Output-Compressing Randomized Encodings and Applications

Huijia Lin* Rafael Pass† Karn Seth‡ Sidharth Telang§

July 10, 2015

Abstract

We consider *randomized encodings (RE)* that enable encoding a Turing machine Π and input x into its "randomized encoding" $\hat{\Pi}(x)$ in sublinear, or even polylogarithmic, time in the running-time of $\Pi(x)$, *independent of its output length.* We refer to the former as *sublinear RE* and the latter as *compact RE.* For such efficient RE, the standard simulation-based notion of security is impossible, and we thus consider a weaker (distributional) indistinguishability-based notion of security: Roughly speaking, we require indistinguishability of $\hat{\Pi}_0(x_0)$ and $\hat{\Pi}_0(x_1)$ as long as Π_0, x_0 and Π_1, x_1 are sampled from some distributions such that $\Pi_0(x_0), \mathsf{Time}(\Pi_0(x_0))$ and $\Pi_1(x_1), \mathsf{Time}(\Pi_1(x_1))$ are indistinguishable.

We first observe that compact RE is equivalent to a variant of the notion of indistinguishability obfuscation (**iO**)—which we refer to as *puncturable **iO***—for the class of Turing machines *without inputs.* For the case of circuits, puncturable **iO** and **iO** are equivalent (and this fact is implicitly used in the powerful "punctured program" paradigm by Sahai and Waters [SW14]).

We next show the following:

- **Impossibility in the Plain Model:** Assuming the existence of subexponentially secure one-way functions, subexponentially-secure sublinear RE does not exists. (If additionally assuming subexponentially-secure **iO** for circuits we can also rule out polynomially-secure sublinear RE.) As a consequence, we rule out also puncturable **iO** for Turing machines (even those without inputs).
- **Feasibility in the CRS model and Applications to iO for circuits:** Subexponentially-secure sublinear RE in the CRS model and one-way functions imply **iO** for circuits through a simple construction generalizing GGM's PRF construction. Additionally, any compact (even with sublinear compactness) functional encryption essentially directly yields a sublinear RE in the CRS model, and as such we get an alternative, modular, and simpler proof of the results of [AJ15, BV15] showing that subexponentially-secure sublinearly compact FE implies **iO**.
- **Applications to iO for Unbounded-input Turing machines:** Subexponentially-secure compact RE for natural *restricted* classes of distributions over programs and inputs (which are not ruled out by our impossibility result, and for which we can give candidate constructions) imply **iO** for *unbounded-input* Turing machines. This yields the first construction of **iO** for unbounded-input Turing machines that does not rely on (public-coin) differing-input obfuscation.

*University of California at Santa Barbara, Email: `rachel.lin@cs.ucsb.edu`.

†Cornell University, Email: `rafael@cs.cornell.edu`. Work supported in part by a Alfred P. Sloan Fellowship, Microsoft New Faculty Fellowship, NSF Award CNS-1217821, NSF CAREER Award CCF-0746990, NSF Award CCF-1214844, AFOSR YIP Award FA9550-10-1-0093, and DARPA and AFRL under contract FA8750-11-2-0211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

‡Cornell University, Email: `karn@cs.cornell.edu`.

§Cornell University, Email: `sidtelang@cs.cornell.edu`.

1 Introduction

The beautiful notion of a *randomized encoding (RE)*, introduced by Ishai and Kushilevitz [IK00], aims to trade the computation of a "complex" (deterministic) function Π on a given input x for the computation of a "simpler" randomized function—the "encoding algorithm"—whose output distribution $\hat{\Pi}(x)$ encodes $\Pi(x)$ (from which $\Pi(x)$ can be efficiently decoded, or "evaluated"). Furthermore, the encoding $\hat{\Pi}(x)$ should not reveal anything beyond $\Pi(x)$; this is referred to as the *privacy*, or *security*, property of randomized encodings and is typically defined through the simulation paradigm [GMR89].

Most previous work have focused on randomized encodings where encodings can be computed in lower parallel-time complexity than what is required for computing the original function Π. For instance, all log-space computations have *perfectly-secure* randomized encodings in $\mathbf{NC}^0$ [IK00, IK02a, AIK04], and assuming low-depth pseudo-random generators, this extends to all polynomial-time computations (with computational security) [AIK06, Yao82]. Such randomized encodings have been shown to have various applications to parallel cryptography, secure computation, verifiable delegation, etc (see [App11] for a survey).

Bitansky, Garg, Lin, Pass and Telang [BGL$^+$15] recently initiated a study of *succinct randomized encodings* where we require that the *time* required to compute $\hat{\Pi}(x)$ is smaller than the time required to compute $\Pi(x)$; their study focused on functions Π that have *single-bit* outputs. [BGL$^+$15, CHJV14, KLW14] show that subexponentially-secure indistinguishability obfuscators ($\mathbf{iO}$) [BGI$^+$01, GGH$^+$13] and one-way functions[1] imply the existence of such succinct randomized encodings for all polynomial-time Turing machines that output just a single bit.

We here further the study of such objects, focusing on functions Π with *long* outputs. Given a description of a Turing machine Π and an input x, we consider two notions of efficiency for randomized encodings $\hat{\Pi}(x)$ of $\Pi(x)$ with running time T.

- *compact RE*: Encoding time (and thus also size of the encodings) is $\mathrm{poly}(|\Pi|, |x|, \log T)$
- *sublinear RE*: Encoding time (and thus also size) is bounded by $\mathrm{poly}(|\Pi|, |x|) * T^{1-\epsilon}$, for some $\epsilon > 0$.

We assume without loss of generality that the randomized encoding $\hat{\Pi}(x)$ of Π, x itself is a program, and that the decoding/evaluation algorithm simply executes $\hat{\Pi}(x)$.

It is easy to see that for such notions of efficiency, the standard simulation-based notion of security is impossible to achieve—roughly speaking, the simulator given just $\Pi(x)$ needs to output a "compressed" version of it, which is impossible if $\Pi(x)$ has high pseudo-Kolmogorov complexity (e.g., if Π is a PRG); we formalize this argument in Theorem 10 in Section 7. Consequently, we consider weaker indistinguishability-based notions of privacy. One natural indistinguishability based notion of privacy simply requires that encoding $\hat{\Pi}_0(x_0)$ and $\hat{\Pi}_1(x_1)$ are indistinguishable as long as $\Pi_0(x_0) = \Pi_1(x_1)$ and $\mathsf{Time}(\Pi_0(x_0)) = \mathsf{Time}(\Pi_1(x_1))$, where $\mathsf{Time}(\Pi(x))$ is the running-time of $\Pi(x)$; such a notion was recently considered by Ananth and Jain [AJ15]. In this work, we consider a stronger notion which requires indistinguishability of $\hat{\Pi}_0(x_0)$ and $\hat{\Pi}_0(x_1)$ as long as Π_0, x_0 and Π_1, x_1 are sampled from some distributions such that $\Pi_0(x_0), \mathsf{Time}(\Pi_0(x_0))$ and $\Pi_1(x_1), \mathsf{Time}(\Pi_1(x_1))$ are indistinguishable. We refer to this notion as *distributional indistinguishability security*, and note that it easily follows that the standard simulation-based security implies distributional indistinguishability security.

[1]The one-way function assumption can be weakened to assume just that $\mathsf{NP} \not\subseteq io\mathsf{BPP}$ [KMN$^+$14].

The goal of this paper is to investigate compact and sublinear RE satisfying the above-mentioned distributional indistinguishability notion. For the remainder of the introduction, we refer to randomized encodings satisfying distributional indistinguishability security as simply *RE*. For comparison, we refer to randomized encodings with the weaker (non-distributional) indistinguishability security as *weak RE*.

Compact RE v.s. Obfuscation Before turning to describe our results, let us point out that RE can be viewed as (a degenerate form) of obfuscation for special classes of programs.

Recall that an indistinguishability obfuscator (**iO**) [BGI$^+$01, GGH$^+$13] is a method $\mathcal{O}$ for "scrambling" a program Π into $\mathcal{O}(\Pi)$ such that for any two functionally equivalent programs Π_0, Π_1 (that is, their outputs and run-time are the same on all inputs,) $\mathcal{O}(\Pi_0)$ is indistinguishable from $\mathcal{O}(\Pi_1)$. **iO** for Turing machines [BGI$^+$01, BCP14, ABG$^+$13] additionally requires that the size of the obfuscated code does not grow (more than polylogarithmically) with the running-time of the Turing machine.

