [AJ15, BV15] require certain succinctness properties. We observe that the transformation, when instantiated with the BGW-based DSFE, satisfies the required compactness, when assuming additionally (sub-exponentially-secure) puncturable PRFs in $\mathbf{NC}^1$.

Overall, we prove

Theorem 5.1. *Assuming approximate IO for* $\mathbf{P}/\mathbf{poly}$ *and puncturable PRFs in* $\mathbf{NC}^1$*, both with sub-exponential security there exists (almost) perfectly correct IO* $\mathbf{P}/\mathbf{poly}$.

We next provide further details.

5.1 Approximate FE from Approximate IO

The starting point for this step is the Garg et al. [GGH+13b]. To obtain FE from IO and PKE, and NIZKs, the transformation works as follows. Each encryption has the form (e_0, e_1, π), where e_0, e_1 encrypt a message M under two independent copies of a plain (exact) public-key encryption scheme, and π is a proof that (e_0, e_1) are indeed well-formed using an (exact) NIZK with statistical simulation-soundness.

A functional key for a function f is an obfuscation of a circuit $C_{\mathsf{SK}_0,\mathsf{CRS}}$ that given (e_0, e_1, π):

- verifies the correctness of π with respect to the hardwired common reference string CRS,
- if the proof is accepting, decrypts e_0 using the hardwired secret key SK_0 to obtain M,
- and outputs $f(M)$.

It follows readily that if we replace exact IO in this transformation with approximate IO (say while still using exact PKE and NIZKs) the resulting FE scheme would be approximately-correct. Concretely to get α-correct FE for a message sampler $\mathcal{X}$, we will start with IO that is α-correct for an input sampler $\mathcal{X}'$ that samples FE encryptions (e_0, e_1, π) of random messages M taken from $\mathcal{X}$.

In fact, even if we start with α-correct versions of PKE and NIZKs we would get $\Omega(\alpha)$-correct FE, however, the security of the FE scheme might no longer hold; indeed, the exact correctness of the PKE and NIZK play an important role in the security proof in [GGH+13b]. To fill this gap we will show how to obtain exact NIZK and PKE directly from approximate IO. More accurately, we would obtain almost exactly correct versions where the NIZK and PKE are exactly correct with overwhelming probability over the choice of their public parameters (i.e., the common reference string and public-keys), which is sufficient for the security proof in [GGH+13b].

(Almost) Exact PKE. To obtain (almost) exact PKE, we start with the PKE of Sahai and Waters [SW14] based on exact IO and one-way functions. Here the public key consists of an obfuscation of a circuit C_{K} that give a PRG seed s outputs $\mathsf{PRF}_{\mathsf{K}}(\mathsf{PRG}(s))$ for an appropriately stretching pseudo-random generator and a puncturable PRF. An encryption of M consists of $\mathsf{PRG}(s), M \oplus \mathsf{PRF}_{\mathsf{K}}(\mathsf{PRG}(s))$. Replacing exact IO with α-correct IO in their transformation results in PKE that is correct with probability α over a random encryption of any message M. Faulty encryption schemes such as the latter can be corrected using techniques from the literature [DNR04]. In fact, in our setting, for $\alpha \gg 1/2$ it suffices to perform standard BPP amplification (i.e., invoke polynomially many copies of the scheme and take majority when decrypting).

In the resulting scheme, the probability of decryption error is over the choice of public-key and the randomness used in encryption. Using a technique from [DNR04], we can shift the error probability to the choice of the public-key alone; namely, get a scheme where with overwhelming probability over the choice of keys there are no decryption errors at all. This is done as follows, assume the decryption error is bounded by $2^{-\lambda}$, and encryption uses $r(\lambda) = \lambda^{O(1)}$ bits of randomness. We will now publish together with the public key a random string $R \leftarrow \{0,1\}^r$. Encryption will now be done with randomness $R' = R \oplus \mathsf{PRG}(s)$, where $\mathsf{PRG} : \{0,1\}^{\lambda/2} \rightarrow \{0,1\}^r$ is a pseudo-random generator and $s \leftarrow \{0,1\}^{\lambda/2}$ is a random seed. Due to the sparseness of the PRG with probability $2^{-\Omega(\lambda)}$ over the choice of the keys the are no decryption errors. Semantic-security is maintained due to the security of the PRG.

