# The Usefulness of Sparsifiable Inputs: How to Avoid Subexponential iO

Thomas Agrikola[1(✉)], Geoffroy Couteau[2], and Dennis Hofheinz[3]

[1] Karlsruhe Institute of Technology, Karlsruhe, Germany
thomas.agrikola@kit.edu
[2] IRIF, Paris-Diderot University, CNRS, Paris, France
couteau@irif.fr
[3] ETH Zurich, Zurich, Switzerland
hofheinz@inf.ethz.ch

**Abstract.** We consider the problem of removing subexponential reductions to indistinguishability obfuscation (iO) in the context of obfuscating probabilistic programs. Specifically, we show how to apply complexity absorption (Zhandry Crypto 2016) to the recent notion of probabilistic indistinguishability obfuscation (piO, Canetti et al. TCC 2015). As a result, we obtain a variant of piO which allows to obfuscate a large class of probabilistic programs, from polynomially secure indistinguishability obfuscation and extremely lossy functions. Particularly, our piO variant is able to obfuscate circuits with specific input domains regardless of the performed computation. We then revisit several (direct or indirect) applications of piO, and obtain

- a fully homomorphic encryption scheme (without circular security assumptions),
- a multi-key fully homomorphic encryption scheme with threshold decryption,
- an encryption scheme secure under arbitrary key-dependent messages,
- a spooky encryption scheme for all circuits,
- a function secret sharing scheme with additive reconstruction for all circuits,

all from polynomially secure iO, extremely lossy functions, and, depending on the scheme, also other (but polynomial and comparatively mild) assumptions. All of these assumptions are implied by polynomially secure iO and the (non-polynomial, but very well-investigated) exponential DDH assumption. Previously, all the above applications required to assume the *subexponential* security of iO (and more standard assumptions).

**Keywords:** Indistinguishability obfuscation · Extremely lossy functions · Subexponential assumptions

T. Agrikola, G. Couteau and D. Hofheinz—Supported by ERC Project PREP-CRYPTO 724307.

Work done while all authors were at Karlsruhe Institute of Technology.


A. Kiayias et al. (Eds.): PKC 2020, LNCS 12110, pp. 187–219, 2020.
https://doi.org/10.1007/978-3-030-45374-9_7

## 1 Introduction

*Obfuscation.* Code obfuscation has been formalized already in the early 2000s as a cryptographic building block, by Hada [42] and Barak et al. [5], along with a number of early positive [23,45,47,56,61] and negative [5,38,61] results. However, prior to the candidate obfuscation scheme of Garg et al. [31], only relatively few positive results on obfuscation were known.

The first candidate obfuscator from [31] changed things. Their work identified indistinguishability obfuscation (iO, cf. [5,39]) as an achievable *and* useful general notion of obfuscation: it presented a candidate indistinguishability obfuscator, along with a first highly non-trivial application. Since then, a vast number of applications have been proposed, ranging from functional [31], deniable [59], and fully homomorphic [25] encryption, over multi-party computation (e.g., [30]), to separation results (e.g., [46]). In the process, powerful techniques like "puncturing" [59] have been discovered, which have found applications even beyond obfuscation (e.g., in multi-party computation [8,36], instantiating the Fiat-Shamir paradigm [24], and verifiable random functions [9,40]). Besides, the notion of iO itself has been refined, and related to other notions of obfuscation [2,10,11,20,25,50], and various different constructions of obfuscators have been presented [3,4,13,53,54,57,63].

*Subexponential Assumptions.* It is currently hard to find a cryptographic primitive that can *not* be constructed from iO (in combination with another mild assumption such as the existence of one-way functions). However, some of the known iO-based constructions come only with *subexponential* reductions to iO. For instance, the only known iO-based constructions of fully homomorphic encryption [25], spooky encryption [27], and graded encoding schemes [29] suffer from reductions with a subexponential loss.

Hence, while iO has generally been recognized as an extremely powerful primitive (even to the extent being called a "central hub" for cryptography [59]), it is not at all clear if this also holds for *polynomially* secure iO. Indeed, it is conceivable that only polynomially secure iO exists, in which case much of iO's power stands in question.

More generally, subexponential reductions (in particular to iO) are undesirable. Namely, the security of existing iO constructions is still not well-understood, and in particular current state-of-the-art constructions of iO schemes (such as [4,53,54]) already require subexponential computational assumptions themselves. Hence, assuming subexponential iO is a particularly risky bet. This suspicion is confirmed in part by [58], who separate polynomial and subexponential security for virtual black-box obfuscation.

Removing subexponential assumptions in general and from iO-based constructions in particular has already explicitly been considered in [35,52] and [33,34,55] respectively. These works offer general techniques and ideas to turn subexponential reductions into polynomial ones. For instance, [34,55] offer ways to replace (subexponential) iO-based constructions with (polynomial) constructions based on functional encryption. Of course, this requires a special structure

of the primitive to be implemented, and is demonstrated for several primitives, including non-interactive key exchange and short signature schemes.

*Our Contribution.* In this work, we are also concerned with substituting subexponential with polynomial reductions in iO-based constructions. Unlike [34,55], however, we do not follow the approach of using functional encryption directly in place of iO, but instead will employ extremely lossy functions (ELFs) [62] to "absorb" subexponential complexity.[1]

We will implement a variant of probabilistic indistinguishability obfuscation (piO, introduced in [25]) using polynomially secure iO (and ELFs). piO schemes can be used to obfuscate *probabilistic* (i.e., randomized) programs, and are currently the only way to obtain, e.g., fully homomorphic encryption (FHE) schemes without circular security assumptions [25]. However, the only previous construction of piO schemes required subexponentially secure iO [25]. Hence, our construction yields the first FHE scheme from polynomially secure iO (and ELFs). Similarly, we can turn the assumption of subexponentially secure iO into polynomially secure iO (plus ELFs) in the construction of spooky encryption from [27].

Both FHE and spooky encryption are quite powerful primitives, and we obtain several "spin-off results" by revisiting their implications. For instance, when instantiating the piO-based FHE construction of [25] with our piO scheme and a suitable public-key encryption scheme, we obtain a fully key-dependent message (KDM) secure public-key encryption scheme from (polynomially secure) iO and the exponentially secure DDH assumption (and no further assumptions). Under the same assumptions, we obtain multi-key FHE with threshold decryption and function secret sharing schemes from the spooky encryption construction from [27].

*On the Plausibility of ELFs.* One could argue that we trade one exponential assumption for another, and it is not clear that assuming polynomial iO and exponential DDH is any better than assuming only subexponential iO in the first place. Seconding Zhandry [62] here, we think that exponential DDH is a realistic assumption that is far more popular, better-investigated, and arguably more plausible than subexponential iO. Much of the currently deployed cryptography relies on (in fact a strong variant of) exponential DDH, because parameters are almost always chosen according to the best known attacks.

*On the Number of Assumptions.* Another natural observation is that iO for general circuits is already an exponential family of assumptions in itself (one for each obfuscated circuit). It might seem that this lets the challenge of relying on polynomially secure iO instead of subexponentially secure iO appear less appealing. We make two comments on that.

[1] That means that our final schemes depend on ELFs, which are currently only known to be instantiable from exponential assumptions. However, we stress that ELFs can be built from exponential variants of very standard assumptions, such as the decisional Diffie-Hellman (DDH) assumption..

– First, being an exponential family of assumptions and assuming resistance against subexponential adversaries are orthogonal issues. Many cryptographic assumptions have several dimensions of strengths, and relaxing the assumption in any of these dimensions is desirable.[2] In this work, we make progress in one important dimension. By replacing subexponential iO by polynomial iO plus exponential DDH, we effectively trade an *exponential* number of subexponential hardness assumptions in exchange for a *single* (plausible, well-studied) exponential hardness assumption (plus an exponential family of polynomial hardness assumptions).
– Second, iO being an exponential family of assumptions can be considered an artificial consequence of working on the general notion of iO for *arbitrary circuits*. When using iO in concrete constructions (e.g. in all the constructions described in this paper), one almost never needs to assume iO for all circuits. It usually suffices to assume iO for a constant number of specific circuits (namely those being obfuscated in the construction and the analysis). Hence, iO is a small number of assumptions when used for building a cryptographic primitive.

### 1.1 Technical Overview

*The piO Construction of Canetti et al.* To describe our ideas, it will be helpful to briefly review the work of Canetti et al. [25]. In a nutshell, they define the notion of piO as a way to obfuscate probabilistic programs, and show how to use piO to implement the first FHE scheme without any circular security assumption. Intuitively, where the notion of iO captures that the obfuscation $\mathsf{iO}(P)$ of a *deterministic* program $P$ does not leak anything beyond the functionality of $P$, piO captures the same for *probabilistic* programs $P$.[3]

They also show how to implement piO with an indistinguishability obfuscator $\mathsf{iO}$ and a pseudorandom function (PRF) $F$. Namely, in order to obfuscate a probabilistic program $P$, Canetti et al. obfuscate the *deterministic* program $P'$ that, on input $x$, runs $P(x)$ with random coins $r = F(K, x)$. Here, $K$ is a PRF key hardcoded into $P'$. The security proof uses "puncturing" techniques [59] and a hybrid argument over all possible $P$-inputs $x$. More specifically, for each $P$-input $x$, separate reductions to the security of $\mathsf{iO}$ and $F$ show that the execution of $P'(x)$ is secure.[4]

This proof strategy is very general and does not need to make any specific assumptions about the structure of $P$. (In fact, this strategy can be viewed

[2] For example, if a protocol relies on the subexponential hardness of LWE with exponential modulus-to-noise ratio, it would be desirable to achieve the same while relying only on polynomially secure LWE, even if the modulus-to-noise ratio remains exponential.

[3] This is of course an oversimplification. Also, [25] define several types of piO security that provide a tradeoff between security and achievability.

[4] Again, we are not very specific about the form of desired or assumed security. However, we believe that for this exposition, these specifics do not matter.

as a specific form of "complexity leveraging", technically similar to the conversion of selective security into adaptive security, e.g., [16].) However, the price to pay is a reduction loss which is linear in the size of the input domain (which usually is exponentially large). In particular, even after scaling security parameters suitably, Canetti et al. still require subexponentially secure iO and PRFs.

*More on Previous Works to Remove Subexponentiality.* There are a number of known ways to deal with subexponential reduction losses due to complexity leveraging (or related techniques). For instance, various semi-generic (pre-iO) techniques seek to achieve adaptive security (for different primitives) by establishing an algebraic or combinatorial structure on the used inputs [17,44,49,60], and can sometimes be adapted to the iO setting [48]. But like the already-mentioned, somewhat more general approaches [34,55], these works make specific assumptions about the structure of the involved computations.

A somewhat more general approach (that works for more general classes of programs) was outlined by Zhandry [62], who introduces the notion of "extremely lossy functions" (ELFs). Intuitively, an ELF is an injective function $G$ that can be switched into an "extremely lossy mode", in which its range is polynomially small. Such an ELF can sometimes be used to "preprocess" inputs in a cryptographic scheme, with the following benefit: a security reduction can switch the ELF to extremely lossy mode, so that only a polynomial number of (preprocessed) inputs $G(x)$ need to be considered. This simplifies a potential hybrid argument over all (preprocessed) inputs $G(x)$, and can lead to a polynomial (instead of a subexponential) reduction.

However, trying to apply this strategy to the construction and reduction of Canetti et al. (as sketched above) directly fails. Namely, in their application, inputs will be inputs $x$ to an arbitrary (probabilistic) program $P$; preprocessing them with an ELF will destroy their structure, and it is not clear how to run $P$ on ELF-preprocessed inputs $G(x)$. Indeed, applying ELFs to realize piO requires fundamentally different techniques.

*Main Idea: piO with Sparsifiable Inputs.* Instead, we will restrict ourselves to programs $P$ that take as input an element $x$ from a small number of (arbitrary but efficiently samplable) distributions. In other words, all possible inputs $x$ need to be in the range of one of a small number of efficient samplers $S_i$. As an example, for $i \in \{0,1\}$, sampler $S_i$ could sample ciphertexts $C$ that encrypt plaintext $i$. Moreover, we require that all inputs to a program $P$ to be obfuscated are at some point actually sampled from some $S_i$ according to a certain process.

Obfuscating a given probabilistic program $P$ (that takes as inputs one or more $x$ as above) now consists of two steps:

1. First, we *encode* all inputs $x$, in the sense that we compile $S_i$ to attach a "certificate" $\mathsf{aux}$ to $x$. This certificate $\mathsf{aux}$ guarantees that $x$ has really been sampled using $S_i$. Furthermore, the compiled sampler $S_i$ uses preprocessed random coins of the form $G(r)$ (instead of $r$) for an ELF $G$. (When $G$ is

in injective mode, this does not affect the distribution of sampled $x$.) The certificate $\mathsf{aux}$ additionally guarantees this choice of random coins.[5]

2. Second, we produce the actual obfuctation of the probabilistic program $P$ as follows. We use an indistinguishability obfuscator $\mathsf{iO}$ to obfuscate the following (deterministic) variant $P'$ of $P$: on inputs $x_1, \ldots, x_\ell$ with certificates $\mathsf{aux}_1, \ldots, \mathsf{aux}_\ell$, $P'$ first checks the certificates $\mathsf{aux}_i$ and aborts if one of them is invalid. Next, $P'$ runs $P(x_1, \ldots, x_\ell)$, with random coins $F(K, (x_i)_{i=1}^{\ell})$ for a PRF $F$ and a hardcoded PRF key $K$. Finally, $P'$ outputs $P$'s output.