We may also consider a useful strengthening of this notion—which we call "puncturable **iO**"—which, roughly speaking, requires indistinguishability of $\mathcal{O}(\Pi_0)$ and $\mathcal{O}(\Pi_1)$ as long as Π_0 and Π_1 differ *on at most one input* x^* and their outputs on input x^* are indistinguishable. More precisely, we say that a distribution D is *admissible* if there exists some x^* such that a) for every triple (Π_0, Π_1, Π) in the support of D, and every $x \neq x^*$, it holds that $\Pi_0(x) = \Pi_1(x) = \Pi(x)$, and b) $(\Pi, \Pi_0(x^*))$ and $(\Pi, \Pi_1(x^*))$ are computationally indistinguishable when (Π_0, Π_1, Π) are sampled randomly from D. Puncturable **iO** requires indistinguishability of $\mathcal{O}(\Pi_0)$ and $iO(\Pi_1)$ for Π_0, Π_1 sampled from any admissible distribution. Interestingly, for the case of *circuits*, puncturable **iO** is equivalent to (standard) **iO**.[2] Indeed, such a notion is implicitly used in the beautiful and powerful punctured-program paradigm by Sahai and Waters [SW14], and all its applications. (In this context, think of Π as the "punctured" version of the programs Π_0, Π_1.)

In the case of Turing machines, when restricting to the degenerate case of Turing machines with no inputs (or more precisely, we only consider the execution of $\Pi()$ on the "empty" input), the notion of **iO** for Turing machines is equivalent to the notion of a compact *weak* RE. Compact RE, on the other hand, is equivalent to puncturable **iO** for Turing machines (without inputs). (Jumping ahead, as we shall see, for the case of Turing machines it is unlikely that puncturable **iO** is equivalent to standard **iO**.)

1.1 Our results

iO from sublinear RE We start by showing that sublinear RE is an extremely useful primitive: Subexponentially-secure sublinear RE implies indistinguishability obfuscators for all polynomial-size circuits.

Theorem 1. *The existence of subexponentially-secure sublinear RE and one-way functions implies the existence of subexponentially-secure **iO** for circuits.*

Before continuing, let us mention that Theorem 1 is related to a recent beautiful result by Ananth and Jain [AJ15] which shows that *under the LWE assumption*, subexponentially-secure compact RE (satisfying only the weak indistinguishability security) implies **iO** for circuits. Their

[2] To see this, consider a hybrid program $\Pi^y(x)$ that runs $\Pi(x)$ if $x \neq x^*$ and otherwise (i.e., if $x = x^*$ outputs y). By the **iO** property we have that for every Π, Π_0, Π_1 in the support of D, $\mathcal{O}(\Pi^{\Pi_b(x^*)})$ is indistinguishable from $\mathcal{O}(\Pi_b)$. Thus, if $\mathcal{O}(\Pi_0)$, $\mathcal{O}(\Pi_1)$ are distinguishable, so are $\mathcal{O}(\Pi^{\Pi_0(x^*)})$, $\mathcal{O}(\Pi^{\Pi_1(x^*)})$, which contradicts indistinguishability of $(\Pi, \Pi_0(x^*))$ and $(\Pi, \Pi_1(x^*))$.

construction goes from RE to *functional encryption* (FE) [BSW11], and then from FE to **iO**; (the first step relies on previous constructions of FE [GKP$^+$13, GVW13], while the second step relies on a sequence of complex transformations and analysis). In contrast, the proof of Theorem 1 directly constructs **iO** from RE in a surprisingly simple way: We essentially use the GGM construction [GGM86] that builds a PRF from a PRG using a tree, but replace the PRG with a RE. Let us explain in more details below.

Consider a program Π taking n-bit inputs. We consider a binary tree where the leafs are randomized encodings of the function applied to all possible inputs, and each node in the tree is a randomized encoding that generates its two children. More precisely, given a sequence of bits $x_1, \cdots, x_i$, let $\tilde{\Pi}_{R,x_1,\cdots,x_i}$ denote an (input-less) program that

- if $i = n$ simply outputs a RE of the program Π and input $(x_1, \cdots, x_n)$ using R as randomness, and
- otherwise, after expanding R_0, R_1, R_2, R_3 from R using a PRG, outputs randomized encodings of (input-less) programs $\tilde{\Pi}_{R_0,x_1,\cdots,x_i,0}$ and $\tilde{\Pi}_{R_1,x_1,\cdots,x_i,1}$ using respectively R_2, R_3 as randomness.

We associate each node in the binary tree that has index $x_1, \cdots, x_i$ with a randomized encoding of the program $\tilde{\Pi}_{R,x_1,\cdots,x_i}$, denoted as $\hat{\Pi}_{R,x_1,\cdots,x_i}$. In particular, the root of the tree is associated with a randomized encoding $\hat{\Pi}$ of the (initial) program $\tilde{\Pi}_R$ hardwired with a randomly chosen R.

The obfuscation of Π is now a program with the "root" $\hat{\Pi}$ hardcoded, and given an input x, computes the path from the root to the leaf x – by recursively evaluating the randomized encodings associated with nodes on the path – and finally outputs the evaluation of the leaf. More precisely, on input x, evaluate $\hat{\Pi}$ to obtain $\hat{\Pi}_0, \hat{\Pi}_1$, next evaluate $\hat{\Pi}_{x_1}$ to obtain $\hat{\Pi}_{x_1,0}, \hat{\Pi}_{x_1,1}$, so on and so forth until $\hat{\Pi}_{x_1,\cdots,x_n}$ is evaluated, yielding the output $\Pi(x_1, \cdots, x_n)$.

Note that for any two functionally equivalent programs, the randomized encodings associated with individual leaf node are computationally indistinguishable by the indistinguishability security property (the non-distributional version suffices here). Then, by the distributional indistinguishability security, the randomized encodings associated with tree nodes one layer above are also indistinguishable. Thus, by induction, it follows that the roots are indistinguishable, which implies that obfuscations of functionally equivalent programs are indistinguishable. Let us note that the reason that subexponential security is needed is that each time we go up one level in the tree (in the inductive argument), we loose at least a factor 2 in the indistinguishability gap (as each node generates two randomized encodings, its children). Hence, we need to ensure that encodings are at least poly(2^n)-indistinguishable, which can be done by scaling up the security parameter.

On the existence of Compact and Sublinear RE We next turn to investigating the existence of compact and sublinear RE. We show—assuming just the existence of subexponentially-secure one-way functions—*impossibility* of subexponentially-secure sublinear (and thus also compact) RE.[3]

Theorem 2. *Assume the existence of subexponentially secure one-way functions. Then, there do not exists subexponentially-secure sublinear RE.*

As observed above, compact RE can be interpreted as a stronger notion of **iO** (which we referred to as *puncturable* ***iO***) for "degenerate" input-less Turing machines, and as such Theorem

[3]This result was established after hearing that Bitansky and Paneth had ruled out compact RE assuming public-coin differing-input obfuscation for Turing Machines and collision-resistant hashfunctions. We are very grateful to them for informing us of their result.

2 rules out (assuming just one-way functions) such a natural strengthening of **iO** for (input-less) Turing machines. We note that this impossibility stands in contrast with the case of *circuits* where puncturable **iO** is equivalent to **iO**.

We remark that although it may seem like Theorem 2 makes Theorem 1 pointless, it turns out that Theorem 1 plays a crucial role in the proof of Theorem 2: Theorem 2 is proven by first ruling out sublinear (even just polynomially-secure) RE *assuming* **iO** and one-way functions. Next, by using Theorem 1, the **iO** assumption comes for free if considering subexponentially-secure RE. That is, assuming one-way functions, we have the following paradigm:

$$\text{sub-exp secure sublinear RE} \overset{\text{Theorem 1}}{\Longrightarrow} \mathbf{iO} \Longrightarrow \text{impossibility of (poly secure) sublinear RE}$$

Let us now briefly sketch how to rule out sublinear RE assuming **iO** and one-way functions (as mentioned, Theorem 2 is then deduced by relying on Theorem 1). The idea is somewhat similar to the non-black-box zero-knowledge protocol of Barak [Bar01].

Let $\Pi^b_{s,u}$ be a program that takes no input and outputs a sufficiently long pseudo-random string $y = \mathsf{PRG}(s)$ and an indistinguishability obfuscation $\tilde{R}^b_y$ (generated using pseudo-random coins $\mathsf{PRG}(u)$) of the program R^b_y. The program R^b_y takes input Σ of length $|y|/2$, and outputs b iff Σ, when interpreted as an input-less Turing machine, generates y; in all other cases, it outputs $\perp$.[4] We note that the size of the program $\Pi^b_{s,u}$ is linear in the security parameter λ, whereas the pseudo-random string y it generates could have length $|y| = \lambda^\alpha$ for any sufficiently large constant α.