(Almost) Exact NIZK. Statistical simulation-sound NIZKs can be constructed from any NIZK proof and non-interactive commitment schemes in the common reference string model [GGH+13b]. The same also holds for the case that the NIZK is almost exact (where the resulting SSS NIZK will also be almost exact). The required commitments commitments can be constructed from one-way functions [Nao91]. We now describe how to obtain the required NIZKs from approximate IO.

Concretely, we examine the NIZK construction of Bitansky and Paneth [BP15] based on exact IO and one-way functions. In their construction, IO is used to implement *invariant signatures* [GO92], which are in turn used to implement the *hidden-bit model* [FLS99]. Concretely, a verification key VK in their scheme consists of an obfuscated circuit $C_{\mathsf{CRS},\mathsf{K}}$ that given a message $M \in \{0,1\}^n$, computes $(b, r) \leftarrow \mathsf{PRF}_{\mathsf{K}}(M)$ using a puncturable PRF, and outputs a Naor commitment $\mathsf{C} = \mathsf{COM}_{\mathsf{CRS}}(b, r)$, with respect to common reference string CRS.

Replacing exact IO with α-correct IO preserves two of the guarantees of the invariant signatures: 1) it is invariant in the sense that for every verification key VK and message M, $\mathsf{C} = \mathsf{VK}(M)$ can be opened to a unique bit b, due to the binding of the commitment; 2) it satisfies pseudorandomness of the unique property b, since the obfuscator is as secure as in the exact case. However, now completeness only holds with probability α over random messages M. The implementation of the hidden bit model indeed invokes the invariant signatures for random messages. This leads to a corresponding NIZK with completeness error $(1-\alpha) \cdot \mathrm{poly}(\lambda)$, for some poly that depends on the NIZK construction (and soundness error $2^{-\lambda}$). Assuming $\alpha > 1 - \frac{1}{\lambda \cdot \mathrm{poly}(\lambda)}$, we can then take say λ^2 independent copies, requiring that the prover succeeds only on a single instance, resulting in a NIZK with completeness error $2^{-\lambda}$ and soundness error $\lambda^2 \cdot 2^{-\lambda}$.

In the resulting scheme, the completeness error is over the choice of the common-reference string and the randomness used by the prover. As before we can use the technique from [DNR04], to shift the error probability to the choice of the CRS alone by sparsifying the coins used by the prover using a PRG. This transformation still maintains computational zero-knowledge due to the pseudo-randomness of the PRG, and has the same unconditional soundness.

A caveat of the latter transformation is that it only works if the approximate IO errors on a very small polynomial fraction $1 - \alpha = \lambda^{-\Theta(1)}$ of inputs, and not say a constant. We stress that in the de-idealized constructions of obfuscation [CKP15, PS15, MMN15] the error rate can be made an arbitrary small polynomial. Thus the implication to constructions of IO with an ideal assisting oracle still holds.

5.2 FE to IO

Exact FE vs Almost Exact FE. The transformations of [AJ15, BV15] from FE to IO are naturally described in terms of perfectly correct FE, nevertheless it is easy to verify that they also work starting from FE that is perfectly-correct with overwhelming probability only over the setup phase generating the keys. The resulting IO will be almost perfectly correct.

To almost exact FE given in Section 4 can be turned to one that satisfies the above property using again the randomness sparsification technique of [DNR04] described above.