Maybe the most important property of this setup is that now the sets of inputs $x_i$ are "sparsifiable" in the following sense. If we set $G$ to extremely lossy mode, then only a polynomial number of different random coins $r$ can occur. Hence, each $S_i$ will output one of only a small number of possible samples (e.g., encryptions $C$ generated with random coins from a small set). In that sense, the set of possible inputs $x_i$ to $P$ has been "sparsified", and a hybrid argument over all possible inputs as in [25] is possible with polynomial loss.

We stress that our technique of applying ELFs fundamentally differs from [62]. In [62], the constructed primitive itself ensures that $G$ is applied on all inputs. When approaching the challenge of constructing piO, however, the input to the primitive must externally be sampled using random coins that are preprocessed with $G$. This process is not under the control of the primitive and therefore requires a mechanism certifying that inputs are generated according to this specific process. We implement this mechanism using the combination of compiling the sampler for the input distribution into a "certifying sampler" (step 1) and restricting correctness of the obfuscated program (step 2).

Surprisingly, our piO scheme achieves the notion of "dynamic-input piO" [25], a very strong variant of piO security. On a high level, dynamic-input piO guarantees indistinguishability between obfuscations of probabilistic programs as long as their output distributions on adversarially chosen inputs are indistinguishable. This constitutes a very strong requirement and, in fact, implies differing-inputs obfuscation [2,5], a notion for which strong impossibility results exist [7,32]. However, our obfuscator produces circuits which are only required to work on inputs certifiably generated according to a specific process. Hence, our piO scheme enjoys a restricted form of correctness. This enables us to circumvent the impossibility results [7,32].

*Applications.* One obvious question is of course how restrictive our assumption on input domains really is. We show that our assumptions apply to two existing piO-based constructions, with a number of interesting consequences.

First, we revisit the piO-based construction of fully homomorphic encryption from [25]. Here, piO is used to obfuscate the FHE evaluation algorithm that takes two ciphertexts (say, of two bit plaintexts $b_0$ and $b_1$) as input, and outputs a ciphertext of the NAND of the two plaintexts (i.e., $b_0 \overline{\wedge} b_1$). If we set $S_b$ to be a sampler that samples an encryption of $b$, this setting perfectly fits our scheme. Hence, we obtain first a leveled homomorphic encryption (LHE) scheme,

[5] Looking ahead, this "certificate" will be implemented using a NIZK in our construction.

and from this an FHE scheme using the high-level strategy from [25]. Hence, putting this together with our piO construction, we obtain an FHE scheme from polynomially secure iO and an ELF (and no further assumptions).

We note that the above FHE scheme is also fully key-dependent message (KDM, see [14]) secure when implemented with a suitable basic public-key encryption scheme (such as the DDH-based scheme of [18]). In that case, the FHE is secure even when an encryption of its own secret key $C_{\mathsf{sk}} = \mathsf{Enc}(\mathsf{pk}, \mathsf{sk})$ is public. However, such an encryption $C_{\mathsf{sk}}$ can be transformed into an encryption $\mathsf{Enc}(\mathsf{pk}, f(\mathsf{sk}))$ of an arbitrary function of $\mathsf{sk}$ thanks to the fully homomorphic properties of the FHE scheme. This leads to a conceptually very simple fully KDM-secure encryption scheme from polynomial assumptions (and ELFs). (We stress that we do not claim novelty for this observation. The connection between FHE and KDM security has already been observed in [6] and [27] have observed that the FHE construction of Canetti et al. preserves interesting properties of the underlying encryption scheme. However, [27] do not explicitly mention KDM security, and we find these consequences interesting enough to point out.)

As our second application, we consider spooky encryption (with CRS) introduced by Dodis et al. [27]. Intuitively, a spooky encryption scheme features a particular type of homomorphism in a multi-key, multi-ciphertext setting. More precisely, given ciphertexts $\{c_i = \mathsf{Enc}(\mathsf{pk}_i, x_i)\}_i$, a spooky encryption scheme allows to produce ciphertexts $\{c'_i\}_i$ with $y_i = \mathsf{Dec}(\mathsf{sk}_i, c'_i)$ such that certain so-called "spooky" relations between between the $x_i$'s and the $y_i$'s hold. An important subclass of spooky relations allows to ensure that the $y_i$'s are random subject to $\sum_i y_i = f(x_1, \ldots, x_n)$, for any polynomial-time computable function $f$. Dodis et al. show that spooky encryption implies (among other things) function secret sharing, and they give a piO-based instantiation of spooky encryption (without the need of a CRS). At the heart of their construction is an obfuscated public "spooky evaluation" algorithm with a hardcoded decryption key. Since this algorithm also takes ciphertexts (and a public key) as input, its input domain can be sparsified much like in the FHE case.

In contrast to the FHE application, however, the spooky encryption application contains more technical subtleties. In particular, some inputs to the "spooky evaluation" algorithm may depend on other inputs, and hence sparsifying inputs needs to proceed in a certain order. The main difficulty here is to find a suitably flexible definition of sparsification; we omit the details in this overview. We note that our results of course also yield all applications of spooky encryption, only from polynomially secure iO (and ELFs). In particular, we obtain a simple protocol for function secret sharing for all functions (with additive reconstruction) from these assumptions [21].

We believe that our new notion of obfuscation will prove useful in other applications; for example, it would likely allow to improve the recent result of [26], which constructed CCA1-secure FHE from subexponentially secure iO.

*Follow-Up Work.* In the recent work [28], Döttling and Nishimaki define the notion universal proxy re-encryption (UPRE). UPRE schemes allow a proxy to convert any ciphertext under any public key of any existing PKE scheme

into a ciphertext under any public key of any possibly different existing PKE scheme. [28] instantiate UPRE based on probabilistic IO due to [25]. UPRE for all PKE schemes (including non re-randomizable ones) requires dynamic-input pIO, which implies differing-inputs obfuscation. However, [28] observe that our notion of doubly-probabilistic IO suffices which yields an instantiation of UPRE for all PKE schemes based on polynomial IO and exponential DDH.

*Organization.* In Sect. 2, we introduce our notations and recall standard preliminaries. Section 3 formally introduces our new variant of piO, called dpiO. Section 4 shows how to instantiate dpiO using polynomially secure iO and ELFs. Eventually, in Sect. 5 and the full version [1] we revisit the construction of leveled homomorphic encryption from [25], using dpiO instead of piO. In the full version [1], we revisit the construction of spooky encryption from [27] using dpiO and analyze our new construction.

## 2 Preliminaries

**Notations.** Throughout this paper, $\lambda$ denotes the security parameter. For a natural number $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\}$. A probabilistic polynomial time algorithm (PPT, also denoted *efficient* algorithm) runs in time polynomial in the (implicit) security parameter $\lambda$. A positive function $f$ is *negligible* if for any polynomial $p$ there exists a bound $B > 0$ such that, for any integer $k \geq B$, $f(k) \leq 1/|p(k)|$. An event depending on $\lambda$ occurs with *overwhelming probability* when its probability is at least $1 - \mathsf{negl}(\lambda)$ for a negligible function $\mathsf{negl}$. Given a finite set $S$, the notation $x \xleftarrow{\$} S$ means a uniformly random assignment of an element of $S$ to the variable $x$. The notation $\mathcal{A}^{\mathcal{O}}$ indicates that the algorithm $\mathcal{A}$ is given oracle access to $\mathcal{O}$. Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \geq 0}$ be a family of sets of (possibly randomized) circuits, where $\mathcal{C}_\lambda$ contains circuits of size $\mathsf{poly}(\lambda)$. A circuit sampler for $\mathcal{C}$ is a distribution ensemble $D = \{D_\lambda\}_{\lambda \geq 0}$, such that $D_\lambda$ ranges over triples $(C_0, C_1, z)$ with $(C_0, C_1) \in \mathcal{C}_\lambda^2$ of identical size and taking inputs of the same length, and $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$. A class of samplers $\mathbf{S}$ is a set of circuit samplers for $\mathcal{C}$.

### 2.1 Indistinguishability Obfuscation for General Samplers

Indistinguishability obfuscation ($\mathsf{iO}$) for general samplers was introduced in [25]. This notion generalizes the classical notion of $\mathsf{iO}$ introduced in [5]. Informally, an $\mathsf{iO}$ scheme for a sampler $D$ allows to obfuscate circuits sampled with $D$ so that, given a sample $(C_0, C_1)$ from $D$, $\mathsf{iO}(C_0) \approx \mathsf{iO}(C_1)$. The standard notion of $\mathsf{iO}$ is recovered by considering samplers over functionally equivalent deterministic circuits of the same size. Stronger notions of obfuscation, denoted $\mathsf{piO}$, can be defined for samplers over *probabilistic* circuits, satisfying various indistinguishability notions. We recall below the general definition of [25] of $\mathsf{piO}$ for a class of samplers (using a different notion of correctness defined in [27]). The original correctness definition states that an efficient adversary given oracle access to

either the original circuit or the obfuscation (with the restriction that no input can be queried twice), can not tell the difference.

**Definition 1 ($\mathsf{piO}$ for a Class of Samplers [25,27]).** *A uniform PPT machine* $\mathsf{piO}$ *is an* indistinguishability obfuscator *for a class of samplers* $\mathbf{S}$ *over a family* $\mathcal{C} = \{C_\lambda\}_{\lambda\geq 0}$ *of possibly randomized circuits if it satisfies the following conditions:*

**Correctness.** *For every security parameter* $\lambda$*, every circuit* $C \in C_\lambda$*, and every input* $x$*, the distributions of* $C(x)$ *over the random coins of* $C$ *and of* $\mathsf{piO}(1^\lambda, C)(x)$ *over the random coins of the obfuscator are identical.*

**$\mu$-Indistinguishability.** *For every sampler* $D = \{D_\lambda\}_{\lambda\geq 0} \in \mathbf{S}$*, and for every non-uniform PPT machine* $\mathcal{A}$*, it holds that*

$$\begin{aligned}&|\Pr[(C_0, C_1, z) \xleftarrow{\$} D_\lambda : \mathcal{A}(C_0, C_1, \mathsf{piO}(1^\lambda, C_0), z) = 1]\\ &-\Pr[(C_0, C_1, z) \xleftarrow{\$} D_\lambda : \mathcal{A}(C_0, C_1, \mathsf{piO}(1^\lambda, C_1), z) = 1]| \leq \mu(\lambda).\end{aligned}$$

We remark that the construction of $\mathsf{piO}$ from [25] satisfies this notion of correctness if instantiated with a perfect puncturable PRF, see Definition 4. Note that this does not extend to multiple evaluations of the obfuscated circuit. Further, note that this notion of correctness implies that the obfuscated circuit respects the support of the original circuit.

To recover the standard notion of $\mathsf{iO}$, we introduce the class $\mathbf{S}^{\mathsf{eq}}$ of samplers for functionally equivalent (possibly randomized) circuits, *i.e.*, samplers over triplets $(C_0, C_1, z)$ such that $|C_0| = |C_1|$, and for any input $x$ and random coin $r$, $C_0(x; r) = C_1(x; r)$. The standard $\mathsf{iO}$ notion is obtained by considering $\mathsf{piO}$ over the subclass $\mathbf{S}^{\mathrm{det}} \subset \mathbf{S}^{\mathsf{eq}}$ of samplers for deterministic functionally equivalent circuits. We denote by $\mathsf{Adv}_{\mathsf{iO}}(\mathcal{A})$ the advantage of a PPT adversary $\mathcal{A}$ in distinguishing between the obfuscation of functionaly equivalent deterministic circuits.

The work of [25] introduced four types of samplers over probabilistic circuits, which define four corresponding variants of $\mathsf{piO}$: dynamic-input $\mathsf{piO}$, worst-case $\mathsf{piO}$, memoryless worst-case $\mathsf{piO}$, and $\mathsf{X\text{-}Ind}$ $\mathsf{piO}$. Informally, a dynamic-input sampler is required to output (possibly randomized) circuits $C_0, C_1$ such that the output of these circuits on a dynamically chosen input is computationally indistinguishable. The corresponding notion, dynamic-input $\mathsf{piO}$, is the strongest notion defined in [25] and a randomized equivalent of the notion of differing-input obfuscation. Therefore, it inherits the implausibility results of differing-input obfuscation for general circuits [7,32]. On the other hand, [25] shows that the weaker notion $\mathsf{X\text{-}Ind}$ $\mathsf{piO}$ can be realized from subexponentially secure $\mathsf{iO}$ (and subexponentially secure one-way functions). Below, we recall the notion of dynamic-input samplers and dynamic-input $\mathsf{piO}$ from [25].