Consider the pair of distributions $\Pi^0_{U_\lambda,U_\lambda}$ and $\Pi^1_{U_\lambda,U_\lambda}$ that samples respectively programs $\Pi^0_{s,u}$ and $\Pi^1_{s,u}$ as described above with random s and u. We first argue that their outputs are computationally indistinguishable. Recall that the output of $\Pi^b_{s,u}$ is a pair $(y, \tilde{R}^b_y)$. By the pseudorandomness of PRG, this output distribution is indistinguishable from $(X, \tilde{R}^b_X)$ where X a uniformly distributed random variable over λ^α bit strings. With overwhelming probability X has high Kolmogorov complexity, and when this happens R^b_X is functionally equivalent to the program $R_\perp$ that always outputs $\perp$. Therefore, by the security of the **iO**, the output of programs sampled from $\Pi^b_{U_\lambda,U_\lambda}$ is computationally indistinguishable to $(X, \tilde{R}_\perp)$, and hence outputs of $\Pi^0_{U_\lambda,U_\lambda}$ and $\Pi^1_{U_\lambda,U_\lambda}$ are indistinguishable.

Let us now turn to showing that randomized encodings of $\Pi^0_{U_\lambda,U_\lambda}$ and $\Pi^1_{U_\lambda,U_\lambda}$ can be distinguished. Recall that a randomized encoding $\hat{\Pi}^b$ of $\Pi^b_{U_\lambda,U_\lambda}$ itself can be viewed as a (input-less) program that outputs $(y, \tilde{R}^b_y)$. Given $\hat{\Pi}^b$, the distinguisher can thus first evaluate $\hat{\Pi}^b$ to obtain $(y, \tilde{R}^b_y)$ and next evaluate $\tilde{R}^b_y(\hat{\Pi}^b)$ to attempt to recover b. Note that $\hat{\Pi}^b$ clearly is a program that generates y (as its first input); furthermore, if the RE scheme is compact, the length of the program $|\hat{\Pi}^b|$ is bounded by $\mathrm{poly}(\lambda, \log \lambda^\alpha)$, which is far smaller than $|y|/2 = \lambda^\alpha/2$ when α is sufficiently large. Therefore, $\Sigma = \hat{\Pi}^b$ is indeed an input that makes $\tilde{R}^b_y$ output b, enabling the distinguisher to distinguish $\hat{\Pi}^0$ and $\hat{\Pi}^1$ with probability close to 1!

Finally, if the RE is only sublinear, the length of the encoding $|\hat{\Pi}^b|$ is only sublinear in the output length, in particular, bounded by $\mathrm{poly}(\lambda)(\lambda^\alpha)^{1-\epsilon}$ for some constant $\epsilon > 0$. If $\alpha > 1/(1-\epsilon)$ (which clearly happens if ϵ is sufficiently small), then we do not get enough "compression" for the above proof to go through. We circumvent this problem by composing a sublinear RE with itself a sufficient (constant) number of times—to compose once, consider creating randomized encoding

[4]To enable this, we require **iO** for bounded-input Turing machines, whereas Theorem 1 only gives us **iO** for circuits. However, by the results of [BGL+15, CHJV14, KLW14] we can go from **iO** for circuits to **iO** for bounded-inputs Turing machines.

of the randomized encoding of a function, instead of the function itself; each time of composition reduces the size of the encoding to be w.r.t. a smaller exponent $1-\epsilon'$. Therefore, it is without loss of generality to assume that ϵ is any sufficiently *big* constant satisfying $\alpha << 1/(1-\epsilon)$; so the desired compression occurs.

Sublinear RE in the CRS model from sublinear FE Despite Theorem 2, not all is lost. We remark that any sublinear functional encryption scheme (FE) [AJ15, BV15] almost directly yields a sublinear RE in the *Common Reference String (CRS)* model.

Theorem 3. *Assume the existence of subexponentially-secure sublinear (resp. compact) FE. Then there exists a subexponentially-secures sublinear (resp. compact) RE in the CRS model.*

Furthermore, Theorem 1 straightforwardly extends also to RE in the CRS model. Taken together, these result provide an alternative, modular, simpler proof of the recent results of Ananth and Jain [AJ15] and Bitansky and Vaikuntanathan [BV15] showing that subexponentially-secure sublinear FE implies subexponentially-secure **iO**. (All these approaches, including a related work by Brakerski, Komargodski and Segev [BKS15] have one thing in common though: they all proceed by processing inputs one bit at a time, and hard-coding parts of input to the program.)

Theorem 4 (informal, alternative proof of [BV15, AJ15])**.** *Assume the existence of subexponentially-secure sublinear FE. Then there exists a subexponentially-secure **iO** for circuits.*

Toward Turing Machine Obfuscation with Unbounded Inputs We finally address the question of constructing indistinguishability obfuscators for Turing machines with *unbounded* inputs. (For the case of Turing machine obfuscation with unbounded-length inputs, the same obfuscated code needs to work for every input-length, and in particular, the size of the obfuscated code cannot grow with it.) Although it is known that subexponentially secure **iO** for circuits implies **iO** for Turing machines with *bounded inputs lengths* [BGL$^+$15, CHJV14, KLW14] , the only known construction of **iO** for Turing machines with unbounded inputs relies on (public-coin) differing-input obfuscation for circuits and (public-coin) SNARKs [BCP14, ABG$^+$13, IPS15]—these are strong "extractability" assumptions (and variants of them are known to be implausible [BCPR13, GGHW13, BP15]).

We note that the construction from Theorem 1 easily extends to show that subexponentially-secure *compact* RE implies **iO** for Turing machines with unbounded input: instead of having a binary tree, we have a ternary tree where the "third" child of a node is always a leaf; that is, for a tree node corresponding to $x_1, \cdots, x_i$, its third child is associated with a randomized encoding of program Π, and input $(x_1, \cdots, x_i)$, which can be evaluated to obtain output $\Pi(x_1, \cdots x_i)$. Then, by using a tree of *super-polynomial* depth, we can handle any polynomial-length input. Note that since obfuscating a program only involves computing the root RE (as before), the obfuscation is still efficient. Moreover, for any input, we still compute the output of the program in time polynomial in the length of the input by evaluating the "third" child of the node when all input bits have been processed.[5]

But as shown in Theorem 2, compact RE cannot exists (assuming one-way functions)! However, just as for the case of differing-inputs obfuscation and SNARKs, we may assume the existence of compact RE for *restricted* types of "nice" distributions (over programs and inputs), for which

[5]Proving security becomes slightly more problematic since there is no longer a polynomial bound on the depth of the tree (recall that we required poly(2^n)-indistinguishable RE to deal with inputs of length n). Thus issue, however, can be dealt with by using larger and larger security parameters for RE that are deeper down in the tree.

impossibility does not hold, yet the construction in Theorem 1 still works. We formalize one natural class of such distributions, and may assume that the **iO** for bounded-input Turing machines construction of [KLW14] (based on **iO** for circuits) yields such a compact RE (for the restricted class of distributions). This yields a new candidate construction of unbounded input Turing machines (based on a very different type of assumption than known constructions).

2 Preliminaries

Let $\mathcal{N}$ denote the set of positive integers, and $[n]$ denote the set $\{1, 2, \ldots, n\}$. We denote by PPT probabilistic polynomial time Turing machines. The term negligible is used for denoting functions that are (asymptotically) smaller than one over any polynomial. More precisely, a function $\nu(\cdot)$ from non-negative integers to reals is called negligible if for every constant $c > 0$ and all sufficiently large n, it holds that $\nu(n) < n^{-c}$.

Turing machine notation For any Turing machine Π, input $x \in \{0,1\}^*$ and time bound $T \in \mathbb{N}$, we denote by $\Pi^T(x)$ the output of Π on x when run for T steps. We refer to $\{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ as a class of Turing machines. One particular class we will consider is the class of Turing machines that have 1-bit output. We call such a machine a Boolean Turing machine. Throughout this paper, by *Turing machine* we refer to a machine with *multi-bit* output unless we explicitly mention it to be a Boolean Turing machine.

2.1 Concrete Security

Definition 1 ($(\lambda_0, S(\cdot))$-indistinguishability)**.** *A pair of distributions X, Y are S-indistinguishable for some $S \in \mathbb{N}$ if every S-size distinguisher D it holds that*

$$|\Pr[x \xleftarrow{\$} X : D(x) = 1] - \Pr[y \xleftarrow{\$} Y : D(y) = 1]| \leq \frac{1}{S}$$

A pair of ensembles $\{X_\lambda\}$, $\{Y_\lambda\}$ are $(\lambda_0, S(\cdot))$-indistinguishable for some $\lambda_0 \in \mathbb{N}$ and $S : \mathbb{N} \to \mathbb{N}$ if for every security parameter $\lambda > \lambda_0$, the distributions X_λ and Y_λ are $S(\lambda)$ indistinguishable.

Discussion on $(\lambda_0, S(\cdot))$-indistinguishability: We remark that the above definition requires that there is a universal λ_0 that works for all distinguisher D. A seemingly weaker variant could switch the order of quantifiers and only require that for every distinguisher D there is a λ_0. We show that the above definition is w.l.o.g, since it is implied by the following standard definition with auxiliary inputs in the weaker fashion.