Succinctness. In the previous subsection, we described how to obtain an approximate FE scheme where the complexity of encryption is independent of the circuit and output size of the corresponding functions, as inherited from the exact scheme of [GGH+13b]. To fulfill our approach we need to make sure that applying our transformation to exact FE still preserves certain succinctness properties required by the transformations in [AJ15, BV15]. Concretely, we note that our approximate to exact FE transformation inherits its succinctness from the underlying DSFE scheme. As discussed in 4.3, using the BGW-based DSFE, incurs a $2^{O(d)}$ overhead in the complexity of encryption, where d is the maximal depth of any circuit in the class, but is otherwise as efficient. Fortunately, Bitansky and Vaikuntanathan [BV15] show that this is still sufficient for a variant of their transformation from FE to IO, under the additional assumption of sub-exponentially-secure puncturable PRFs in $\mathbf{NC}^1$.

6 From Almost Exact to Perfect via NW-PRGs

The transformation presented in Section 3 gives IO where the obfuscated circuit is correct on all inputs with overwhelming probability over the coins of the obfuscator alone (similarly, the transformation in Section 4 gives FE with a negligible decryption error). In this section, we show that, under worst-case complexity assumptions typically used for derandomization, we can turn any such obfuscator to an absolutely exact one.

The transformation at high level. We follow a similar approach to that of Barak, Ong, and Vadhan [BOV07] for showing how to derandomize the first random message in Naor's commitment scheme [Nao91] or in Naor's and Dwork's ZAPs [DN07]. This consists of two main steps:

1. **Step 1:** We translate almost exact IO to "two-message IO", where the first message consists of a random string R, and the second message is the obfuscated circuit, while guaranteeing:
    - With high probability over the string R, the obfuscation is perfectly correct; namely it preserves exact functionality with probability one over the coins of the obfuscator.
    - For any string R, and any circuits of equal size and functionality, their obfuscations are indistinguishable.

   This step is done using the randomness sparsification technique from [Nao91, DNR04] (also mentioned in Section 5), based on a cryptographic PRG.

2. **Step 2:** We derandomize the choice of the message R using a Nissan-Wigderson type PRG against non-deterministic circuits. The final obfuscator will consists of an obfuscation with respect to each possible choice of $R = \mathsf{PRG}_{\mathsf{NW}}(s)$ for $s \in \{0,1\}^{O(\log \lambda)}$.

Before describing the actual construction we now define NW-type PRGs [NW94].

Definition 6.1 (Nondeterministic Circuits)**.** *A nondeterministic boolean circuit $C(x, w)$ takes x as a primary input and w as a witness. We define $C(x) := 1$ if and only if there exists w such that $C(x, w) = 1$.*

Definition 6.2 (NW-Type PRGs against Nondeterministic Circuits)**.** *An efficiently computable function* $\mathsf{PRG}_{\mathsf{NW}} : \{0,1\}^{d(n)} \to \{0,1\}^n$ *is an NW-generator against non-deterministic circuits of size* $\ell(n)$ *if any non-deterministic circuit* C *of size at most* $\ell(n)$ *distinguishes* $U \leftarrow \{0,1\}^{m(n)}$ *from* $\mathsf{PRG}_{\mathsf{NW}}(s)$*, where* $s \leftarrow \{0,1\}^n$*, with advantage at most* $1/\ell(n)$*.*

We shall rely on the following theorem by Shaltiel and Umans regarding the existence NW-type PRGs as above assuming worst-case for non-deterministic circuits.

Theorem 6.3 ([SU01])**.** *Assume there exists a function in* $\mathbf{E} = \mathbf{Dtime}(2^{O(n)})$ *with nondeterministic circuit complexity* $2^{\Omega(n)}$*. Then, for any polynomial* $\ell(\cdot)$*, then there exists an NW-generator* $\mathsf{PRG}_{\mathsf{NW}} : \{0,1\}^{d(n)} \to \{0,1\}^n$ *against non-deterministic circuits of size* $\ell(n)$*, where* $d(n) = O(\log n)$*.*

We remark that the above worst-case assumption can be seen as a natural generalization of the assumption that $\mathbf{EXP} \not\subseteq \mathbf{NP}$. We also note that there is a universal candidate for the corresponding PRG, by instantiating the the hard function with any $\mathbf{E}$-complete language under linear reductions. See further discussion in [BOV07].