### 2.2 Dynamic-Input Samplers

**Definition 2 (Dynamic-Input Indistinguishable Samplers [25]).** *The class* $\mathbf{S}^{\mathsf{d\text{-}Ind}}$ *of dynamic-input samplers for a circuit family* $\mathcal{C}$ *contains all*

*circuits samplers* $D = \{D_\lambda\}_{\lambda\in\mathbb{N}}$ *for* $\mathcal{C}$ *with the following properties: for every non-uniform PPT* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, the advantage* $\mathsf{Adv}_{\mathsf{d\text{-}Ind}}(\mathcal{A}) := \Pr[\mathsf{Exp\text{-}d\text{-}Ind}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}$ *of* $\mathcal{A}$ *in the experiment* Exp-d-Ind *represented in Fig. 1 is negligible.*

Experiment $\mathsf{Exp\text{-}d\text{-}Ind}_{\mathcal{A}}(\lambda)$

$(C_0, C_1, z) \xleftarrow{\$} D_\lambda$

$(x, \mathsf{st}) \xleftarrow{\$} \mathcal{A}_1(C_0, C_1, z)$ // the challenge input is chosen dynamically

$y \xleftarrow{\$} C_b(x)$ for $b \xleftarrow{\$} \{0,1\}$

$b' \xleftarrow{\$} \mathcal{A}_2(\mathsf{st}, C_0, C_1, z, x, y)$

**return** $b = b'$

**Fig. 1.** Experiment Exp-d-Ind for the indistinguishability property of dynamic-input samplers.

**Definition 3 (dynamic-input piO).** *A uniform PPT machine is a dynamic-input* piO *scheme if it is a* piO *for the class of dynamic-input samplers* $\mathbf{S}^{\mathsf{d\text{-}Ind}}$ *over* $\mathcal{C}$ *that includes all randomized circuits.*

Note that the class $\mathbf{S}^{\mathsf{eq}}$ of samplers for functionally equivalent circuits that we defined previously, is a subclass of $\mathbf{S}^{\mathsf{d\text{-}Ind}}$: any sampler for triples $(C_0, C_1, z)$ where $C_0$ and $C_1$ are functionally equivalent is trivially a dynamic-input sampler.

### 2.3 Puncturable Pseudorandom Function

A pseudorandom function (PRF) originally introduced in [37] is a tuple of PPT algorithms $\mathsf{F} = (\mathsf{F.KeyGen}, \mathsf{F.Eval})$. Let $\mathcal{K}$ denote the key space, $\mathcal{X}$ denote the domain, and $\mathcal{Y}$ denote the range. The key generation algorithm F.KeyGen on input of $1^\lambda$, outputs a random key from $\mathcal{K}$ and the evaluation algorithm F.Eval on input of a key $K$ and $x \in \mathcal{X}$, evaluates the function $F\colon \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. The core property of PRFs is that, on a random choice of key $K$, no probabilistic polynomial-time adversary should be able to distinguish $F(K, \cdot)$ from a truly random function, when given black-box access to it. Puncturable PRFs (pPRFs) have the additional property that some keys can be generated *punctured* at some point, so that they allow to evaluate the PRF at all points except for the punctured point. As observed in [19,22,51], it is possible to construct such punctured keys for the original construction from [37], which can be based on any one-way functions [43].

**Definition 4 (Puncturable Pseudorandom Function [19,22,51]).** *A* puncturable pseudorandom function (pPRF) *with punctured key space* $\mathcal{K}_p$ *is a triple of PPT algorithms* $(\mathsf{F.KeyGen}, \mathsf{F.Punct}, \mathsf{F.Eval})$ *such that*

- $\mathsf{F.KeyGen}(1^\lambda)$ *outputs a random key* $K \in \mathcal{K}$*,*

– $\mathsf{F.Punct}(K, x)$, *on input* $K \in \mathcal{K}$, $x \in \mathcal{X}$, *outputs a punctured key* $K\{x\} \in \mathcal{K}_p$,
– $\mathsf{F.Eval}(K', x')$, *on input a key* $K'$ *(punctured or not), and a point* $x'$, *outputs an evaluation of the PRF.*

*We require* $\mathsf{F}$ *to meet the following conditions:*

**Functionality Preserved Under Puncturing.** *For all* $\lambda \in \mathbb{N}$, *for all* $x \in \mathcal{X}$,

$$\Pr[K \xleftarrow{\$} \mathsf{F.KeyGen}(1^\lambda), K\{x\} \xleftarrow{\$} \mathsf{F.Punct}(K, x) : \\ \forall x' \in \mathcal{X} \setminus \{x\} : \mathsf{F.Eval}(K, x') = \mathsf{F.Eval}(K\{x\}, x')] = 1.$$

**Pseudorandom at Punctured Points.** *For all PPT adversaries* $\mathcal{A}$,

$$\mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\mathcal{A}) := \Pr[\mathsf{Exp\text{-}s\text{-}pPRF}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}$$

*is negligible, where* $\mathsf{Exp\text{-}s\text{-}cPRF}$ *is represented Fig. 2.*

*We call a pPRF* $\mathsf{F}$ perfect, *if the distribution* $\{\mathsf{F.Eval}(K, x) \mid K \xleftarrow{\$} \mathsf{F.KeyGen}(1^\lambda)\}$ *is identical to the uniform distribution over* $\mathcal{Y}$, *for all inputs* $x \in \mathcal{X}$.[6]

Definition 4 corresponds to a selective security notion for puncturable pseudorandom functions; adaptive security can also be considered, but will not be required in our work. For ease of notation we often write $F(\cdot, \cdot)$ instead of $\mathsf{F.Eval}(\cdot, \cdot)$.

Experiment $\mathsf{Exp\text{-}s\text{-}pPRF}_{\mathcal{A}}(\lambda)$

$(x^*, \mathsf{state}) \xleftarrow{\$} \mathcal{A}(1^\lambda)$

$K \xleftarrow{\$} \mathsf{F.KeyGen}(1^\lambda)$, $K\{x^*\} \xleftarrow{\$} \mathsf{F.Punct}(K, x^*)$

$b \xleftarrow{\$} \{0, 1\}$, $y_0 \leftarrow \mathsf{F.Eval}(K, x^*)$, $y_1 \xleftarrow{\$} \mathcal{Y}$

$b' \xleftarrow{\$} \mathcal{A}(\mathsf{state}, K\{x^*\}, y_b)$

**return** $b = b'$

**Fig. 2.** Selective security game for puncturable pseudorandom functions.

### 2.4 Extremely Lossy Function

In this section we present extremely lossy functions (ELFs) introduced in [62]. ELFs are an extremely powerful primitive for complexity absorption allowing to replace subexponential or even exponential security assumptions with polynomial ones. Informally, an ELF is a function that can be generated in two

[6] Given any pPRF $\mathsf{F}'$, we can build a perfect pPRF $\mathsf{F}$ by sampling two keys $K_1 \xleftarrow{\$} \mathsf{F}'.\mathsf{KeyGen}(1^\lambda)$ and $K_2 \xleftarrow{\$} \mathcal{Y}$ in the key generation algorithm and defining the evaluation algorithm to output $\mathsf{F}'.\mathsf{Eval}(K_1, x) \oplus K_2$ on input of $x$, see [27].

different modes: an injective mode and an extremely lossy mode. In injective mode, the range of the ELF has exponential size whereas the range comprises only polynomially many elements in extremely lossy mode.

**Definition 5 (Extremely Lossy Function [62]).** *An extremely lossy function* $\mathsf{ELF}$ *is an algorithm* $\mathsf{ELF.Gen}$ *which, on input* $(M, r)$*, where* $M$ *is an integer and* $r \in [M]$*, outputs the description of a function* $G\colon [M] \to [N]$ *such that*

- $G$ *can be computed in time* $\mathsf{poly}(\log M)$
- *If* $r = M$*,* $G$ *is injective with overwhelming probability (in* $\log M$*) over the randomness of* $\mathsf{ELF.Gen}(M, M)$*;*
- *For any* $r \in [M]$*,* $|G([M])| < r$ *with overwhelming probability (in* $\log M$*) over the randomness of* $\mathsf{ELF.Gen}(M, r)$*;*
- ***Indistinguishability:*** *For any large enough* $M$*, any polynomial* $P$*, and any inverse polynomial function* $\delta$*, there exists a polynomial* $Q$ *such that for any adversary* $\mathcal{A}$ *running in time at most* $P(\log M)$ *and any* $r \in [Q(\log M), M]$*, the advantage of* $\mathcal{A}$ *in distinguishing* $\mathsf{ELF.Gen}(M, M)$ *from* $\mathsf{ELF.Gen}(M, r)$ *is bounded by* $\delta(\log M)$*.*

In addition, we will consider extremely lossy functions satisfying *strong regularity*, as defined below.

**Definition 6 (Strong regularity).** *An* $\mathsf{ELF}$ *is strongly regular if for any (polynomial)* $r$*, the distribution* $\{x \xleftarrow{\$} [M] : G(x)\}$ *is statistically close to uniform over* $G([M])$*, with overwhelming probability over the choice of* $G \xleftarrow{\$} \mathsf{ELF.Gen}(M, r)$*.*

We note that, if an ELF is strongly regular, it is possible to efficiently enumerate its image: the set of values obtained by evaluating an ELF on $\lambda r \log r$ random inputs, where $r$ is a bound on the size of its image, contains the entire image of the ELF with overwhelming probability.

**Instantiating ELFs.** A construction of strongly regular extremely lossy function is given in [62]. It can be based on the exponential hardness of the decision Diffie-Hellman assumption (or any of its variants, such as the decision linear assumption), which we denote $\mathsf{eDDH}$. The $\mathsf{eDDH}$ assumption for a group generator $\mathsf{GroupGen}$ (which generates a tuple $(\mathbb{G}, p, g)$ where $\mathbb{G}$ is a group, $p$ is its order, and $g$ is a generator of $\mathbb{G}$) states that there exists a polynomial $q$ such that for any time bound $t$ and probability $\varepsilon$, denoting $\kappa \leftarrow \log q(t, 1/\varepsilon)$, any adversary $\mathcal{A}$ running in time at most $t$ has advantage at most $\varepsilon$ in distinguishing the following distributions:

$$\{(\mathbb{G}, p, g) \xleftarrow{\$} \mathsf{GroupGen}(1^\kappa), (a, b, c) \xleftarrow{\$} \mathbb{Z}_p^3 : (\mathbb{G}, g, g^a, g^b, g^c)\},$$
$$\{(\mathbb{G}, p, g) \xleftarrow{\$} \mathsf{GroupGen}(1^\kappa), (a, b) \xleftarrow{\$} \mathbb{Z}_p^2 : (\mathbb{G}, g, g^a, g^b, g^{ab})\}.$$

As noted in [62], groups based on elliptic curves are plausible candidates for groups where this assumption holds: in practical instantiations of DDH over elliptic curves, the size of the group is chosen assuming that the best attack takes

time $O(\sqrt{p})$, hence disproving eDDH (which amounts to showing that there is an attack which takes time less than $p^c$ for any constant $c$) would have considerable practical implications. Furthermore, relying on some form of exponential hardness assumption seems necessary, as a construction from polynomial hardness only would have surprising complexity-theoretic implications. More precisely, given access to only some super-logarithmic amount of non-determinism (i.e. $\omega(\log \log M)$ bits, where $[M]$ is the domain of the ELF), it is easy to distinguish between injective and lossy mode of the ELF. This is due to the fact that in lossy mode, the codomain of $G$ has only polynomial size which means that the restriction of $G$ to the set $D = [2^{\omega(\log \log M)}]$ (having super-polynomial cardinality) is guaranteed to have a collision (which is not the case in injective mode), and using only $\omega(\log \log M)$ bits of non-determinism this collision can be guessed.

### 2.5 Non-interactive Zero-Knowledge Proof System

A non-interactive zero-knowledge (NIZK) proof system for a language $L$ with witness relation $R$ enables to prove in a non-interactive manner that some statements are in $L$ without leaking information about corresponding witnesses. NIZK proof systems were originally introduced in [15].

**Definition 7 (Non-interactive zero-knowledge proof system [41]).** *A* non-interactive zero-knowledge (NIZK) proof system *for a language* $L \in \mathsf{NP}$ *(with witness relation* $R$*) is a tuple of PPT algorithms* $\mathsf{NIZK} = (\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ *such that* NIZK.Setup *is a common reference string generation algorithm,* NIZK.Prove *is a proving algorithm* NIZK.Verify *is a (deterministic) verification algorithm.*

- $\mathsf{NIZK.Setup}(1^\lambda)$ *outputs a common reference string* crs.
- $\mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$*, on input* crs*, a statement* $x$ *and a witness* $w$*, outputs a proof* $\pi$*.*
- $\mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi)$*, on input* crs*, a statement* $x$ *and a proof* $\pi$*, outputs either* 1 *or* 0*.*

*We require* NIZK *to meet the following properties:*

**Perfect Completeness.** *For every* $(x, w) \in R$*, we have that*

$$\Pr[\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Setup}(1^\lambda),\ \pi \xleftarrow{\$} \mathsf{NIZK.Prove}(\mathsf{crs}, x, w) : \mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi) = 1] = 1.$$

**Statistical Soundness.** *For every* $x \notin L$ *with* $|x| = \lambda$ *and every (possibly unbounded) adversary* $\mathcal{A}$*, we have that*

$$\Pr[\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Setup}(1^\lambda), \pi \xleftarrow{\$} \mathcal{A}(\mathsf{crs}, x) : \mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi) = 1] < 2^{-\lambda}.$$

**Computational Zero-Knowledge.** *There exists a PPT algorithm* $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ *such that for every PPT adversary* $\mathcal{A}$*,*

$$\mathsf{Adv}_{ZK}(\mathcal{A}) := |\Pr\left[\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Setup}(1^\lambda) \colon \mathcal{A}^{\mathsf{NIZK.Prove}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1\right] - \Pr\left[(\mathsf{crs}, \tau) \xleftarrow{\$} \mathsf{Sim}_0(1^\lambda) \colon \mathcal{A}^{\mathsf{Sim}_1'(\mathsf{crs},\tau,\cdot,\cdot)}(\mathsf{crs}) = 1\right]|$$

*is negligible in* $\lambda$*, where* $\mathsf{Sim}_1'(\mathsf{crs}, \tau, x, w)$ *returns* $\mathsf{Sim}_1'(\mathsf{crs}, \tau, x)$ *only if* $(x, w) \in R$*.*

For simplicity in the analysis we use a NIZK proof system that satisfies the following property: with overwhelming probability over the coins of $\mathsf{NIZK.Setup}(1^\lambda)$, there does not exist any pair $(x, \pi)$ such that $x \notin L$ and $\mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi) = 1$. We call a NIZK that satisfies this property *almost perfectly sound*. We note that there is a simple folklore method which allows to construct an almost perfectly sound NIZK proof system starting from any statistically sound NIZK proof system. Consider a $2^{-\lambda}$-statistically sound NIZK proof system, for statements $x \in \{0,1\}^n$, for some polynomial $n = n(\lambda)$. Using parallel repetitions, the soundness of the proof system can be amplified to $2^{-\lambda-n}$.[7] Then, it necessarily holds that for all possible $\mathsf{crs}$ except a $2^{-\lambda}$ fraction of them, there does not exist any pair $(x, \pi)$ where $x \notin L$ and $\pi$ is an accepting proof. To realize this, let $E_x^{\mathsf{crs}}$ denote the event that there exists a proof $\pi$ such that $\mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi) = 1$. Then, by a union bound argument, $\Pr_{\mathsf{crs}}[\exists x \in \{0,1\}^n \setminus L \colon E_x^{\mathsf{crs}}] \leq \sum_{x \in \{0,1\}^n \setminus L} \Pr_{\mathsf{crs}}[E_x^{\mathsf{crs}}] \leq 2^n \cdot 2^{-\lambda-n}$. Hence, the NIZK proof system obtained via parallel repetitions is almost perfectly sound.