Let U be a universal TM that on an input x and a circuit C computes $C(x)$. Let $S' : N \to N$ denote the run time $S'(S)$ of U on input a size S circuit.

Definition. *A pair of ensembles $\{X_\lambda\}$, $\{Y_\lambda\}$ are $S(\cdot)$-indistinguishable if for every $S' \circ S(\cdot)$-time uniform TM distinguisher D, there exists a $\lambda_0 \in N$, such that, for every security parameter $\lambda > \lambda_0$, and every auxiliary input $z = z_\lambda \in \{0,1\}^*$,*

$$|\Pr[x \xleftarrow{\$} X_\lambda : D(1^\lambda, x, z) = 1] - \Pr[y \xleftarrow{\$} Y_\lambda : D(1^\lambda, y, z) = 1]| \leq \frac{1}{S(\lambda)}$$

This definition implies $(\lambda_0, S(\cdot))$-indistinguishability. Consider a distinguisher D that on input $(1^\lambda, x, z)$ runs the universal TM $U(x, z)$, and let λ_U be the constant associated with it. For any

$\lambda > \lambda_U$, and every $S(\lambda)$-size circuit C, by setting the auxiliary input $z = C$, the above definition implies that the distinguishing gap by C is at most $1/S(\lambda)$. Therefore, λ_U is effectively the universal constant that works for all (circuit) distinguisher.

Below, we state definitions of cryptographic primitives using $(\lambda_0, S(\cdot))$-indistinguishability. Traditional polynomial or sub-exponential security can be directly derived from such more concrete definitions as follows:

Definition 2 (Polynomial Indistinguishability)**.** *A pair of ensembles $\{X_\lambda\}$, $\{Y_\lambda\}$ are polynomially indistinguishable if for every polynomial $p(\cdot)$, there is a constant $\lambda_p \in N$, such that, the two ensembles are $(\lambda_p, p(\cdot))$-indistinguishable.*

Definition 3 (Sub-exponential Indistinguishability)**.** *A pair of ensembles $\{X_\lambda\}$, $\{Y_\lambda\}$ are sub-exponentially indistinguishable, if there is a sub-exponential function $S(\lambda) = 2^{\lambda^\varepsilon}$ with $\varepsilon \in (0,1)$ and a constant $\lambda_0 \in N$, such that, the two ensembles are $(\lambda_0, S(\cdot))$-indistinguishable.*

2.2 Standard cryptographic primitives

Definition 4 (Pseudorandom Generator)**.** *A deterministic PT uniform machine* PRG *is a pseudorandom generator if the following conditions are satisfied:*

Syntax *For every $\lambda, \lambda' \in \mathbb{N}$ and every $r \in \{0,1\}^\lambda$, $\mathsf{PRG}(r, \lambda')$ outputs $r' \in \{0,1\}^{\lambda'}$*

$(\lambda_0, S(\cdot))$-Security *For every function $p(\cdot)$, such that, $p(\lambda) \leq S(\lambda)$ for all λ, the following ensembles are $(\lambda_0, S(\cdot))$ indistinguishable*

$$\left\{r \xleftarrow{\$} \{0,1\}^\lambda : \mathsf{PRG}(r, p(\lambda))\right\} \ \left\{r' \xleftarrow{\$} \{0,1\}^{p(\lambda)}\right\}$$

2.3 Indistinguishability Obfuscation

In this section, we recall the definition of indistinguishability obfuscation for Turing machines from [BGI+01, BCP14, ABG+13]. Following [BCP14], we considers two notions of obfuscation for Turing machines. The first definition, called *bounded-input* indistinguishability obfuscation, only requires the obfuscated program to work for inputs of *bounded length* and furthermore the size of the obfuscated program may depend polynomially on this input length bound. (This is the notion achieved in [BGL+15, CHJV14, KLW14] assuming subexponentially-secure **iO** for circuits and one-way functions.)

The second notion considered in [BCP14] is stronger and requires the obfuscated program to work on any arbitrary polynomial length input (and the size of the obfuscated machine thus only depends on the program size and security parameter). We refer to this notion as *unbounded-input* indistinguishability obfuscation. (This stronger notion of unbounded-input indistinguishability obfuscator for Turing machines is only known to be achievable based on strong "extractability assumptions"—namely, (public-coin) differing-input obfuscation for circuits and (public-coin) SNARKs [BCP14, ABG+13, IPS15], variants of which are known to be implausible [BCPR13, GGHW13, BP15]).

Definition 5 (Indistinguishability Obfuscator ($i\mathcal{O}$) for a class of Turing machines)**.** *An indistinguishability obfuscator for a class of Turing machines $\{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ is a uniform machine that behaves as follows:*

$\hat{\Pi} \leftarrow i\mathcal{O}(1^\lambda, \Pi, T)$: *$i\mathcal{O}$ takes as input a security parameter 1^λ, the Turing machine to obfuscate $\Pi \in \mathcal{M}_\lambda$ and a time bound T for Π. It outputs a Turing machine $\hat{\Pi}$.*

We require the following conditions to hold.

Correctness: *For every $\lambda \in N$, $\Pi_\lambda \in \mathcal{M}_\lambda$, input x_λ and time bound T_λ,*

$$\Pr[(\tilde{\Pi} \xleftarrow{\$} i\mathcal{O}(1^\lambda, \Pi_\lambda, T_\lambda) : \ \tilde{\Pi}(x_\lambda) = \Pi^T(x_\lambda)] = 1 \ .$$

Efficiency: *The running times of $i\mathcal{O}$ and $\hat{\Pi}$ are bounded as follows:*

There exists polynomial p such that for every security parameter λ, Turing machine $\Pi \in \mathcal{M}_\lambda$, time bound T and every obfuscated machine $\hat{\Pi} \leftarrow i\mathcal{O}(1^\lambda, \Pi, T)$ and input x, we have that

$$\mathsf{Time}_{i\mathcal{O}}(1^\lambda, \Pi, T) \le p(\lambda, |\Pi|, \log T)$$
$$\mathsf{Time}_{\hat{\Pi}}(x) \le p(\lambda, |\Pi|, |x|, T)$$

$(\lambda_0, S(\cdot))$-Security: *For every ensemble of pairs of Turing machines and time bounds $\{\Pi_{0,\lambda}, \Pi_{1,\lambda}, T_\lambda\}$ where for every $\lambda \in \mathbb{N}$, $\Pi_0 = \Pi_{0,\lambda}$, $\Pi_1 = \Pi_{1,\lambda}$, $T = T_\lambda$, satisfying the following*

$$\Pi_0, \Pi_1 \in \mathcal{M}_\lambda \quad |\Pi_0| = |\Pi_1| \le \mathrm{poly}(\lambda) \quad T \le \mathrm{poly}(\lambda)$$
$$\forall x, \Pi_0^T(x) = \Pi_1^T(x) \ ,$$

the following ensembles are $(\lambda_0, S(\cdot))$-indistinguishable

$$\left\{ i\mathcal{O}(1^\lambda, \Pi_{0,\lambda}, T_\lambda) \right\} \ \left\{ i\mathcal{O}(1^\lambda, \Pi_{1,\lambda}, T_\lambda) \right\} \ .$$

Definition 6 (Unbounded-input indistinguishability obfuscator for Turing machines)**.** *An unbounded-input indistinguishability obfuscator for Turing machines $i\mathcal{O}(\cdot, \cdot, \cdot)$ is simply an indistinguishability obfuscator for the class of all Boolean Turing machines.*

Remark 1 (Obfuscation for Boolean Turing machines is without loss of generality)**.** *The above definition is equivalent to one that considers the class of all Turing machines. Any Turing machine with output length m can be represented as a Boolean Turing machine that takes in an additional input $i \in [m]$ and returns the i^{th} bit of the m-bit long output.*

Definition 7 (Bounded-input indistinguishability obfuscator for Turing machines)**.** *A bounded-input indistinguishability obfuscator for Turing machines $i\mathcal{O}(\cdot, \cdot, \cdot, \cdot)$ is a uniform machine such that for every polynomial p, $i\mathcal{O}(p, \cdot, \cdot, \cdot)$ is an indistinguishability obfuscator for the class of Turing machines $\{\mathcal{M}_\lambda\}$ where $\mathcal{M}_\lambda$ are machines that accept only inputs of length $p(\lambda)$. Additionally, $i\mathcal{O}(p, 1^\lambda, \Pi, T)$ is allowed to run in time* $\mathrm{poly}(p(\lambda) + \lambda + |\Pi| + \log T)$.

Finally, we define weaker variants of the above definitions where the size of the obfuscated program is sub-linear (instead of poly-logarithmic) in the time bound.