We now describe the transformation from almost-perfectly-correct to perfectly-correct IO.

Ingredients. In the following, let λ be a security parameter. We rely on the following primitives:

- An almost-perfectly-correct indistinguishability obfuscator $\mathcal{O}$ for a class of circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}$. Recall that the obfuscator errors with probability at most $2^{-\lambda}$ over its randomness $R^{\mathcal{O}}$, and denote $|R^{\mathcal{O}}| = n(\lambda) = \lambda^{O(1)}$.
- A cryptographic pseudo-random generator $\mathsf{CRPRG} : \{0,1\}^{\lambda/2} \to \{0,1\}^{n(\lambda)}$.
- An NW-generator $\mathsf{NWPRG} : \{0,1\}^{d(\lambda)} \to \{0,1\}^{n(\lambda)}$ against nondeterministic circuits of size $\ell(\lambda)$, where $\ell(\lambda) = \lambda^{O(1)}$ depends on the size of circuits in $\mathcal{C}_\lambda$ and the obfuscator $\mathcal{O}$, the cryptographic PRG, and $d(\lambda) = O(\log \lambda)$. We shall denote $D = 2^d = \lambda^{O(1)}$.

The New Obfuscator:

Given a circuit $C \in \mathcal{C}_\lambda$:

1. Sample cryptographic pseudo-random strings $(R_1^{\mathsf{CR}}, \ldots, R_D^{\mathsf{CR}})$, where $R_i^{\mathsf{CR}} = \mathsf{CRPRG}(s_i)$ and $s_i \leftarrow \{0,1\}^{\lambda/2}$.
2. Compute all NW-pseudo-random strings $(R_1^{\mathsf{NW}}, \ldots, R_D^{\mathsf{NW}})$, where $R_i = \mathsf{NWPRG}(i)$.
3. Output D obfuscations $(\widetilde{C}_1, \ldots, \widetilde{C}_D)$, where $\widetilde{C}_i = \mathcal{O}(C; R_i^{\mathcal{O}})$, where the randomness is computed as $R_i^{\mathcal{O}} = R_i^{\mathsf{NW}} \oplus R_i^{\mathsf{CR}}$.

To evaluate $(\widetilde{C}_1, \ldots, \widetilde{C}_D)$ on x, compute $y_i = \widetilde{C}_i(x)$ and output $y = \mathsf{majority}(y_1, \ldots, y_N)$.

Proposition 6.1. *The above obfuscator is a perfectly-correct indistinguishability obfuscator.*

Proof. The security of the obfuscator follows directly by the pseudo-randomness of the cryptographic PRG and a standard hybrid argument.

We now turn to show perfect correctness. We start by noting that for any $C \in \mathcal{C}_\lambda$ and input x if we sample R^{NW} truly at random from $\{0,1\}^n$, then except with probability $2^{-\lambda/2}$ there exists no $s \in \{0,1\}^{\lambda/2}$ such that $\mathcal{O}(C; R^{\mathcal{O}})(x) \neq C(x)$ for $R^{\mathcal{O}} = R^{\mathsf{NW}} \oplus \mathsf{CRPRG}(s)$. Indeed, for a truly random $R^{\mathcal{O}}$ the obfuscation errors with probability at most $2^{-\lambda}$, and thus the above follows by taking a union bound over all $s \in \{0,1\}^{\lambda/2}$.