In [12] Bitansky et al. showed that statistically sound NIZK proof systems can be obtained from polynomially secure indistinguishability obfuscation in conjunction with polynomially secure one-way functions.

## 3 Indistinguishability Obfuscation of Probabilistic Circuits over Distributions of Inputs

We first define the notion of a *sampler with input*. A sampler with input is a family of PPT algorithms which, on input $x$, sample from some distribution $\mathcal{D}_x$. This notion is convenient to capture the fact that, in many scenarios, the inputs to an obfuscated (probabilistic) circuit are sampled from some distribution $\mathcal{D}_x$, where $x$ is some private input of a player.

**Definition 8 (Sampler with Input).** *We say that* $\mathcal{SI} = \{\mathcal{SI}_\lambda\}_{\lambda \in \mathbb{N}}$ *is a family of* samplers with input*, with input domain* $\mathcal{I} = \{\mathcal{I}_\lambda\}_{\lambda \in \mathbb{N}}$*, if for any* $\lambda \in \mathbb{N}$*,* $\mathcal{SI}_\lambda$ *is a set of probabilistic algorithms running in polynomial time (in* $1^\lambda$*) with input domain* $\mathcal{I}_\lambda$ *such that for any* $S \in \mathcal{SI}_\lambda$*, and* $x \in \mathcal{I}_\lambda$*,* $S(x)$ *samples from* $\{0,1\}^\lambda$*.*

[7] That is, for any statement $x \notin L$, the probability $\Pr_{\mathsf{crs}}[\exists \pi \colon \mathsf{NIZK.Verify}(\mathsf{crs}, x, \pi) = 1] \leq 2^{-\lambda-n}$.

### 3.1 Doubly-Probabilistic Indistinguishability Obfuscation

Below, we define a variant of indistinguishability obfuscation, that takes into account the fact that in many applications, obfuscated (probabilistic) circuits might only have to be evaluated on inputs coming from specific distributions. This is formalized by defining an encoding procedure for a sampler with input, which additionally produces auxiliary material that an obfuscated circuit can use to verify that its inputs were produced correctly, and by restricting the correctness of the obfuscated circuit to only hold for such well-formed inputs. We also refer to this auxiliary material as "certificate".

However, this approach faces two issues. First, the inputs to an obfuscated circuit might not be sampled "all at once" from a single distribution; rather, they can come from different and independent sources. We capture this behavior by defining *$\ell$-source obfuscation*, to account for the fact that different inputs might have been sampled independently. Second, when inputs are sampled by different parties, there might still be interdependencies which must be accounted for. For example, a party might sample an input (e.g. a public key of an encryption scheme), pass it to a second party, who then samples a second input from a distribution that is parametrized by the first input (e.g. a ciphertext under that public key). We handle this possibility by ordering the $\ell$ inputs to the obfuscated circuit, and by considering a *stateful* sampler with input $S$: when $S$ is used to generate the $i$'th sample $y_i$, it receives in addition to its input a state $\mathsf{stf}(y_1, \ldots, y_{i-1})$, where $\mathsf{stf}$ is some fixed efficiently computable *state function* (which depends on the particular application), and the $y_j$ are outputs sampled by the first $i-1$ sources. The state function captures the fact that a particular application might define an arbitrary communication pattern, and specifies which samples a party should have access to when generating his sample.

Additionally, we admit the possibility that a sampler produces some additional correlated output, that will not serve as input to an obfuscated circuit. Hence, there is no need to "certify" this input using the auxiliary information, and we call this output unauthenticated output. Continuing the use case from above, given a sampler producing some public key, the unauthenticated part of that sampler's output could be a corresponding secret key.

**Definition 9 (Doubly-Probabilistic Indistinguishability Obfuscation ($\mathsf{dpiO}$)).** *Let $\ell$ be an integer. Let $\{\mathsf{stf}_\lambda : (\{0,1\}^\lambda \cup \{\bot\})^{\ell-1} \to \mathcal{T}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of efficiently computable functions. Let $\mathcal{SI} = \{\mathcal{SI}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of samplers with inputs, with input domain $\{\mathcal{T}_\lambda \times \mathcal{I}\}_{\lambda\in\mathbb{N}}$. Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of (probabilistic) circuits, and let $\mathbf{CS}$ be a class of circuit samplers over $\mathcal{C}$. An $\ell$-source $\mathsf{dpiO}$ scheme for $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{CS})$ is a triple of PPT algorithms $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Obfuscate})$ such that*

- $\mathsf{Setup}(1^\lambda)$*, on input the security parameter (in unary), outputs public parameters* $\mathsf{pp}$*;*
- $\mathsf{Encode}(\mathsf{pp}, S)$*, on input the public parameters* $\mathsf{pp}$*, and a sampler with input $S \in \mathcal{SI}_\lambda$, outputs an encoded sampler $S'$;*

– $\mathsf{Obfuscate}(\mathsf{pp}, S, C)$, *on input public parameters* $\mathsf{pp}$, *a sampler with input* $S \in \mathcal{SI}_\lambda$, *and a circuit* $C \in \mathcal{C}_{\ell\lambda}$, *outputs a circuit* $C'$ *of size* $\mathsf{poly}(\lambda, |C|)$. *We call* $C'$ *an* obfuscation of $C$ with respect to $S$.

*We further assume that the outputs of* $S$ *on any input* $(\mathsf{state}, x)$ *is of the form* $(y; y')$ *(looking ahead, we will call* $y$ *the* authenticated output*, and* $y'$ *the* unauthenticated output*). The scheme should satisfy the three properties given below.*

Informally, the first security requirement ensures that, on any (adversarially chosen) input $x$, state $\mathsf{state}$, and sampler with input $S$, the sampler $S'$ obtained by encoding $S$ outputs samples of the form $(y, \mathsf{aux}; y')$ where $(y; y')$ is distributed as an output of $S(\mathsf{state}, x)$, and $\mathsf{aux}$ does not leak any non-trivial information about the inputs. This is formalized by requiring the existence of a simulator that can simulate $\mathsf{aux}$ given only $y$.

**Definition 10 (Simulatability of Encodings).** *An* $\ell$*-source* $\mathsf{dpiO}$ *scheme for* $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{CS})$ *satisfies* simulatability of encodings *if for any large enough* $\lambda$ *and any (stateful) PPT adversary* $\mathcal{A}$*, there exists a PPT simulator* $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ *such that the advantage of* $\mathcal{A}$ *in distinguishing the experiments* $\mathsf{Exp}^{0\text{-}\mathsf{enc}}$ *and* $\mathsf{Exp}^{1\text{-}\mathsf{enc}}$ *represented on Fig. 3 is negligible. We denote by* $\mathsf{Adv}_{\mathsf{enc}}(\mathcal{A})$ *the advantage of* $\mathcal{A}$ *in this experiment.*

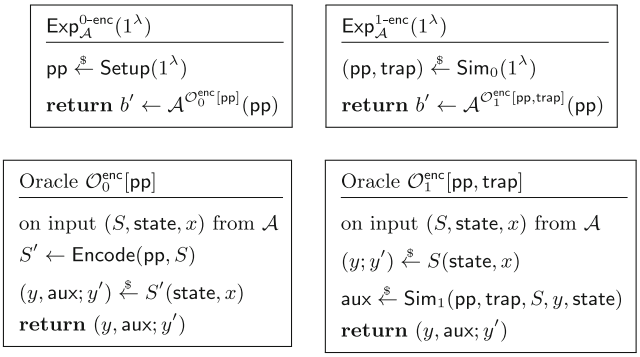


**Fig. 3.** Experiments $\mathsf{Exp}^{0\text{-}\mathsf{enc}}_{\mathcal{A}}(1^\lambda)$ and $\mathsf{Exp}^{1\text{-}\mathsf{enc}}_{\mathcal{A}}(1^\lambda)$ for the simulatability of encodings in an $\ell$-source $\mathsf{dpiO}$. The PPT algorithm $\mathcal{A}$ can interact polynomially many times with either $\mathcal{O}_0^{\mathsf{enc}}[\mathsf{pp}]$ or $\mathcal{O}_1^{\mathsf{enc}}[\mathsf{pp}, \mathsf{trap}]$. $\mathcal{A}$ wins the experiment when it outputs $b' = b$ in $\mathsf{Exp}^{b\text{-}\mathsf{enc}}_{\mathcal{A}}(1^\lambda)$

We now introduce the restricted correctness requirement. Intuitively, it states the following: in an honest scenario, the inputs $(y_1, \ldots, y_\ell)$ should be constructed using the sampler with input $S$. The restricted correctness property guarantees that if the inputs have indeed been constructed "according to $S$", then the obfuscated circuit will behave correctly, and its output distribution (taken over the

coins of the obfuscator) will be (statistically) indistinguishable from the output distribution of the circuit $C$ (taken over its internal random coins). Note that this statistical indistinguishability does not extend to multiple evaluations. Additionally, when evaluated on such inputs, the obfuscated circuit respects the support of the original circuit.

To make this definition meaningful, we need a way to let the obfuscated circuit verify that the inputs are well-formed. Note that we do not want to ensure that they were generated through $S$ with uniformly random coins, but only that they were generated through $S$ with *some* random coins (and some input). To make this verification possible, we let the parties generate their input using the encoded sampler $S'$ instead. This encoded sampler should correctly sample as $S$, but it will in addition produce auxiliary information which can be used by the obfuscated program to verify that the inputs were honestly constructed (more formally, for a given $y$, that there exists an input $x$, coins $r$, and an unauthenticated part $y'$ such that $(y; y') = S(x; r)$).

A small technicality is that we must allow the sampler with input to depend on state information, to capture the possible interdependencies between the inputs. This means that the auxiliary information will have to certify that an input was generated correctly, with respect to some state that the obfuscated circuit might not have access too (which would prevent it from verifying the certificate). However, this issue disappears by restricting the interdependencies to only involve a state computed from the previous *samples* (as opposed to more complex interdependencies which would involve, for example, the coins used to produce these samples). In this case, the obfuscated circuit can check the certificates in an incremental way: it first checks that $y_1$ was correctly constructed with respect to the state $\mathsf{st}_\lambda(\bot, \dots, \bot)$, then it checks that $y_2$ was correctly constructed with respect to the state $\mathsf{st}_\lambda(y_1, \bot, \dots, \bot)$, and so on.

**Definition 11 (Statistical Restricted Correctness).** *An $\ell$-source* dpiO *scheme for* $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{CS})$ *satisfies* restricted correctness *if for any large enough $\lambda \in \mathbb{N}$, any $S \in \mathcal{SI}_\lambda$, $(x_1, \dots, x_\ell) \in \mathcal{I}_\lambda^\ell$, and $C \in \mathcal{C}_{\ell\lambda}$, the advantage of any*

$\mathsf{Exp}^{0\text{-rcorr}}_{\mathcal{A}}(1^\lambda)$

$\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$S' \leftarrow \mathsf{Encode}(\mathsf{pp}, S)$
$C' \xleftarrow{\$} \mathsf{Obfuscate}(\mathsf{pp}, S, C)$
**for** $j \in [\ell]$ **do**
  $\mathsf{state}_j \leftarrow \mathsf{stf}_\lambda(y_1, \dots, y_{j-1}, \bot, \dots)$
  $(y_j, \mathsf{aux}_j; y'_j) \leftarrow S'(\mathsf{state}_j, x_j)$
$z \leftarrow C'(y_1, \mathsf{aux}_1, \dots, y_\ell, \mathsf{aux}_\ell)$
**return** $\mathcal{A}(z)$

$\mathsf{Exp}^{1\text{-rcorr}}_{\mathcal{A}}(1^\lambda)$

**for** $j \in [\ell]$ **do**
  $\mathsf{state}_j \leftarrow \mathsf{stf}_\lambda(y_1, \dots, y_{j-1}, \bot, \dots)$
  $(y_j; y'_j) \leftarrow S(\mathsf{state}_j, x_j)$
$z \leftarrow C(y_1, \dots, y_\ell)$
**return** $\mathcal{A}(z)$

**Fig. 4.** Experiments $\mathsf{Exp}^{0\text{-rcorr}}_{\mathcal{A}}(1^\lambda)$ and $\mathsf{Exp}^{1\text{-rcorr}}_{\mathcal{A}}(1^\lambda)$ for the restricted correctness property an $\ell$-source dpiO. $\mathcal{A}$ wins the experiment when it outputs $b' = b$ in $\mathsf{Exp}^{b\text{-rcorr}}_{\mathcal{A}}(1^\lambda)$ when $b \xleftarrow{\$} \{0, 1\}$.