Definition 8 (Sub-linear efficiency for Indistinguishability Obfuscators)**.** *We say an indistinguishability obfuscator $i\mathcal{O}$ for a class of Turing machines $\{\mathcal{M}_\lambda\}$ has sub-linear efficiency if it satisfies the requirements in Definition 5 with the efficiency requirement replaced with the following.*

Efficiency: *The running times of $i\mathcal{O}$ and $\hat{\Pi}$ are bounded as follows:*

There exists polynomial p and constant $\epsilon > 0$ such that for every security parameter λ, Turing machine $\Pi \in \mathcal{M}_\lambda$, time bound T and every obfuscated machine $\hat{\Pi} \leftarrow i\mathcal{O}(1^\lambda, \Pi, T)$ and input x, we have that

$$\mathsf{Time}_{i\mathcal{O}}(1^\lambda, \Pi, T) \le p(\lambda, |\Pi|) T^{1-\epsilon}$$
$$\mathsf{Time}_{\hat{\Pi}}(x) \le p(\lambda, |\Pi|, |x|, T)$$

2.4 Functional Encryption

Definition 9 (Selectively-secure Single-Query Public-key Functional Encryption)**.** *A tuple of PPT algorithms* $(\mathsf{FE.Setup}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *is a selectively-secure functional encryption scheme for a class of circuits* $\{\mathcal{C}_\lambda\}$ *if it satisfies the following properties.*

Completeness *For every* $\lambda \in \mathbb{N}$, $C \in \mathcal{C}_\lambda$ *and message* $m \in \{0,1\}^*$,

$$\Pr\left[\begin{array}{c}(mpk, msk) \leftarrow \mathsf{FE.Setup}(1^\lambda)\\ c \leftarrow \mathsf{FE.Enc}(1^\lambda, m)\\ sk_C \leftarrow \mathsf{FE.KeyGen}(msk, C)\end{array} : C(m) \leftarrow \mathsf{FE.Dec}(sk_C, c)\right] = 1$$

$(\lambda_0, S(\cdot))$**-Selective-security** *For every ensemble of circuits and pair of messages* $\{C_\lambda, m_{0,\lambda}, m_{1,\lambda}\}$ *where* $C_\lambda \in \mathcal{C}_\lambda$, $|C_\lambda|, |m_{0,\lambda}|, |m_{1,\lambda}| \leq \mathrm{poly}(\lambda)$, *and* $C_\lambda(m_{0,\lambda}) = C_\lambda(m_{1,\lambda})$, *the following ensembles of distributions* $\{D_{0,\lambda}\}$ *and* $\{D_{1,\lambda}\}$ *are* $(\lambda_0, S(\cdot))$*-indistinguishable.*

$$D_{b,\lambda} = \left(\begin{array}{c}(mpk, msk) \leftarrow \mathsf{FE.Setup}(1^\lambda)\\ c \leftarrow \mathsf{FE.Enc}(1^\lambda, m_{b,\lambda})\\ sk_C \leftarrow \mathsf{FE.KeyGen}(msk, C_\lambda)\end{array} : mpk, c, sk_C\right)$$

We note that in this work, we only need the security of the functional encryption scheme to hold with respect to statically chosen challenge messages and functions.

Definition 10 (Compact Functional Encryption)**.** *We say a functional encryption scheme is* compact *if it additionally satisfies the following requirement:*

Compactness *The running time of* $\mathsf{FE.Enc}$ *is bounded as follows.*

There exists a polynomial p *such that for every security parameter* $\lambda \in \mathbb{N}$ *and message* $m \in \{0,1\}^*$, $\mathsf{Time}_{\mathsf{FE.Enc}}(1^\lambda, m) \leq p(\lambda, |m|)$.

Furthermore, we say the functional encryption scheme has sub-linear compactness *if there exists a polynomial* p *and constant* $\epsilon > 0$ *such that for every security parameter* $\lambda \in \mathbb{N}$ *and message* $m \in \{0,1\}^*$, $\mathsf{Time}_{\mathsf{FE.Enc}}(1^\lambda, m) \leq p(\lambda, |m|)s^{1-\epsilon}$ *where* $s = \max_{C \in \mathcal{C}_\lambda} |C|$.

3 Randomized Encoding Schemes

Roughly speaking, randomized encoding schemes encodes a computation of a program Π on an input x, into an encoded computation $(\hat{\Pi}, \hat{x})$, with the following two properties: First, the encoded computation evaluates to the same output $\Pi(x)$, while leaking no other information about Π and x. Second, the encoding is "simpler" to compute than the original computation. In the literature, different measures of simplicity have been considered. For instance, in the original works by [IK02a, AIK06], the depth of computation is used and it was shown that any computation in P can be encoded in NC_1 using Yao's garbled circuits [Yao82]. A recent line of works [BGL+15, CHJV14, KLW14] uses the time-complexity as the measure and show that any *Boolean* Turing machine computation can be encoded in time poly-logarithmic in the run-time of the computation.

Traditionally, the security of randomized encoding schemes are capture via simulation. In this work, we consider a new *distributional* indistinguishability-based security notion, and show that it is implied by the transitional simulation security. Additionally, we further explore how compact the encoded computation can be: Similar to the recent works [BGL+15, CHJV14, KLW14], we consider encoding whose size depends poly-logarithmically on the run-time of the encoded computation; but differently, we directly consider Turing machines with arbitrary length outputs, and require the size of the encoding to be independent of the output length. Such scheme is called a compact randomized encoding scheme.

3.1 Randomized Encoding with Simulation Security

In this section, we recall the traditional definition of randomized encoding with simulation security [IK02a, AIK06].

Definition 11 (Randomized Encoding Scheme for a Class of Turing Machines). *A Randomized Encoding scheme* RE *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *consists of two algorithms,*

- $(\hat{\Pi}, \hat{x}) \overset{\$}{\leftarrow} \mathsf{Enc}(1^\lambda, \Pi, x, T)$*: On input a security parameter* 1^λ*, Turing machine* $\Pi \in \mathcal{M}_\lambda$*, input* x *and time bound* T*,* Enc *generates an encoded machine* $\hat{\Pi}$ *and encoded input* $\hat{x}$*.*
- $y = \mathsf{Eval}(\hat{\Pi}, \hat{x})$*: On input* $(\hat{\Pi}, \hat{x})$ *produced by* $\mathsf{Enc}, \mathsf{Eval}$ *outputs* y*.*

Correctness: *The two algorithms* Enc *and* Eval *satisfy the following correctness condition: For all security parameters* $\lambda \in \mathbb{N}$*, Turing machines* $\Pi \in \mathcal{M}_\lambda$*, inputs* x *and time bounds* T*, it holds that,*

$$\Pr[(\hat{\Pi}, \hat{x}) \overset{\$}{\leftarrow} \mathsf{Enc}(1^\lambda, \Pi, x, T) : \ \mathsf{Eval}(\hat{\Pi}, \hat{x}) = \Pi^T(x))] = 1$$

Definition 12 ($(\lambda_0, S(\cdot))$-Simulation Security). *A randomized encoding scheme* RE *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *satisfies* $(\lambda_0, S(\cdot))$**-simulation security***, if there exists a PPT algorithm* Sim *and a constant* c*, such that, for every polynomial* B*, and ensemble* $\{\Pi_\lambda, x_\lambda, T_\lambda\}$ *where* $\Pi_\lambda \in \mathcal{M}_\lambda$ *and* $|\Pi_\lambda|, |x_\lambda|, T_\lambda \leq B(\lambda)$*, the following ensembles are* $(\lambda_0, S'(\lambda))$ *indistinguishable with* $S'(\lambda) = S(\lambda) - B(\lambda)^d$ *for all* $\lambda \in N$*.*

$$\left\{(\hat{\Pi}, \hat{x}) \overset{\$}{\leftarrow} \mathsf{Enc}(1^\lambda, \Pi, x, T) : \hat{\Pi}, \hat{x}\right\}$$
$$\left\{(\hat{\Pi}, \hat{x}) \overset{\$}{\leftarrow} \mathsf{Sim}(1^\lambda, \Pi^T(x), 1^{|\Pi|}, 1^{|x|}, T) : \hat{\Pi}, \hat{x}\right\}$$

where $\Pi = \Pi_\lambda$*,* $x = x_\lambda$*, and* $T = T_\lambda$*.*

A recent line of works [BGL$^+$15, CHJV14, KLW14] constructed randomized encoding with polynomial simulation security (i.e., the simulation is polynomially indistinguishable to the honest encoding in the above definition) for the class of *Boolean* Turing machines, where the time complexity of encoding is independent of the run-time of the Turing machine.