We now claim that the same holds, except with probability $\frac{1}{\ell(\lambda)} + 2^{-\lambda/2} \leq \lambda^{-\Omega(1)}$, when $R^{\mathsf{NW}} = \mathsf{NWPRG}(s')$ for $s' \leftarrow \{0,1\}^d$, namely it is pseudo-random and not truly random. Indeed, otherwise we would get a (small) non-deterministic distinguisher for NWPRG. This distinguisher, given R^{NW}, non-deterministically guesses $s \in \{0,1\}^{\lambda/2}$, computes $R^{\mathsf{CR}} = \mathsf{CRPRG}(s)$ and $R^{\mathcal{O}} = R^{\mathsf{NW}} \oplus R^{\mathsf{CR}}$ and checks whether $\mathcal{O}(C; R^{\mathcal{O}})(x) \neq C(x)$. Recall that we've shown that when R^{NW} is truly random such an $s \in \{0,1\}^{\lambda/2}$ exists with probability at most $2^{-\lambda/2}$, whereas, by our assumption above, when R^{NW} is pseudo-random such an s exists with probability at least $\frac{1}{\ell(\lambda)} + 2^{\lambda/2}$.

Note that the size of the above distinguisher is some fixed polynomial ℓ' in λ (and thus also in $n(\lambda)$) that depends only on the size of the circuit C, the polynomial blowup of $\mathcal{O}$, and the running time of the cryptographic PRG. Accordingly, ℓ is chosen such that $\ell(\lambda) > \ell'(\lambda)$.

It follows that for any C and x, and any choice of seeds $s_1, \ldots, s_D \in \{0,1\}^{\lambda/2}$, at most a $\frac{1}{\ell(\lambda)} + 2^{-\lambda/2} < \frac{1}{2}$ fraction of the obfuscations $(\widetilde{C}_1, \ldots, \widetilde{C}_D)$ do not output $C(x)$. Perfect correctness follows. □

Remark 6.4 (Eliminating Decryption Errors). We remark that the same transformation can be used for immunizing encryption against decryption errors. Indeed, Dwork, Naor, and Reingold [DNR04] already show how to use cryptographic PRGs to shift all errors to the choice of the public key (akin to Step 1 above). Using the same derandomization technique as above, once can eliminate errors completely, under similar assumptions. Likewise a similar transformation can be established for FE.

Acknowledgements

We thank Ilan Komargodsky for pointing out [KMN+14, Appendix B].

References

[AB15] Benny Applebaum and Zvika Brakerski. Obfuscating circuits via composite-order graded encoding. In Dodis and Nielsen [DN15], pages 528–556.

[ABSV14] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. The trojan method in functional encryption: From selective to adaptive security, generically. *IACR Cryptology ePrint Archive*, 2014:917, 2014.

[AFK89] Martín Abadi, Joan Feigenbaum, and Joe Kilian. On hiding information from an oracle. *J. Comput. Syst. Sci.*, 39(1):21–50, 1989.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Crypto*, 2015.

[AL11] Gilad Asharov and Yehuda Lindell. A full proof of the BGW protocol for perfectly-secure multiparty computation. *IACR Cryptology ePrint Archive*, 2011:136, 2011.

[BCC+14] Nir Bitansky, Ran Canetti, Henry Cohn, Shafi Goldwasser, Yael Tauman Kalai, Omer Paneth, and Alon Rosen. The impossibility of obfuscation with auxiliary input or a universal simulator. In *CRYPTO*, pages 71–89, 2014.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6, 2012.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519. Springer, 2014.

[BGJ+15] Nir Bitansky, Shafi Goldwasser, Abhishek Jain, Omer Paneth, Vinod Vaikuntanathan, and Brent Waters. Time-lock puzzles from randomized encodings. *IACR Cryptology ePrint Archive*, 2015:514, 2015.

[BGK+14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2014.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10, 1988.

[BOV07] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.

[BP12] Nir Bitansky and Omer Paneth. From the impossibility of obfuscation to a new non-black-box simulation technique. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 223–232, 2012.

[BP15] Nir Bitansky and Omer Paneth. Zaps and non-interactive witness indistinguishability from indistinguishability obfuscation. In Dodis and Nielsen [DN15], pages 401–427.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73. ACM, 1993.