*(possibly unbounded) adversary* $\mathcal{A}$ *in distinguishing the experiments* $\mathsf{Exp}^{0\text{-rcorr}}$ *and* $\mathsf{Exp}^{1\text{-rcorr}}$ *represented on Fig. 4 is negligible. We denote by* $\mathsf{Adv}_{\mathsf{rcorr}}(\mathcal{A})$ *the advantage of* $\mathcal{A}$ *in this experiment. Additionally, we require that the encoded sampler and the obfuscated circuit respect the support of the original sampler and the original circuit, respectively. That is for all* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ *and all* $S' \leftarrow \mathsf{Encode}(\mathsf{pp}, S)$ *and all* $C' \leftarrow \mathsf{Obfuscate}(\mathsf{pp}, S, C)$*, we have that for all inputs* $(\mathsf{state}, x)$*,* $S'(\mathsf{state}, x) \in \mathsf{Supp}(S(\mathsf{state}, x))$ *and for all* $(y_1, \mathsf{aux}_1, \ldots, y_\ell, \mathsf{aux}_\ell)$ *produced as in* $\mathsf{Exp}^{0\text{-rcorr}}$*,* $C'(y_1, \mathsf{aux}_1, \ldots, y_\ell, \mathsf{aux}_\ell) \in \mathsf{Supp}(C(y_1, \ldots, y_\ell))$*.*

We now introduce the indistinguishability notion. It is close in spirit to the standard indistinguishability notion for obfuscation of probabilistic circuits of [25]. However, in our scenario, the security notion must account for the fact that a set of public parameters pp is generated in a setup phase; the indistinguishability property of obfuscated circuits must therefore hold when (polynomially) many circuits are obfuscated with respect to a single string of public parameters. This suggests an oracle-based security notion.

**Definition 12 (Indistinguishability with Respect to CS).** *An* $\ell$*-source* $\mathsf{dpiO}$ *scheme for* $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{CS})$ *satisfies* indistinguishability with respect to**CS** *if for every circuit sampler* $D = \{D_\lambda\}_{\lambda \in \mathbb{N}} \in \mathbf{CS}$*, for any large enough* $\lambda$*, the advantage of any PPT adversary* $\mathcal{A}$ *in distinguishing the experiments* $\mathsf{Exp}^{0\text{-ind}}$ *and* $\mathsf{Exp}^{1\text{-ind}}$ *represented on Fig. 5 is negligible. We denote by* $\mathsf{Adv}_{\mathsf{ind}}(\mathcal{A})$ *the advantage of* $\mathcal{A}$ *in this experiment.*

$\mathsf{Exp}^{b\text{-ind}}_{\mathcal{A}}(1^\lambda)$

$\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$

**return** $b' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{ind}}_b[\mathsf{pp}, D_\lambda](\cdot)}(\mathsf{pp})$

Oracle $\mathcal{O}^{\mathsf{ind}}_b[\mathsf{pp}, D_\lambda]$

on input $S$ from $\mathcal{A}$

$(C_0, C_1, z) \xleftarrow{\$} D_\lambda$

$C' \xleftarrow{\$} \mathsf{Obfuscate}(\mathsf{pp}, S, C_b)$

**return** $(C_0, C_1, z, C')$

**Fig. 5.** Experiment $\mathsf{Exp}^{b\text{-ind}}_{\mathcal{A}}(1^\lambda)$ for the indistinguishability with respect to **CS** in an $\ell$-source $\mathsf{dpiO}$. The PPT algorithm $\mathcal{A}$ can interact polynomially many times with $\mathcal{O}^{\mathsf{ind}}_b[\mathsf{pp}, D_\lambda]$. The oracle $\mathcal{O}^{\mathsf{ind}}_b[\mathsf{pp}, D_\lambda]$ is stateful and has $(\mathsf{pp}, D_\lambda)$ hardcoded in its description. $\mathcal{A}$ wins the experiment when it outputs $b' = b$ in $\mathsf{Exp}^{b\text{-ind}}_{\mathcal{A}}(1^\lambda)$ when $b \xleftarrow{\$} \{0, 1\}$.

## 4 Construction

In this section, we will construct an $\ell$-source $\mathsf{dpiO}$ scheme (for any constant $\ell$), for samplers with input over an input domain $\mathcal{I}$ of polynomial size[8], and dynamic-input indistinguishable circuit-samplers. Our construction relies on polynomially-secure indistinguishability obfuscation, a perfect puncturable pseudorandom function, an almost perfectly sound non-interactive zero-knowledge proof system, and an extremely lossy function.

[8] We note that the output domain of such samplers can be of exponential size.

### 4.1 Overview

We start by providing a high-level overview of our construction. The $\mathsf{Setup}$ procedure generates parameters for the ELF and for the NIZK proof system. To encode a sampler with input $S$, we define the encoded sampler $S'$ as follows: on input $(\mathsf{state}, x; r)$, $S'$ computes $(y; y') \xleftarrow{\$} S(\mathsf{state}, x; G(r))$ and $\mathsf{aux} \xleftarrow{\$} \mathsf{NIZK.Prove}(y, L^{G,S}_{\mathsf{state}}, (y', x, r))$, and outputs $(y, \mathsf{aux}; y')$. Here, $G$ is the ELF defined by the public parameters, and the language $L^{G,S}_{\mathsf{state}}$ contains all values $y$ for which there exists $(y', x, r)$ such that $(y; y') = S(\mathsf{state}, x, G(r))$. We call *valid input* a value $y \in L^{G,S}_{\mathsf{state}}$. Note that when $G$ is in injective mode, $L^{G,S}_{\mathsf{state}}$ will in general be a trivial language. The simulatability of the encodings directly follows from the injectivity of $G$, and the zero-knowledge property of the proof system.

We construct the $\mathsf{Obfuscate}$ algorithm for a circuit $C$ as follows (we assume a single source in this overview for simplicity). It first samples a pPRF key $K$ for the pPRF $\mathsf{F}$. Then, it returns an obfuscation of the following circuit: on input $(y, \mathsf{aux})$, run $\mathsf{NIZK.Verify}$ on $\mathsf{aux}$ to check that $y$ is a valid input (and output $\perp$ otherwise). Set $r \leftarrow F(K, y)$, and output $C(y; r)$. Restricted correctness follows from the correctness of the NIZK scheme. For indistinguishability between obfuscations of two dynamic-input indistinguishable circuits $(C_0, C_1)$, we follow the standard puncturing strategy of [25]: we proceed through a sequence of hybrids, with successive modifications of the obfuscated circuit. For every possible input $y$, we construct a sequence of hybrids where the outputs $C_0(y; r)$ are gradually replaced by $C_1(y; r)$. Each replacement relies on the security of the $\mathsf{iO}$ scheme, the PRF security, and the dynamic-input indistinguishability of $C_0$ and $C_1$.

The main issue of this approach is that the number of possible inputs $y$ (hence the number of hybrids) is exponential – indeed, this is the reason why the $\mathsf{piO}$ scheme of [25] requires subexponentially secure primitives ($\mathsf{iO}$ and PRF). To get around this issue, we first switch $G$ to an appropriate extremely lossy mode, that the adversary cannot distinguish from the injective mode. Now, the soundness of the NIZK proof system ensures that all valid inputs $y$ are of the form $S(\mathsf{state}, x; G(r))$ for some $(x, r)$ (omitting $y'$ for simplicity). For a given $\mathsf{state}$, the quantity of such values is bounded by the size of the range of $G$ (which is polynomial), times the size of the input domain $\mathcal{I}$. Therefore, in all applications where the inputs to the obfuscated circuit are sampled using private inputs from a small domain, we can base security on polynomially secure $\mathsf{iO}$.

### 4.2 Construction

For our construction, we employ a perfectly sound NIZK proof system for the following (parametrized) language

$$L^{G,S}_{\mathsf{state}} := \{y \mid \exists (y', x, r) \colon (y; y') = S(\mathsf{state}, x; G(r))\}.$$

Let $\ell \in \mathbb{N}$ be a constant, let $\{\mathsf{stf}_\lambda \colon (\{0,1\}^\lambda \cup \{\perp\})^{\ell-1} \to \mathcal{T}_\lambda\}_\lambda$ be a family of efficiently computable state functions, and let $\mathcal{C} = \{C_\lambda\}_\lambda$ be a family of (randomized) circuits with random space $\{0,1\}^M$ (where $M = M(\lambda)$ is polynomial).

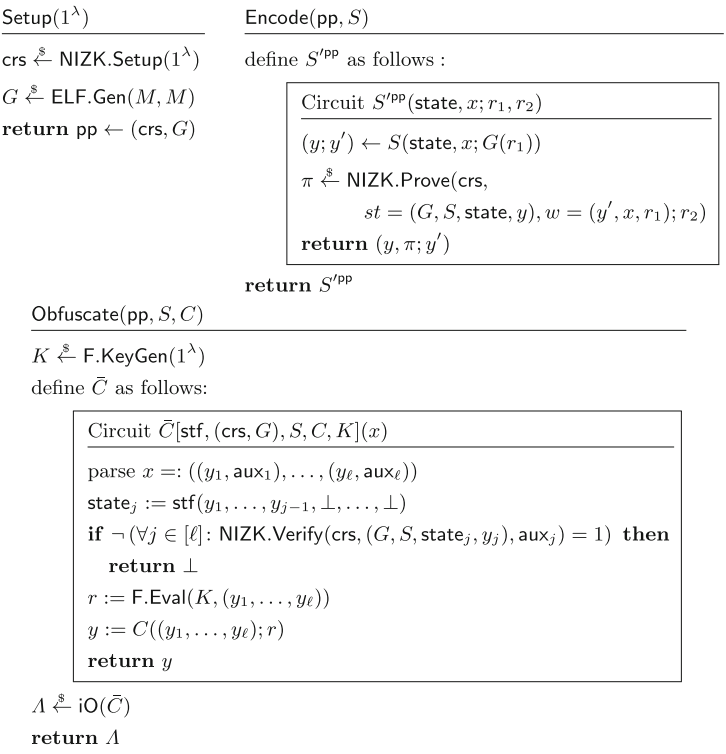


**Fig. 6.** Construction of $\ell$-source dpIO scheme $\mathsf{dpiO} = (\mathsf{Setup}, \mathsf{Encode}, \mathsf{Obfuscate})$.

Let $\mathcal{SI}$ be a family of samplers with input domain $\mathcal{I}$ of polynomial size. Further, let $\mathbf{S}^{\mathsf{d\text{-}Ind}}$ be the class of dynamic-input indistinguishable samplers (over $\mathcal{C}$).

**Theorem 13.** *If* ELF *is a strongly regular extremely lossy function,* iO *is a perfectly correct polynomially secure IO scheme,* F *is a polynomially secure perfect puncturable PRF, and* NIZK *is a perfectly sound polynomially zero-knowledge NIZK proof system for the family of languages* $\{L^{G,S}_{\mathsf{state}}\}_{\mathsf{state},G,S}$*, then* $\mathsf{dpiO} = (\mathsf{Setup}, \mathsf{Encode}, \mathsf{Obfuscate})$ *defined in Fig. 6 is an* $\ell$*-source dpIO scheme for* $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{S}^{\mathsf{d\text{-}Ind}})$.

As noted in Sect. 2.5, almost perfectly correct NIZKs can be constructed from polynomially-secure indistinguishability obfuscation and extremely lossy functions. ELFs also imply the existence of one-way functions, hence of perfect puncturable PRFs [37,43]. Therefore, we get as corollary:

**Corollary 14.** *Assuming polynomially-secure indistinguishability obfuscation and extremely lossy functions, there exists (for any constant* $\ell$*) an* $\ell$*-source doubly-probabilistic indistinguishability obfuscation scheme for the class of dynamic-input circuit-samplers, and input-samplers with a polynomial size input domain.*

*Proof (of Theorem 13).* We prove that $\mathsf{dpiO}$ as defined in Fig. 6 satisfies simulatability of encodings (cf. Definition 10), statistical restricted correctness (cf. Definition 11), and indistinguishability (cf. Definition 12).

**Simulatability of Encodings.** We prove that there exists a PPT simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ such that for every PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathsf{enc}}(\mathcal{A})$ is negligible. By the zero-knowledge property of $\mathsf{NIZK}$, there exists a simulator $(\mathsf{NIZK.Sim}_0, \mathsf{NIZK.Sim}_1)$. We construct a simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ as follows:

- $\mathsf{Sim}_0$ produces the CRS using $(\mathsf{crs}, \tau) \xleftarrow{\$} \mathsf{NIZK.Sim}_0(1^\lambda)$, samples the parameters of the ELF $G$ in injective mode, and outputs $\mathsf{pp} := (\mathsf{crs}, G)$ together with $\mathsf{trap} := \tau$.
- $\mathsf{Sim}_1$ on input $(\mathsf{pp}, \mathsf{trap})$, a sampler $S$, a state $\mathsf{state}$, and a value $y$ sampled via $(y; y') \xleftarrow{\$} S(\mathsf{state}, x)$, $\mathsf{Sim}_1$ produces a simulated proof via $\pi \xleftarrow{\$} \mathsf{NIZK.Sim}_1(\mathsf{crs}, \tau, (G, S, \mathsf{state}, y))$ and outputs $\mathsf{aux} := \pi$.