Theorem 5 (Simulation-Based Randomized Encoding for Boolean Turing Machines [KLW14]). *Assuming the existence of indistinguishability obfuscation for circuits and injective pseudo-random generators, there is a randomized encoding scheme* $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ *satisfying polynomial simulation security for the class of* Boolean Turing machines*, with the following efficiency:*

- *For every security parameter* λ*, Boolean Turing machine* Π*, input* x*, time bound* T *and every encoded pair* $(\hat{\Pi}, \hat{x}) \leftarrow \mathsf{Enc}(1^\lambda, \Pi, x, T)$*, we have that*

$$\mathsf{Time}_{\mathsf{Enc}}(1^\lambda, \Pi, x, T) = \mathrm{poly}(\lambda, |\Pi|, |x|, \log T)$$
$$\mathsf{Time}_{\mathsf{Eval}}(\hat{\Pi}, \hat{x}) = \mathrm{poly}(\lambda, |\Pi|, |x|, T)$$

In this paper, we consider the class of *all* Turing machines, including ones with arbitrarily long outputs. One can obtain a randomized encoding for Turing machines with ℓ-bit outputs, by encoding a collection of ℓ Turning machines each outputting one output bit, using a randomized encoding for Boolean Turing machines. It yields a scheme whose encoding time, as well as the size of encoding, scales linearly with the output length (and still poly-logarithmically with the run-time of the encoded computation). As we show later in the paper, this essentially is tight, meaning that there is certain computation (namely, evaluating pseudo-random generators) for which the size of the randomized encoding cannot be sub-linear in the output length. In other words, when simulation security is required, encoding has to be as long as the output length in general.

3.2 Distributional Indistinguishability Security

In this paper, we study randomized encoding for all Turing machine computation, whose encoding size is independent of the output length of the computation—we say such randomized encoding schemes are **compact**. Towards this, we must consider weaker security notions than simulation security, and indistinguishability-based security notions are natural candidates. One weaker notion that has been considered in the literature requires encoding of two computation, (Π_1, x_1) and (Π_2, x_2) with the same output $\Pi_1(x_1) = \Pi_2(x_2)$, to be indistinguishable. In this work, we generalize this notion to, what called *distributional* indistinguishability security—this notion requires encoding of computations sampled from two distributions, $(\Pi_1, x_1) \xleftarrow{\$} D_1$ and $(\Pi_2, x_2) \xleftarrow{\$} D_2$, to be indistinguishable, provided that their outputs are indistinguishable.

Definition 13 (Distributional $(\lambda_0, S(\cdot))$-Indistinguishability Security)**.** *A randomized encoding scheme* RE *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *satisfies* distributional $(\lambda_0, S(\cdot))$-indistinguishability security, *(or $(\lambda_0, S(\cdot))$-ind-security for short) if the following is true w.r.t. some constant $c > 0$:*

For every ensembles of distributions $\{D_{0,\lambda}\}$ and $\{D_{1,\lambda}\}$ with the following property:

1. *there exists a polynomial B, such that, for every $b \in \{0,1\}$, $D_{b,\lambda}$ is a distribution over tuples of the form (Π_b, x_b, T_b), where Π_b is a Turing machine, x_b is an input and T_b is a time bound, and $\lambda, |\Pi_b|, |x_b|, T_b \leq B(\lambda)$.*

2. *there exist an integer $\lambda_0' \geq \lambda_0$, and a function S' with $S'(\lambda) \leq S(\lambda)$ for all λ, such that, the following ensembles of output distributions are $(\lambda_0', S'(\cdot))$-indistinguishable,*

$$\left\{(\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \Pi_0^{T_0}(x_0), T_0, |\Pi_0|, |x_0|\right\}$$
$$\left\{(\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \Pi_1^{T_1}(x_1), T_1, |\Pi_1|, |x_1|\right\}$$

the following ensembles of encoding is $(\lambda_0', S''(\cdot))$-indistinguishable, where $S''(\lambda) = \frac{S'(\lambda)}{\lambda^c} - B(\lambda)^c$.

$$\left\{(\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \mathsf{Enc}(1^\lambda, \Pi_0, x_0, T_0)\right\}$$
$$\left\{(\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Enc}(1^\lambda, \Pi_1, x_1, T_1)\right\}$$

For convenience, in the rest of the paper, we directly refer to distributional indistinguishability security as indistinguishability security. The above concrete security directly gives the standard polynomial and sub-exponential security.

Definition 14 (Polynomial and Sub-exponential Indistinguishability Security)**.** *A randomized encoding scheme* RE *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *satisfies* polynomial ind-security *, if it satisfies* $(\lambda_p, p(\cdot))$*-indistinguishability security for every polynomial* p *and some* $\lambda_p \in N$*. Furthermore, it satisfies sub-exponential ind-security if it satisfies* $(\lambda_0, S(\cdot))$*-indistinguishability security for* $S(\lambda) = 2^{\lambda^\varepsilon}$ *with some* $\varepsilon \in (0,1)$*.*

We note that, by definition, it holds that any randomized encoding scheme that is $(\lambda_0, S(\cdot))$-ind-secure, is also $(\lambda_0', S'(\cdot))$-ind-secure for any $\lambda_0' \geq \lambda_0$ and S' s.t. $S'(\lambda) \leq S(\lambda)$ for every λ. Therefore, naturally, sub-exponential ind-security is stronger than polynomial ind-security.

Later, in Section 3.4 we show that RE schemes with ind-security are composable just as RE schemes with simulation security are. Additionally, in Section 3.5, we show that RE schemes satisfying simulation security also satisfy ind-security.

3.3 Compactness and Sublinear Compactness

With indistinguishability-security, we now define compact randomized encoding schemes for all Turing machines, whose time-complexity of encoding is independent of the output length.

Definition 15 (Compact Randomized Encoding for Turing machines)**.** *A* $(\lambda_0, S(\cdot))$*-ind-secure* compact randomized encoding *scheme for Turing machines, is a randomized encoding scheme with* $(\lambda_0, S(\cdot))$*-indistinguishability security for the class of all Turing machines, with the following efficiency:*

- *For every security parameter* λ*, Turing machine* Π*, input* x*, time bound* T *and every encoded pair* $(\hat{\Pi}, \hat{x}) \leftarrow \mathsf{Enc}(1^\lambda, \Pi, x, T)$*, it holds*

$$\mathsf{Time}_{\mathsf{Enc}}(1^\lambda, \Pi, x, T) = \text{poly}(\lambda, |\Pi|, |x|, \log T)$$
$$\mathsf{Time}_{\mathsf{Eval}}(\hat{\Pi}, \hat{x}) = \text{poly}(\lambda, |\Pi|, |x|, T)$$

In this work, we also consider a weaker variant of the above compactness requirement, where the encoding time is sub-linear (instead of poly-logarithmic) in the computation time. For our results a compact randomized encoding scheme with sub-linear efficiency will suffice.

Definition 16 (Sub-linear Compactness of Randomized Encoding schemes)**.** *We say a randomized encoding scheme* $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *has sub-linear compactness if the efficiency requirement on* Enc *in Definition 15 is relaxed to: For some constant* $\varepsilon \in (0,1)$*,*

$$\mathsf{Time}_{\mathsf{Enc}}(1^\lambda, \Pi, x, T) \leq \text{poly}(\lambda, |\Pi|, |x|) \cdot T^{1-\epsilon}$$

3.4 Composition of Ind-Security

It was shown in [IK02b, AIK06] that randomized encoding schemes with simulation security are composable. We show that the same holds for indistinguishability security. Let $RE_1 = (\mathsf{Enc}_1, \mathsf{Eval}_1)$ be a randomized encoding scheme for a class of Turing machines $\{\mathcal{M}_\lambda\}$, $\mathsf{RE}_2 = (\mathsf{Enc}_2, \mathsf{Eval}_2)$ a randomized encoding for an appropriate class of Turing machines that includes the ensembles of functions $\{G[\lambda, \Pi, T, s]\}$ defined below, and PRG a pseudo-random generator. Consider the following composed randomized encoding scheme $(\mathsf{Enc}, \mathsf{Eval})$:

$$\begin{aligned}
\mathsf{Enc}(1^\lambda, \Pi, x, T) \quad &: \quad (\hat{G}, \hat{x}) \xleftarrow{\$} \mathsf{Enc}_2(1^\lambda, G, x, T_G(x)) \\
&\quad\;\; \text{where } G[\lambda, \Pi, T, s](x) = \mathsf{Enc}_1(1^\lambda, \Pi, x, T\ ;\ \mathsf{PRG}(s)) \\
\mathsf{Eval}(\hat{G}, \hat{x}) \quad &: \quad (\hat{\Pi}, \hat{x}') = \mathsf{Eval}_2(\hat{G}, \hat{x}),\ y = \mathsf{Eval}_1(\hat{\Pi}, \hat{x}')
\end{aligned}$$

where $T_G(x)$ is an upper bound on the run-time of G on input x, it can be efficiently calculated using an upper bound on the run-time of Enc_1 and PRG, in particular, $T_G(x) = \text{poly}(\mathsf{Time}_{\mathsf{Enc}_1}(1^\lambda, \Pi, x, T))$.