[BR14a] Zvika Brakerski and Guy N. Rothblum. Black-box obfuscation for d-cnfs. In Moni Naor, editor, *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 235–250. ACM, 2014.

[BR14b] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *Theory of Cryptography - 11th*

Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings, volume 8349 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 2014.

[BSW12] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: a new vision for public-key cryptography. *Commun. ACM*, 55(11):56–64, 2012.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *FOCS*, 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT (2)*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300. Springer, 2013.

[BWZ14] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. *IACR Cryptology ePrint Archive*, 2014:930, 2014.

[CGH+15] Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède Lepoint, Hemanta K. Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO*, Lecture Notes in Computer Science. Springer, 2015. To appear.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 3–12. Springer, 2015.

[CKP15] Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On obfuscation with random oracles. In Dodis and Nielsen [DN15], pages 456–467.

[CKV10] Kai-Min Chung, Yael Tauman Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 483–501. Springer, 2010.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *CRYPTO, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493. Springer, 2013.

[CLT15] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO*, Lecture Notes in Computer Science. Springer, 2015. To appear.

[CLTV15] Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *TCC*, 2015.

[DN07] Cynthia Dwork and Moni Naor. Zaps and their applications. *SIAM J. Comput.*, 36(6):1513–1543, 2007.

[DN15] Yevgeniy Dodis and Jesper Buus Nielsen, editors. *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, volume 9015 of *Lecture Notes in Computer Science*. Springer, 2015.

[DNR04] Cynthia Dwork, Moni Naor, and Omer Reingold. Immunizing encryption schemes from decryption errors. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 342–360. Springer, 2004.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM J. Comput.*, 29(1):1–28, 1999.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GGH12] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices and applications. *IACR Cryptology ePrint Archive*, 2012:610, 2012.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, Mariana Raikova, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 498–527. Springer, 2015.

[GGHZ14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. *IACR Cryptology ePrint Archive*, 2014:666, 2014.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GK05] Shafi Goldwasser and Yael Tauman Kalai. On the impossibility of obfuscation with auxiliary input. In *FOCS*, pages 553–562. IEEE Computer Society, 2005.

[GLSW14] Craig Gentry, Allison B. Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. *IACR Cryptology ePrint Archive*, 2014:309, 2014.

[GLW14] Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 426–443. Springer, 2014.

[GO92] Shafi Goldwasser and Rafail Ostrovsky. Invariant signatures and non-interactive zero-knowledge proofs are equivalent (extended abstract). In *CRYPTO*, pages 228–245, 1992.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO*, pages 162–179, August 2012.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HJ15] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. *IACR Cryptology ePrint Archive*, 2015:301, 2015.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *In Proc. 29th ICALP*, pages 244–256, 2002.

[KMN+14] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im)perfect obfuscation. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 374–383. IEEE Computer Society, 2014.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *CCS*, pages 669–684. ACM, 2013.

[LS14] Hyung Tae Lee and Jae Hong Seo. Security analysis of multilinear maps over the integers. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 224–240. Springer, 2014.

[MMN15] Mohammad Mahmoody, Ameer Mohammed, and Soheil Nematihaji. More on impossibility of virtual black-box obfuscation in idealized models. Cryptology ePrint Archive, Report 2015/632, 2015. http://eprint.iacr.org/.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA*, pages 448–457, 2001.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[O'N10] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010.

[PS15] Rafael Pass and Abhi Shelat. Impossibility of VBB obfuscation with ideal constant-degree graded encodings. *IACR Cryptology ePrint Archive*, 2015:383, 2015.

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 500–517. Springer, 2014.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571, 2008.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, 1997.

[SU01] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 648–657, 2001.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *STOC*, pages 475–484. ACM, 2014.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

[Zim15] Joe Zimmerman. How to obfuscate programs directly. In *Eurocrypt*, 2015.