Let $\mathcal{A}$ be a PPT adversary on the simulatability property of $\mathsf{dpiO}$. We prove indistinguishability between the real and the simulated distribution via a series of hybrids starting from the simulated game $\mathsf{Exp}^{\text{1-enc}}_{\mathcal{A}}(1^\lambda)$.

**Game $\mathbf{G}_0$**: This game is identical to $\mathsf{Exp}^{\text{1-enc}}_{\mathcal{A}}(1^\lambda)$. We remark that in this game, the tuple $(y; y')$ is produced using the adversarially chosen sampler $S$ on input of the adversarially chosen state $\mathsf{state}$ and input $x$ supplied with true randomness.

**Game $\mathbf{G}_1$**: This game is identical to $\mathbf{G}_0$ except for the fact that for each query $(S, \mathsf{state}, x)$, the sampler $S$ is supplied with randomness $G(r)$ for uniform $r$ (instead of true randomness). Due to the strong regularity of $G$ and by a standard hybrid argument over all queries, the statistical distance between $\mathbf{G}_0$ and $\mathbf{G}_1$ is negligible.

**Game $\mathbf{G}_2$**: This game is the same as $\mathbf{G}_1$ with the difference that $\mathsf{crs}$ is produced honestly using $\mathsf{NIZK.Setup}(1^\lambda)$. Additionally, for each adversarial query $(S, \mathsf{state}, x)$, the proof $\pi$ is produced honestly by $\mathsf{NIZK.Prove}(\mathsf{crs}, (G, S, \mathsf{state}, y), (y', x, r))$, where $G(r)$ are the random coins supplied to the sampler $S$. The view of $\mathcal{A}$ in game $\mathbf{G}_2$ is distributed exactly as in the real game $\mathsf{Exp}^{\text{0-enc}}_{\mathcal{A}}(1^\lambda)$.

We construct a PPT adversary $\mathcal{B}$ on the zero-knowledge property of $\mathsf{NIZK}$. Given a CRS $\mathsf{crs}$, $\mathcal{B}$ samples an ELF $G$ in injective mode and invokes $\mathcal{A}$ on input of $\mathsf{pp} := (\mathsf{crs}, G)$. Each time $\mathcal{A}$ queries its oracle on $(S, \mathsf{state}, x)$, $\mathcal{B}$ draws random coins $r$ and invokes the sampler $S$ on input of $(\mathsf{state}, x)$ with random coins $G(r)$ to obtain $(y; y')$. In order to produce $\pi$, $\mathcal{B}$ calls its prove oracle on input $(G, S, \mathsf{state}, y)$ with witness $(y', x, r)$. Therefore, if $\mathcal{B}$ is supplied with an honest CRS and honestly generated proofs, $\mathcal{B}$ perfectly simulates $\mathbf{G}_2$ for $\mathcal{A}$, else $\mathcal{B}$ perfectly simulates $\mathbf{G}_1$. Hence, $|\Pr[\mathsf{out}_2 = 1] - \Pr[\mathsf{out}_3 = 1]| \leq \mathsf{Adv}_{\mathsf{ZK}}(\mathcal{B})$. This concludes the proof.

**Restricted Correctness.** Let $S \in \mathbf{SI}_\lambda$ be an arbitrary sampler with input, let $y_1, \ldots, y_\ell$ be arbitrary values from the input domain $\mathcal{I}_\lambda$, and let $C$ be a circuit from the family $\mathcal{C}_{\ell\lambda}$. To prove the correctness of $\mathsf{dpiO}$, we proceed over a series of hybrids.

**Game $\mathbf{G}_0$**: This game is the ideal game $\mathsf{Exp}_{\mathcal{A}}^{\text{1-rcorr}}(1^\lambda)$. As the sampler $S$ is called using true randomness whereas in $\mathsf{Exp}_{\mathcal{A}}^{\text{0-rcorr}}(1^\lambda)$ samples are generated using $G(r)$, where $r$ is truly random, we need an intermediate hybrid.

**Game $\mathbf{G}_1$**: This game is identical to $\mathbf{G}_0$ with the difference that each call of the sampler $S$ is supplied with $G(r)$ as randomness (where $r$ is sampled uniformly for each call). Due to the strong regularity of $G$, and by a hybrid argument over all calls of $S$, the statistical distance between $\mathbf{G}_0$ and $\mathbf{G}_1$ is negligible.

**Game $\mathbf{G}_2$**: This game is the real game $\mathsf{Exp}_{\mathcal{A}}^{\text{0-rcorr}}(1^\lambda)$.

We now argue that the view of $\mathcal{A}$ in game $\mathbf{G}_1$ is distributed identically to its view in $\mathbf{G}_2$. $\mathbf{G}_2$ samples public parameters $\mathsf{pp}$ via $\mathsf{Setup}(1^\lambda)$ and $S'$ an encoded sampler via $S' \leftarrow \mathsf{Encode}(\mathsf{pp}, S)$. Further, $(y_j, \mathsf{aux}_j)$ are sampled as $\mathsf{state}_j \leftarrow \mathsf{stf}(y_1, \ldots, y_{j-1}, \bot, \ldots, \bot)$ and $(y_j, \mathsf{aux}_j, y'_j) \xleftarrow{\$} S'(\mathsf{state}_j, x_j)$, for $j \in [\ell]$. Let $\Lambda$ be the obfuscation $\Lambda \xleftarrow{\$} \mathsf{Obfuscate}(\mathsf{pp}, S, C)$ of the circuit $C$ with respect to sampler $S$. Due to the perfect correctness of $\mathsf{iO}$, $\Lambda$ has the same functionality as $\bar{C}[\mathsf{stf}, (\mathsf{crs}, G), S, C, K]$, where $K$ is a freshly generated key for the PRF $\mathsf{F}$. Hence, by the perfect completeness of $\mathsf{NIZK}$, on input of $((y_1, \mathsf{aux}_1), \ldots, (y_\ell, \mathsf{aux}_\ell))$, $\Lambda$ evaluates the circuit $C$ on input of $(y_1, \ldots, y_\ell)$ with random coins $F(K, (y_1, \ldots, y_\ell))$. Therefore, the view of $\mathcal{A}$ in the games $\mathbf{G}_1$ and $\mathbf{G}_2$ only differs in the fact that $\mathbf{G}_1$ supplies $C$ with true random coins whereas $\mathbf{G}_2$ supplies $C$ with $F(K, (y_1, \ldots, y_\ell))$ as randomness. As $\mathsf{F}$ is a perfect PRF, the distribution $\{F(K, (y_1, \ldots, y_\ell)) \mid K \xleftarrow{\$} \mathsf{F.KeyGen}(1^\lambda)\}$ is identical to the uniform distribution over the image of $F$. Therefore, the view of $\mathcal{A}$ in $\mathbf{G}_1$ and $\mathbf{G}_2$ is distributed identically.

By construction, all $S' \leftarrow \mathsf{Encode}(\mathsf{pp}, S)$ respect the support of $S$. Furthermore, by construction, perfect completeness of $\mathsf{NIZK}$ and perfect correctness of $\mathsf{iO}$, for all $C' \leftarrow \mathsf{Obfuscate}(\mathsf{pp}, S, C)$ and all $(y_1, \mathsf{aux}_1, \ldots, y_\ell, \mathsf{aux}_\ell)$ produced as in $\mathsf{Exp}^{\text{0-rcorr}}$, $C'(y_1, \mathsf{aux}_1, \ldots, y_\ell, \mathsf{aux}_\ell) \in \mathsf{Supp}(C(y_1, \ldots, y_\ell))$.

**Security.** Let $D \in \mathbf{S}^{\text{d-Ind}}$ be an arbitrary dynamic-input indistinguishable circuit sampler over $\mathcal{C}$. To prove that $\mathsf{dpiO}$ satisfies indistinguishability (Definition 12), we proceed over a series of hybrids. Toward contradiction, assume that there is a PPT adversary $\mathcal{A}$ distinguishing $\mathsf{Exp}_{\mathcal{A}}^{\text{0-ind}}(1^\lambda)$ from $\mathsf{Exp}_{\mathcal{A}}^{\text{1-ind}}(1^\lambda)$ with non-negligible advantage $\varepsilon$ over the random guess after making a polynomial number $Q$ of queries to the oracle.

**Game $\mathbf{G}_0$.** In this game, the challenger samples $b \xleftarrow{\$} \{0, 1\}$, and sets up the experiment $\mathsf{Exp}_{\mathcal{A}}^{b\text{-ind}}(1^\lambda)$. More precisely, $\mathcal{A}$ has access to the public parameters $\mathsf{pp}$ and an oracle $\mathcal{O}_b^{\mathsf{ind}}[\mathsf{pp}, D_\lambda]$, that on input of a sampler with input $S$, draws a

sample $(C_0, C_1, z)$ from $D$ and outputs $(C_0, C_1, z)$ together with an obfuscation $\mathsf{Obfuscate}(\mathsf{pp}, S, C_b)$. $\mathcal{A}$ outputs a guess $b'$ and the challenger returns 1 if $b' = b$. By assumption, $\Pr[\mathsf{out}_0 = 1] = \varepsilon$.

**Game $\mathbf{G}_1$.** In this game, the challenger samples $G$ as $G \xleftarrow{\$} \mathsf{ELF.Gen}(M, t)$, where $t$ is a polynomial such that any PPT algorithm of circuit size $s$ has advantage at most $\varepsilon/2$ in distinguishing $\mathsf{ELF.Gen}(M, M)$ from $\mathsf{ELF.Gen}(M, t)$. The advantage of $\mathcal{A}$ in this game is therefore lower bounded by $\varepsilon/2$: $\Pr[\mathsf{out}_1 = 1] \geq \varepsilon/2$.

**Game $\mathbf{G}'_1$.** This game proceeds exactly as $\mathbf{G}_1$, except that after sampling $b \xleftarrow{\$} \{0,1\}$, the challenger always sets up the experiment $\mathsf{Exp}^{\text{1-ind}}_{\mathcal{A}}(1^\lambda)$. The challenger still returns 1 iff $b' = b$.

By using a standard hybrid argument over the oracle queries, we prove that $|\Pr[\mathsf{out}_1 = 1] - \Pr[\mathsf{out}'_1 = 1]| \leq Q \cdot \mathsf{negl}(\lambda)$, where $Q$ is a polynomial in $\lambda$.

**Game $\mathbf{G}_{1.q}$** This game is identical to $\mathbf{G}_1$ except for the fact that the first $q$ oracle queries are answered using an obfuscation $\Lambda_q$ of $C_1$ instead of $C_b$. Hence, $\Pr[\mathsf{out}_{1.0} = 1] = \Pr[\mathsf{out}_1 = 1]$ and $\Pr[\mathsf{out}_{1.Q} = 1] = \Pr[\mathsf{out}'_1 = 1]$, where $Q$ is the number of adversarial oracle queries.

As $|\Pr[\mathsf{out}_1 = 1] - \Pr[\mathsf{out}'_1 = 1]| \leq \sum_{q=1}^{Q} |\Pr[\mathsf{out}_{1.q} = 1] - \Pr[\mathsf{out}_{1.q+1} = 1]|$, it suffices to upper bound the distinguishing gap between $\mathbf{G}_{1.q}$ and $\mathbf{G}_{1.q+1}$.

We observe that due to the (almost) perfect soundness of $\mathsf{NIZK}$, the obfuscated circuit in the $q$-th oracle answer simulates the randomized computation of the circuit $C_{q,0}$ only on well-formed inputs, i.e. on outputs of $S_q$ using random coins from the range of $G$. As $\mathsf{ELF}$ is in extremely lossy mode, this set of well-formed inputs is *extremely sparsified*. Therefore, by the strong regularity of $\mathsf{ELF}$, we can enumerate over all possible outputs at all input positions $j \in [\ell]$. Let $B_{q,j}$ be the set of all well-formed inputs for input position $j$:

$$B_{q,j} := \{S_q(\mathsf{stf}(y_1, \ldots, y_{j-1}), x; G(r)) \mid x \in \mathcal{I}_\lambda, r \in \{0,1\}^M, y_k \in B_k \text{ for } k \in [j-1]\}.$$

The set $B_{q,j}$ contains at most $|\mathcal{I}| \cdot t^{j-1}$ elements. Further, let $\gamma_{q,1} < \cdots < \gamma_{q,\bar{t}}$ be the ordered enumeration of all $\ell$-tuples in $B_q := \prod_{j=1}^{\ell} B_{q,j}$.[9] Hence, the total number of well-formed inputs $\bar{t} = \prod_{j=1}^{\ell} |B_{q,j}| \leq (|\mathcal{I}| \cdot t^{\ell-1})^\ell \leq |\mathcal{I}|^\ell \cdot t^{(\ell^2)}$ is polynomial in $\lambda$ (given that $\ell$ is a constant, and $|\mathcal{I}|$ and $t$ are polynomial).

Towards proving indistinguishability between $\mathbf{G}_{1.q}$ and $\mathbf{G}_{1.q+1}$, we conduct a hybrid argument over all well-formed inputs for the obfuscation $\Lambda_q$ and gradually replace the evaluation of circuit $C_{q,b}$ with $C_{q,1}$. From here on, our proof strategy is similar to the one employed in [25]. However, we only need to consider polynomially many hybrids (as we assume $|\mathcal{I}|$ to be polynomial), hence we only lose a polynomial factor to the underlying assumptions.