Lemma 1 (Composition)**.** *If* RE_1 *is* $(\lambda_1, S_1(\cdot))$*-ind-secure, and* RE_2 *and* PRG *are* $(\lambda_2, S_2(\cdot))$*-ind-secure, with* $\lambda_2 \leq \lambda_1$ *and* $S_2(\lambda) \geq S_1(\lambda)$*. then* $(\mathsf{Enc}, \mathsf{Eval})$ *is* $(\lambda_1, S_1(\cdot))$*-ind-secure.*

Proof. Let c_1 be the constant w.r.t. which the ind-security of RE_1 holds, and c_2 the constant for RE_2. We show that the composed scheme $(\mathsf{Enc}, \mathsf{Eval})$ is $(\lambda_1, S_1(\cdot))$-ind-secure w.r.t. some sufficiently large constant c, whose value would become clear in the proof below.

To show this, consider any pair of ensembles of distributions $\{D_{0,\lambda}\}$ and $\{D_{1,\lambda}\}$ that are (λ_0', S')-indistinguishability for $\lambda_0' \geq \lambda_1$ and $S'(\lambda) \leq S_1(\lambda)$ for all λ, and all parameters $\lambda, |\Pi_b|, |x_b|, T_b \leq B(\lambda)$ for some polynomial B. We need to show that encodings generated using Enc are (λ_0', S'')-indistinguishable, where $S''(\lambda) \leq S'(\lambda)/\lambda^c - B(\lambda)^c$.

First, it follows directly from the indistinguishability security of RE_1 that the distributions H_1 and H_2 of encodings produced by Enc_1 are (λ_0', S_1)-indistinguishable, where $S_H(\lambda) = S'(\lambda)/\lambda^{c_1} - B(\lambda)^{c_1}$,

$$H_1 = \left\{(\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \mathsf{Enc}_1(1^\lambda, \Pi_0, x_0, T_0)\right\}$$
$$H_2 = \left\{(\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Enc}_1(1^\lambda, \Pi_1, x_1, T_1)\right\}$$

Consider two modified distributions, where the encodings are generated using pseudo-random coins.

$$H_1^+ = \left\{s \xleftarrow{\$} \{0,1\}^\lambda,\ (\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \mathsf{Enc}_1(1^\lambda, \Pi_0, x_0, T_0; \mathsf{PRG}(s))\right\}$$
$$H_2^+ = \left\{s \xleftarrow{\$} \{0,1\}^\lambda,\ (\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Enc}_1(1^\lambda, \Pi_1, x_1, T_1; \mathsf{PRG}(s))\right\}$$

Since $\lambda, |\Pi_b|, |x_b|, T_b \leq B(\lambda)$, $\mathsf{Time}_{\mathsf{Enc}_1}(1^\lambda, \Pi_b, x_b, T_b)$ is bounded by $B(\lambda)^d$ for some constant d. Then, by the (λ_2, S_2)-indistinguishability of PRG, it follows that for every b no adversary of size $\tilde{S}_H(\lambda) = S_2(\lambda) - B(\lambda)^d$ can distinguish H_b^+ and H_b with probability larger than $1/S_2(\lambda)$. Therefore, (by a hybrid argument,) no adversary of size $\min(S_H(\lambda), \tilde{S}_H(\lambda))$ can distinguish H_1^+ and H_2^+ with probability larger than $2/\tilde{S}_H(\lambda) + 1/S_H(\lambda)$. Since $S'(\lambda) \leq S_1(\lambda) \leq S_2(\lambda)$, there is a $S_H^+(\lambda) = S'(\lambda)/\lambda^e - B(\lambda)^e$ with a sufficiently large constant e, such that, $\{H_1^+\}$ and $\{H_2^+\}$ are (λ_0', S_H^+)-indistinguishable.

Consider the following ensembles of distributions $\{\mathcal{E}_{0,\lambda}\}$ and $\{\mathcal{E}_{1,\lambda}\}$:

$$\begin{aligned}\mathcal{E}_{b,\lambda} \ : \ & s \xleftarrow{\$} \{0,1\}^\lambda,\ (\Pi_b, x_b, T_b) \xleftarrow{\$} \mathcal{D}_{b,\lambda} \text{ output } (G_b, x_b, T_b') \\ & \text{where } G_b[\lambda, \Pi_b, T_b, s](x_b) = \mathsf{Enc}_1(1^\lambda, \Pi_b, x_b, T_b\ ;\ \mathsf{PRG}(s)),\ T_b' = T_{G_b}(x_b).\end{aligned}$$

where $T_{G_b}(x_b) = \text{poly}(\lambda, |\Pi_b|, |x_b|, \log T_b)$, determined by the run time of algorithms PRG and Enc_1. By the indistinguishability of H_1^+ and H_2^+, the output distributions of $\mathcal{E}_{0,\lambda}$ and $\mathcal{E}_{1,\lambda}$ are also (λ_0', S_H^+)-indistinguishable. Moreover, all parameters T_b', $|G_b|$, $|x_b|$ are bounded by $B(\lambda)^k$ for some constant k.

Therefore, it follows from the (λ_2, S_2)-security of RE_2 (and the fact that $\lambda_2 \leq \lambda_1 \leq \lambda_0'$, and $S_H^+(\lambda) \leq S_2(\lambda)$) that, encoding of G_0 and G_1 sampled from $\mathcal{E}_{0,\lambda}$ and $\mathcal{E}_{1,\lambda}$ are (λ_0', S_3'')-indistinguishable, where $S''(\lambda) = S_H^+(\lambda)/\lambda^{c_2} - B(\lambda)^{kc_2} \geq S'(\lambda)/\lambda^c - B(\lambda)^c$ for a sufficiently large c.

$$\left\{(G_0, x_0, T_0') \xleftarrow{\$} \mathcal{E}_{0,\lambda} : \mathsf{Enc}_2(1^\lambda, G_0, x_0, T_0')\right\}$$
$$\left\{(G_1, x_1, T_1') \xleftarrow{\$} \mathcal{E}_{1,\lambda} : \mathsf{Enc}_2(1^\lambda, G_1, x_1, T_1')\right\}$$

This concludes the proof. $\square$

The above composition lemma implies that if there is a sublinear RE with complexity scaling with T^{β}, one can reduce the complexity by an arbitrary polynomial factor by recursively composing the RE with itself. This fact with be very instrumental later. More precisely,

Claim 1. *Let α and β be any constants satisfying $0 < \alpha < \beta < 1$. If there is a sublinear RE* $(\mathsf{Enc}', \mathsf{Eval}')$ *with time complexity*

$$\mathsf{Time}_{\mathsf{Enc}'}(1^{\lambda}, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\beta} \ ,$$

then, there is a sublinear RE $(\mathsf{Enc}, \mathsf{Eval})$ *with time complexity*

$$\mathsf{Time}_{\mathsf{Enc}}(1^{\lambda}, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\alpha} \ .$$

Proof of Claim 1. Given any sublinear RE $\mathsf{RE}' = (\mathsf{Enc}', \mathsf{Eval}')$ with time complexity

$$\mathsf{Time}_{\mathsf{Enc}'}(1^{\lambda}, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\beta}$$

We show how to construct a sublinear RE $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ with time complexity

$$\mathsf{Time}_{\mathsf{Enc}}(1^{\lambda}, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\alpha}$$

The new scheme RE is constructed by "composing" RE' iteratively for a sufficiently large, constant, number of times, depending on α and β.
<u>Composing Once:</u> Recall that in Section 3.4, it was shown that given two ind-secure RE schemes for Turing machines, one can compose them into a new RE scheme that still satisfies indistinguishability security, as shown in Lemma 1. It follows from this lemma that by composing the scheme RE' with itself, we obtain a new scheme RE_1 as follows.

$$G[\lambda, \Pi, T, s](x) = \mathsf{Enc}'(1^{\lambda}, \Pi, x, T \ ; \ \mathsf{PRG}(s))$$
$$\mathsf{Enc}_1(1^{\lambda}, \Pi, x, T) \ : \ (\hat{G}, \hat{x}) \xleftarrow{\$} \mathsf{Enc}'(1^{\lambda}, G, x, T_G(x))$$

where $T_G(x)$ is an upper bound on the run-time of $G(x)$. We show that RE_1 is more "compact" than RE' – with a time complexity depending on T^{β^2} as opposed to T^{β}.

$$T_G(x) = T_{\mathsf{Enc}'}(1^{\lambda}, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\beta}$$
$$T_{\mathsf{Enc}_1}(1^{\lambda}, \Pi, x, T) = T_{\mathsf{Enc}'}(1^{\lambda}, G, x, T_G(x)) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\beta^2}$$

<u>Composing k times:</u> Since RE_1 itself is a sublinear RE scheme, one can apply the same composition technique on it to obtain a new scheme RE_2 whose time complexity depends on $T^{(\beta^2)^2}$. More generally, applying the composition technique recursively for a constant number d of times yields a scheme RE_d with time complexity

$$T_{\mathsf{Enc}_d}(1^{\lambda}, \Pi, x, T) = \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\beta^e}, \quad \text{for } e = 2^d \ .$$