[9] We remark that the values of each set $B_j$ can be computed efficiently by evaluating $S$ on all possible inputs from $\mathcal{I} \times (\prod_{k=1}^{j-1} B_j)$ and all possible images in the range of $G$. Furthermore, it is possible to enumerate the image of $G$ in polynomial time because $G$ is strongly regular.

**Game $\mathbf{G}_{1.q.i}$.** In game $\mathbf{G}_{1.q.i}$ the oracle answers the $q$-th query using an obfuscation of the circuit

$$\bar{C}'[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,b}, C_{q,1}, K_q, \gamma_{q,i}]$$

that is defined in Fig. 7 using $\mathsf{iO}$.

Circuit $\bar{C}'[\mathsf{stf}, (\mathsf{crs}, G), S, C_0, C_1, K, \gamma_i](x)$

parse $x =: ((y_1, \mathsf{aux}_1), \ldots, (y_\ell, \mathsf{aux}_\ell))$
$\mathsf{state}_j := \mathsf{stf}(y_1, \ldots, y_{j-1}, \bot, \ldots, \bot)$
**if** $\neg\,(\forall j \in [\ell] \colon \mathsf{NIZK.Verify}(\mathsf{crs}, (G, S, \mathsf{state}_j, y_j), \mathsf{aux}_j) = 1)$ **then**
  **return** $\bot$
$\gamma := (y_1, \ldots, y_\ell)$
**if** $\gamma < \gamma_i$ **then** $r := F(K, \gamma)$; **return** $C_1(\gamma; r)$
**if** $\gamma = \gamma_i$ **then** $r := F(K, \gamma)$; **return** $C_b(\gamma; r)$
**if** $\gamma > \gamma_i$ **then** $r := F(K, \gamma)$; **return** $C_b(\gamma; r)$

**Fig. 7.** Definition of the circuit $\bar{C}'$.

The circuits $\bar{C}[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,b}, K_q]$ and $\bar{C}'[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,0}, C_{q,1}, K_q, \gamma_{q,1}]$ are functionally equivalent (on input $x = ((y_1, \mathsf{aux}_1), \ldots, (y_\ell, \mathsf{aux}_\ell))$, both return $C_{q,b}(y_1, \ldots, y_\ell)$ with randomness $F(K_q, (y_1, \ldots, y_\ell)))$. Hence, this game hop is justified by the indistinguishability property of $\mathsf{iO}$, more formally there exists a PPT adversary $\mathcal{B}$ such that $|\Pr[\mathsf{out}_{1.q} = 1] - \Pr[\mathsf{out}_{1.q.1}] = 1| \leq \mathsf{Adv}_{\mathsf{iO}}(\mathcal{B})$.

We aim to reduce the game hop from $\mathbf{G}^b_{1.q.i}$ to $\mathbf{G}^b_{1.q.i+1}$ to the dynamic-input indistinguishability of the circuit sampler $D_\lambda$. For this purpose, we first need to supply $C_{q,b}$ with true randomness. Hence, we define an other series of hybrids between $\mathbf{G}_{1.q.i}$ and $\mathbf{G}_{1.q.i+1}$.

**Game $\mathbf{G}_{1.q.i.1}$.** This game is identical to $\mathbf{G}_{1.q.i}$ except for the fact that we use a punctured PRF key $K_q\{\gamma_{q,i}\} \xleftarrow{\$} \mathsf{F.Punct}(K_q, \gamma_{q,i})$ and obfuscate the circuit

$$\bar{C}''[\mathsf{stf}, (\mathsf{crs}, G), C_{q,0}, C_{q,1}, K_q\{\gamma_{q,i}\}, Y := C_{q,b}(\gamma_{q,i}; F(K_q, \gamma_{q,i})), \gamma_{q,i}]$$

defined in Fig. 8 using $\mathsf{iO}$.

As $\mathsf{F}$ preserves the functionality under punctured keys, the circuits $\bar{C}'[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,0}, C_{q,1}, K_q, \gamma_{q,i}]$ and $\bar{C}''[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,0}, C_{q,1}, K_q\{\gamma_{q,i}\}, Y := C_{q,b}(\gamma_{q,i}; F(K_q, \gamma_{q,i})), \gamma_{q,i}]$ are functionally equivalent. Hence, there exists a PPT adversary $\mathcal{B}$ such that $|\Pr[\mathsf{out}_{1.q.i} = 1] - \Pr[\mathsf{out}_{1.q.i.1} = 1]| \leq \mathsf{Adv}_{\mathsf{iO}}(\mathcal{B})$.

We note that the view of $\mathcal{A}$ in game $\mathbf{G}_{1.q.i.1}$ does not depend on the PRF key $K$. This enables to exploit the selective security of $\mathsf{F}$.

Circuit $\bar{C}''[\mathsf{stf}, (\mathsf{crs}, G), S, C_0, C_1, K\{\gamma_i\}, Y, \gamma_i](x)$

parse $x =: ((y_1, \mathsf{aux}_1), \ldots, (y_\ell, \mathsf{aux}_\ell))$
$\mathsf{state}_j := \mathsf{stf}(y_1, \ldots, y_{j-1}, \bot, \ldots, \bot)$
**if** $\neg(\forall j \in [\ell] \colon \mathsf{NIZK.Verify}(\mathsf{crs}, (G, S, \mathsf{state}_j, y_j), \mathsf{aux}_j) = 1)$ **then**
  **return** $\bot$
$\gamma := (y_1, \ldots, y_\ell)$
**if** $\gamma < \gamma_i$ **then** $r := F(K\{\gamma_i\}, \gamma)$; **return** $C_1(\gamma; r)$
**if** $\gamma = \gamma_i$ **then return** $Y$
**if** $\gamma > \gamma_i$ **then** $r := F(K\{\gamma_i\}, \gamma)$; **return** $C_b(\gamma; r)$

**Fig. 8.** Definition of the circuit $\bar{C}''$.

**Game $\mathbf{G}_{1.q.i.2}$.** In this game we replace the randomness $F(K_q, (\gamma_{q,i}))$ by true randomness, i.e. we produce $Y$ as follows: $Y := C_{q,b}(\gamma_{q,i}; R)$. This game hop is justified by the selective PRF property, more formally $|\Pr[\mathsf{out}_{1.q.i.1} = 1] - \Pr[\mathsf{out}_{1.q.i.2} = 1]| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\mathcal{B})$ for some PPT adversary $\mathcal{B}$.

**Game $\mathbf{G}_{1.q.i.3}$.** Game $\mathbf{G}_{1.q.i.3}$ is the same as $\mathbf{G}_{1.q.i.2}$ except for the fact that $Y$ is produced using the circuit $C_{q,1}$, i.e. $Y := C_{q,1}(\gamma_{q,i}; R)$. This game hop is justified by the fact that the circuit sampler $D_\lambda$ is a dynamic-input indistinguishable sampler.

**Game $\mathbf{G}_{1.q.i.4}$.** This game is the same as $\mathbf{G}_{1.q.i.3}$ with the difference that we again use pseudorandom coins to compute $Y$, i.e. $Y := C_{q,1}(\gamma_{q,i}; F(K_q, \gamma_{q,i}))$. For every PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{B}$ such that $|\Pr[\mathsf{out}_{1.q.i.3} = 1] - \Pr[\mathsf{out}_{1.q.i.4} = 1]| \leq \mathsf{Adv}_{\mathsf{s\text{-}cPRF}}(\mathcal{B})$.

As the pPRF $F$ preserves functionality under punctured keys, the two circuits $\bar{C}''[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,0}, C_{q,1}, K_q\{\gamma_{q,i}\}, Y := C_{q,1}(\gamma_{q,i}; F(K_q, \gamma_{q,i})), \gamma_{q,i}]$ and $\bar{C}'[\mathsf{stf}, (\mathsf{crs}, G), S_q, C_{q,0}, C_{q,1}, K_q, \gamma_{q,i+1}]$ are functionally equivalent. Therefore, we have that $|\Pr[\mathsf{out}_{1.q.i.4} = 1] - \Pr[\mathsf{out}_{1.q.i+1} = 1]| \leq \mathsf{Adv}_{\mathsf{iO}}(\mathcal{B})$.

Summing up, the advantage to distinguish $\mathbf{G}_1$ and $\mathbf{G}_{1.Q}$ is bounded by $|\mathcal{I}|^\ell \cdot t^{\ell^2} \cdot \mathsf{negl}(\lambda)$. As $\ell$ is constant and $|\mathcal{I}|, t$ are polynomial, this quantity is negligible. As the circuit obfuscated in $\mathbf{G}_{1.Q}$ is now functionally equivalent to the circuit obfuscated in $\mathbf{G}_1^1$, the game hop to $\mathbf{G}_1'$ is justified by the indistinguishability property of $\mathsf{iO}$. More formally there exists a PPT adversary $\mathcal{B}$ such that $|\Pr[\mathsf{out}_{1.Q} = 1] - \Pr[\mathsf{out}_1'] = 1| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{iO}}(\lambda)$. This implies that the advantage of $\mathcal{A}$ in game $\mathbf{G}_1'$ is lower bounded by $\varepsilon/2 - \mathsf{negl}(\lambda)$, which is non-negligible. However, the view of $\mathcal{A}$ in $\mathbf{G}_1'$ is perfectly independent of $b$, hence its advantage in this game cannot be non-zero; therefore, we reach a contradiction, which concludes the proof. □

### 4.3 Extension

We sketch a straightforward extension of our above construction. It follows easily by inspection that the same proof strategy would work even if the $\ell$ sources,

which sample inputs accorded to an encoding of a sampler $S$ with respect to public parameters $\mathsf{pp}$, are not required anymore to use the *same* public parameters. The $\ell$ sources could even each use different public parameters $(\mathsf{pp}_1, \ldots, \mathsf{pp}_\ell)$. The modified proof for this scenario would proceed by first switching the ELFs in $(\mathsf{pp}_1, \ldots, \mathsf{pp}_\ell)$ to an extremely-lossy mode, through a sequence of $\ell$ hybrids. Each extremely-lossy mode is chosen so that $\mathcal{A}$ as advantage at most $\varepsilon/2\ell$ in distinguishing it from the injective mode. By a union bound, $\mathcal{A}$ has therefore advantage at most $\varepsilon/2$ in distinguishing the all-injective modes from the all-lossy modes. Then, enumerating over all possible valid inputs to an obfuscated circuit takes polynomial time as before, as each input of a source comes from a set of polynomial size. Therefore, the exact same sequence of hybrids proves security, with a polynomial loss in the underlying primitives. To adapt the security properties of our definition of $\mathsf{dpiO}$ to this multi-parameter setting, it suffices to let all experiments initially sample and send to the adversary $\ell$ public parameters $(\mathsf{pp}_1, \ldots, \mathsf{pp}_\ell)$ instead of one. In the simulatability of encodings definition (resp. in the indistinguishability definition), the adversary is allowed to specify under which public parameters it wants to receive a (real or simulated) sample $(y, \mathsf{aux}; y')$ (resp. under which public parameters it wants $C_b$ to be obfuscated in the indistinguishability experiment).

It can prove convenient to simplify the construction in some applications to allow different sources to use different public parameters. Let us illustrate the syntax we adopt on an example: if $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Obfuscate})$ is a 5-source $\mathsf{dpiO}$ scheme, we denote by $\mathsf{Obfuscate}(\mathsf{pp}_1[1-3], \mathsf{pp}_2[4,5],, S, C)$ an obfuscation of a circuit $C$, whose first three inputs should be sampled with respect to $\mathsf{pp}_1$, and whose last two inputs should be sampled with respect to $\mathsf{pp}_2$. We will also sometimes slightly abuse our notation, noting that an $\ell$-source $\mathsf{dpiO}$ scheme directly implies an $i$-source $\mathsf{dpiO}$ scheme for $i \leq \ell$, and allow an $\ell$-source scheme to obfuscate a circuit $C$ that takes $i < \ell$ inputs.

## 5 Leveled Homomorphic Encryption

In this section we show that our notion of dpIO from Sect. 3 can be applied to construct leveled homomorphic encryption in a similar way as in [25]. This construction leads to a transformation which operates on an encryption scheme $E$, satisfying IND-CPA security (and possibly other security properties, e.g., KDM security), and produces a leveled homomorphic encryption scheme that retains the security properties of $E$. We recall the definition of IND-CPA secure encryption schemes in the full version [1].

Let $\mathsf{stf}_\lambda$ be the trivial state function, i.e. $\mathsf{stf}\colon (y_1, y_2) \mapsto \bot$ for each $(y_1, y_2) \in (\{0,1\}^\lambda \cup \{\bot\})^2$. Let $E = (E.\mathsf{KeyGen}, E.\mathsf{Enc}, E.\mathsf{Dec})$ be an $\mathsf{IND\text{-}CPA}$-secure public-key encryption scheme. Let the class $\mathcal{SI}$ contain all samplers $S^{\mathsf{pk}}$ that on input of a state $\mathsf{state}$ and an input $x \in \mathcal{I} := \{0,1\}$, produce an encryption $y := E.\mathsf{Enc}(\mathsf{pk}, x)$ and $y' := \bot$ ignoring $\mathsf{state}$, where $\mathsf{pk}$ is a public key in the range of $E.\mathsf{KeyGen}(1^\lambda)$. Let $\mathcal{C}$ be the class of polynomially sized randomized circuits and let $\mathbf{S}^{\mathsf{d\text{-}Ind}}$ be the class of dynamic-input indistinguishable samplers over $\mathcal{C}$.

**Theorem 15.** *Let* $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Obfuscate})$ *be a* 2*-source* $\mathsf{dpiO}$ *scheme for* $(\mathsf{stf}, \mathcal{SI}, \mathcal{C}, \mathbf{S}^{\mathsf{d\text{-}Ind}})$ *and let* $E$ *be an IND-CPA secure public-key encryption scheme. Then,* $\mathsf{LHE}$ *as defined in Fig. 9 is an IND-CPA secure LHE scheme.*

The proof strategy is similar as in [25].Here we provide an informal sketch of the proof and refer the reader to the full version [1] for the full proof. On a high level, we want to reduce the security of $\mathsf{LHE}$ to the security of the underlying encryption scheme $E$. However, the evaluation key $\mathsf{ek}$ contains information (even though obfuscated) on the secret keys of each level. For the purpose of invoking the security of $E$ on the challenge ciphertext, we need to remove this dependency on $\mathsf{sk}_0$. Therefore, we gradually (starting from level $L$) replace the obfuscations of the circuits $C$ with an obfuscation of *trapdoor* circuits $tC$ that simply output samples produced by the encoded sampler $S'$ on input of 0 (hence, not needing any information on decryption keys). These two circuits only differ in the fact that they sample from the same encoded sampler $S'$ using (possibly) different inputs. Due to the simulatability of encodings and the IND-CPA security of $E$, the two circuits are dynamic-input indistinguishable. Hence, by the indistinguishability property of $\mathsf{dpiO}$ for $\mathbf{S}^{\mathsf{d\text{-}Ind}}$, an honest evaluation key and an evaluation key consisting only of trapdoor circuits are indistinguishable.

$\mathsf{LHE.KeyGen}(1^\lambda, 1^L)$

**for** $i \in \{0, \ldots, L\}$ **do**
  $(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} E.\mathsf{KeyGen}(1^\lambda)$
  $\mathsf{pp}_i \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
  $S'^{\mathsf{pk}_i} \leftarrow \mathsf{Encode}(\mathsf{pp}_i, S^{\mathsf{pk}_i})$
**for** $i \in \{1, \ldots, L\}$ **do**
  $\Lambda_i \xleftarrow{\$} \mathsf{Obfuscate}\left(\mathsf{pp}_{i-1}, S^{\mathsf{pk}_{i-1}}, C[S'^{\mathsf{pk}_i}, \mathsf{sk}_{i-1}]\right)$
$\mathsf{pk} := S'^{\mathsf{pk}_0}$, $\mathsf{sk} := \mathsf{sk}_L$, $\mathsf{ek} := (\Lambda_1, \ldots, \Lambda_L)$
**return** $(\mathsf{pk}, \mathsf{ek}, \mathsf{sk})$

$\mathsf{LHE.Enc}(\mathsf{pk}, m \in \{0,1\})$

parse $\mathsf{pk} =: S'^{\mathsf{pk}_0}$
$(y, \mathsf{aux}, y') \xleftarrow{\$} S'^{\mathsf{pk}_0}(\bot, m)$
**return** $c \leftarrow (y, \mathsf{aux})$

$\mathsf{LHE.Dec}(\mathsf{sk}, c)$

parse $\mathsf{sk} =: \mathsf{sk}_L$
parse $c =: (y, aux)$
**return** $E.\mathsf{Dec}(\mathsf{sk}_L, y)$

$\mathsf{LHE.Eval}(\mathsf{ek}, C, (c_1, \ldots, c_l))$

**for** $i \in \{1, \ldots, L\}$ **do**
  **foreach** gate $g$ on level $i$ **do**
    // let $\alpha_g$, $\beta_g$ denote the respective inputs
    $\gamma_g := \Lambda_i(\alpha_g, \beta_g)$

**Fig. 9.** Description of the LHE scheme $\mathsf{LHE}$. The circuit $C$ is defined in Fig. 10.

$C[S'^{\mathsf{pk}}, \mathsf{sk}'](x_\alpha, x_\beta)$

$\alpha \leftarrow E.\mathsf{Dec}(\mathsf{sk}', x_\alpha)$
$\beta \leftarrow E.\mathsf{Dec}(\mathsf{sk}', x_\beta)$
$(y, \mathsf{aux}, y') \xleftarrow{\$} S'^{\mathsf{pk}}(\bot, \alpha \barwedge \beta)$
**return** $(y, \mathsf{aux})$

$tC[S'^{\mathsf{pk}}](x_\alpha, x_\beta)$

$(y, \mathsf{aux}, y') \xleftarrow{\$} S'^{\mathsf{pk}}(\bot, 0)$
**return** $(y, \mathsf{aux})$

**Fig. 10.** Definition of the circuits $C$ and $tC$.

Given these modifications, the challenge ciphertext $c^*$ consists of an encryption of a bit $b$ under $\mathsf{pk}_0$ accompanied by some auxiliary information produced by the corresponding encoded sampler. This auxiliary information might leak information on the bit $b$ and thereby prevents to directly employ the IND-CPA security of $E$. However, as $\mathsf{dpiO}$ satisfies simulatability of encodings, this auxiliary information can be simulated without knowledge of $b$ and, hence, contains no information about $b$. Therefore, by the IND-CPA security of $E$, $\mathsf{LHE}$ is IND-CPA secure. Given our construction of $\mathsf{dpiO}$ from Sect. 4, we obtain the following corollary:

**Corollary 16.** *Assuming polynomially secure indistinguishability obfuscation and extremely lossy functions, there exists a leveled homomorphic encryption scheme.*

Note that IND-CPA secure cryptosystems, as required in our construction, can be constructed from (polynomially secure) IO and one-way function (the latter being implied by ELFs). Previously, constructions of $\mathsf{LHE}$ were only known from the learning with error assumption, or from *subexponentially secure* indistinguishability obfuscation (together with lossy encryption, which can be based e.g. on DDH). Using the generic transformation from leveled homomorphic encryption to fully homomorphic encryption from [25], we also get:

**Corollary 17.** *Assuming slightly-superpolynomially secure indistinguishability obfuscation and extremely lossy functions, there exists a fully homomorphic encryption scheme.*

Due to space limitations we state here two corollaries concerning FHE and KDM security and refer the reader to the full version [1] for a detailed discussion.

**Corollary 18.** *Assuming polynomially-secure indistinguishability obfuscation and extremely lossy functions, there exists a fully homomorphic encryption scheme.*

**Corollary 19.** *Assuming polynomially-secure indistinguishability obfuscation and $\mathsf{eDDH}$, there exists a fully KDM-secure encryption scheme.*

**Acknowledgments.** We would like to thank the anonymous reviewers for many helpful comments.

## References

1. Agrikola, T., Couteau, G., Hofheinz, D.: The usefulness of sparsifiable inputs: how to avoid subexponential io. Cryptology ePrint Archive, Report 2018/470 (2018). https://eprint.iacr.org/2018/470
2. Ananth, P., Boneh, D., Garg, S., Sahai, A., Zhandry, M.: Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689 (2013). http://eprint.iacr.org/2013/689

3. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 308–326. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_15
4. Ananth, P., Sahai, A.: Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 152–181. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_6
5. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1
6. Barak, B., Haitner, I., Hofheinz, D., Ishai, Y.: Bounded key-dependent message security. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 423–444. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_22
7. Bellare, M., Stepanovs, I., Waters, B.: New negative results on differing-inputs obfuscation. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 792–821. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_28
8. Benhamouda, F., Lin, H.: $k$-round multiparty computation from $k$-round oblivious transfer via garbled interactive circuits. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 500–532. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_17
9. Bitansky, N.: Verifiable random functions from non-interactive witness-indistinguishable proofs. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 567–594. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_19
10. Bitansky, N., Canetti, R., Kalai, Y.T., Paneth, O.: On virtual grey box obfuscation for general circuits. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part II. LNCS, vol. 8617, pp. 108–125. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_7
11. Bitansky, N., Paneth, O.: On the impossibility of approximate obfuscation and applications to resettable cryptography. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC, pp. 241–250. ACM Press, June 2013
12. Bitansky, N., Paneth, O.: ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 401–427. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_16
13. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: Guruswami, V. (ed.) 56th FOCS, pp. 171–190. IEEE Computer Society Press, October 2015
14. Black, J., Rogaway, P., Shrimpton, T.: Encryption-scheme security in the presence of key-dependent messages. In: Nyberg, K., Heys, H. (eds.) SAC 2002. LNCS, vol. 2595, pp. 62–75. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36492-7_6
15. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th ACM STOC, pp. 103–112. ACM Press, May 1988
16. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_14

17. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_27
18. Boneh, D., Halevi, S., Hamburg, M., Ostrovsky, R.: Circular-secure encryption from decision Diffie-Hellman. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 108–125. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_7
19. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part II. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42045-0_15
20. Boyle, E., Chung, K.-M., Pass, R.: On extractability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 52–73. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_3
21. Boyle, E., Gilboa, N., Ishai, Y.: Function secret sharing. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 337–367. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_12
22. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_29
23. Canetti, R.: Towards realizing random oracles: hsh functions that hide all partial information. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 455–469. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052255
24. Canetti, R., Chen, Y., Reyzin, L., Rothblum, R.D.: Fiat-Shamir and correlation intractability from strong KDM-secure encryption. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 91–122. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_4
25. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_19
26. Canetti, R., Raghuraman, S., Richelson, S., Vaikuntanathan, V.: Chosen-ciphertext secure fully homomorphic encryption. In: Fehr, S. (ed.) PKC 2017, Part II. LNCS, vol. 10175, pp. 213–240. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54388-7_8
27. Dodis, Y., Halevi, S., Rothblum, R.D., Wichs, D.: Spooky encryption and its applications. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 93–122. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_4
28. Döttling, N., Nishimaki, R.: Universal proxy re-encryption. Cryptology ePrint Archive, Report 2018/840 (2018). https://eprint.iacr.org/2018/840
29. Farshim, P., Hesse, J., Hofheinz, D., Larraia, E.: Graded encoding schemes from obfuscation. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 371–400. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76581-5_13
30. Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 74–94. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_4
31. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS, pp. 40–49. IEEE Computer Society Press, October 2013

32. Garg, S., Gentry, C., Halevi, S., Wichs, D.: On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 518–535. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_29
33. Garg, S., Pandey, O., Srinivasan, A.: Revisiting the cryptographic hardness of finding a nash equilibrium. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 579–604. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_20
34. Garg, S., Pandey, O., Srinivasan, A., Zhandry, M.: Breaking the sub-exponential barrier in obfustopia. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part III. LNCS, vol. 10212, pp. 156–181. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_6
35. Garg, S., Srinivasan, A.: Single-key to multi-key functional encryption with polynomial loss. In: Hirt, M., Smith, A. (eds.) TCC 2016, Part II. LNCS, vol. 9986, pp. 419–442. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_16
36. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 468–499. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_16
37. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th FOCS, pp. 464–479. IEEE Computer Society Press, October 1984
38. Goldwasser, S., Kalai, Y.T.: On the impossibility of obfuscation with auxiliary input. In: 46th FOCS, pp. 553–562. IEEE Computer Society Press, October 2005
39. Goldwasser, S., Rothblum, G.N.: On best-possible obfuscation. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 194–213. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_11
40. Goyal, R., Hohenberger, S., Koppula, V., Waters, B.: A generic approach to constructing and proving verifiable random functions. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10678, pp. 537–566. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_18
41. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21
42. Hada, S.: Zero-knowledge and code obfuscation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 443–457. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44448-3_34
43. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
44. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_2
45. Hofheinz, D., Malone-Lee, J., Stam, M.: Obfuscation for cryptographic purposes. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 214–232. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_12
46. Hofheinz, D., Rao, V., Wichs, D.: Standard security does not imply indistinguishability under selective opening. In: Hirt, M., Smith, A. (eds.) TCC 2016, Part II. LNCS, vol. 9986, pp. 121–145. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_5

47. Hohenberger, S., Rothblum, G.N., Shelat, A., Vaikuntanathan, V.: Securely obfuscating re-encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 233–252. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_13
48. Hohenberger, S., Sahai, A., Waters, B.: Replacing a random oracle: full domain hash from indistinguishability obfuscation. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 201–220. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_12
49. Hohenberger, S., Waters, B.: Short and stateless signatures from the RSA assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_38
50. Ishai, Y., Pandey, O., Sahai, A.: Public-coin differing-inputs obfuscation and its applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 668–697. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_26
51. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 669–684. ACM Press, November 2013
52. Li, B., Micciancio, D.: Compactness vs collusion resistance in functional encryption. In: Hirt, M., Smith, A. (eds.) TCC 2016, Part II. LNCS, vol. 9986, pp. 443–468. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_17
53. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 599–629. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_20
54. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 630–660. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_21
55. Liu, Q., Zhandry, M.: Decomposable obfuscation: a framework for building applications of obfuscation from polynomial hardness. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 138–169. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_6
56. Lynn, B., Prabhakaran, M., Sahai, A.: Positive results and techniques for obfuscation. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 20–39. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_2
57. Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 500–517. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_28
58. Pass, R., Shelat, A.: Impossibility of VBB obfuscation with ideal constant-degree graded encodings. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016, Part I. LNCS, vol. 9562, pp. 3–17. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49096-9_1
59. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th ACM STOC, pp. 475–484. ACM Press, May/June 2014
60. Waters, B.: Efficient identity-based encryption without random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7

61. Wee, H.: On obfuscating point functions. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC, pp. 523–532. ACM Press, May 2005
62. Zhandry, M.: The magic of ELFs. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 479–508. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_18
63. Zimmerman, J.: How to obfuscate programs directly. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 439–467. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_15