For a sufficiently large d, $T_{\mathsf{Enc}_d} \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^{\alpha}$, which concludes the claim. $\square$

3.5 Simulation Security implies Indistinguishability Security

In this section, we show that simulation security for a randomized encoding scheme implies indistinguishability security for the same scheme. More formally,

Theorem 6. *Let* RE *be a* (λ_0, S)*-simulation-secure randomized encoding scheme for a sufficiently large* λ_0*. Then* RE *is also* (λ_0, S)*-ind-secure.*

Proof. Let Sim be the PPT simulator of $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$; let c be a sufficiently large constant whose value will become clear in the proof below. To show that RE is (λ_0, S)-ind-secure w.r.t. constant c, consider arbitrary ensembles of distributions $\{D_{0,\lambda}\}$ and $\{D_{1,\lambda}\}$ whose output distributions below are (λ_0', S')-indistinguishability for $\lambda_0' \geq \lambda_0$ and $S'(\lambda) \leq S(\lambda)$ for all λ,

$$\mathcal{O}_0 = \left((\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \Pi_0^{T_0}(x_0), T_0, |\Pi_0|, |x_0|\right)$$
$$\mathcal{O}_1 = \left((\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \Pi_1^{T_1}(x_1), T_1, |\Pi_1|, |x_1|\right)$$

and all parameters $\lambda, |\Pi_b|, |x_b|, T_b \leq B(\lambda)$ for some polynomial B. We show that encoding generated using Enc are (λ_0', S'')-indistinguishable, where $S''(\lambda) \leq S'(\lambda)/\lambda^c - B(\lambda)^c$.

Consider the following sequence of hybrids:

H_0:
$$\left((\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{0,\lambda} : \mathsf{Enc}(1^\lambda, \Pi_0, x_0, T_0)\right)$$

H_1:
$$\left((\Pi_0, x_0, T_0) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Sim}(1^\lambda, \Pi_0^{T_0}(x_0), 1^{|\Pi_0|}, 1^{|x_0|}, T_0)\right)$$

H_2:
$$\left((\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Sim}(1^\lambda, \Pi_1^{T_1}(x_1), 1^{|\Pi_1|}, 1^{|x_1|}, T_1)\right)$$

H_3:
$$\left((\Pi_1, x_1, T_1) \xleftarrow{\$} \mathcal{D}_{1,\lambda} : \mathsf{Enc}(1^\lambda, \Pi_1, x_1, T_1)\right)$$

In other words, our goal is to show that $\{H_0\}$ and $\{H_3\}$ are (λ_0', S'')-indistinguishable. Towards this, we argue the indistinguishability of every two neighboring hybrids and use a hybrid argument to conclude.

First, it follows directly from the (λ_0, S)-simulation security of RE (and the fact that $\lambda_0' \geq \lambda_0$) that $\{H_0\}$ and $\{H_1\}$, as well as $\{H_2\}$ and $\{H_3\}$, are $(\lambda_0', S_{0,1})$-indistinguishable with $S_{0,1}(\lambda) = S(\lambda) - B(\lambda)^d$ for some constant d.

Next, to argue the indistinguishability of H_1 and H_2, we first note that the run-time of the simulator $T_{\mathsf{Sim}} = \mathsf{Time}_{\mathsf{Sim}}(1^\lambda, \Pi_b^{T_b}(x_b), 1^{|\Pi_b|}, 1^{|x_b|}, T_b) \leq B(\lambda)^c$ with a sufficiently large constant c determined by Sim, since the length of every input argument is bounded by $B(\lambda)$. Then, it follows from the (λ_0', S')-indistinguishability of the output distributions $\{O_0\}$, $\{O_1\}$ that for every adversary of size $S_{1,2}(\lambda) = S'(\lambda) - T_{\mathsf{Sim}} = S'(\lambda) - B(\lambda)^c$, the probability that it distinguishes H_1 and H_2 is bounded by $1/S'(\lambda)$ (as otherwise one can construct a $S'(\lambda)$-size adversary that distinguishes $\{O_1\}$ and $\{O_2\}$ with probability larger than $1/S'(\lambda)$, by internally running Sim to sample from H_1 or H_2, and then the distinguisher for H_1 and H_2 to distinguish).

Therefore, it follows from a hybrid argument that for every adversary of size $\min(S_{0,1}(\lambda), S_{1,2}(\lambda))$, the probability of distinguishing H_0 and H_3 is bounded by $2/S_{0,1}(\lambda) + 1/S'(\lambda)$. For sufficiently large $c > d$ and λ_0', $\{H_0\}$ and $\{H_3\}$ are (λ_0', S'')-indistinguishable. $\square$

4 Unbounded-Input IO from Compact RE

In this section, we define our succinct indistinguishability obfuscator for Turing machines. Let $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ be a compact randomized encoding scheme for Turing machines with sub-exponential indistinguishability security. Let c be the constant for the security loss associated with the indistinguishability security of RE. We assume without loss of generality that $\mathsf{Enc}(1^\lambda, \cdot, \cdot)$ requires a random tape of length λ. Let PRG be a sub-exponentially secure pseudorandom generator and let ϵ be the constant associated with the sub-exponential security of PRG.

For every $\lambda \in \mathbb{N}$, $D \leq 2^\lambda$, define

$$l(\lambda, -1) = \lambda$$
$$l(\lambda, D) = l(\lambda, D-1) + (2d\lambda)^{1/\epsilon}$$

where $d > 0$ is any constant strictly greater than c.

Construction 1. *Consider a Turing machine Π, security parameter $\lambda \in \mathbb{N}$, and time bound T of Π. For every partial input $s \in \{0,1\}^*$ with $|s| \leq 2^\lambda$ and $R \in \{0,1\}^{2l(\lambda,|s|)}$, we recursively define a Turing machine $\widetilde{\Pi}_{s,R}$ to be as follows:*

When $|s| < 2^\lambda$:

On the empty input, $\widetilde{\Pi}_{s,R}$ outputs:

$$\begin{aligned}
&\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}_{s0,R_0}, T'(\lambda, |s|+1, |\Pi|, \log(T)); R_1)\\
&\mathsf{Enc}(1^{l(\lambda,|s|+2)}, \widetilde{\Pi}_{s1,R_2}, T'(\lambda, |s|+1, |\Pi|, \log(T)); R_3)\\
&\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi, s, T; R_4)
\end{aligned}$$

where $(R_0, R_1, R_2, R_3, R_4) \leftarrow \mathsf{PRG}(R, 5 \cdot 2l(\lambda, |s|+1))$ and T' is some fixed polynomial in $\lambda, |s|+1, |\Pi|$ and $\log(T)$. In the special case when $|s| = 2^\lambda - 1$, the time bound used in the first two encodings is set to T.

On all other inputs, $\widetilde{\Pi}_{s,R}$ outputs $\perp$.

When $|s| = 2^\lambda$:

On the empty input, $\widetilde{\Pi}_{s,R}$ outputs $\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi, s, T; R)$. On all other inputs, $\widetilde{\Pi}_{s,R}$ outputs $\perp$.

We define $T'(\cdot,\cdot,\cdot,\cdot)$ (corresponding to the bound placed on the running time of $\widetilde{\Pi}_{s,R}$) to be the smallest polynomial such that for all λ, $s \in \{0,1\}^{\leq 2^\lambda}$, $R \in \{0,1\}^{2l(\lambda,|s|)}$, Π and T,

$$\begin{aligned}
T'(\lambda, |s|, |\Pi|, \log(T)) \geq\ & p(\lambda_{|s|+1}, |\widetilde{\Pi}_{s0,R}|, 0, \log(T'_{|s|+1}))\\
&+ p(\lambda_{|s|+1}, |\widetilde{\Pi}_{s1,R}|, 0, \log(T'_{|s|+1}))\\
&+ p(\lambda_{|s|+1}, |\Pi|, |s|, \log(T))\\
&+ \mathsf{Time}_{\mathsf{PRG}}(R, 5 \cdot 2l(\lambda, |s|+1))
\end{aligned}$$

where $\lambda_{|s|+1} = l(\lambda, |s|+1)$, $T'_{|s|+1} = T'(\lambda, |s|+1, |\Pi|, \log(T))$ (corresponding to the security parameter and time bound used for each of $\widetilde{\Pi}_{s0,R_0}$ and $\widetilde{\Pi}_{s1,R_1}$), $\mathsf{Time}_{\mathsf{PRG}}$ is the bound on the running time of the PRG, and $p(\cdot,\cdot,\cdot,\cdot)$ is the bound on $\mathsf{Time}_{\mathsf{Enc}}$ from the compactness of RE. We note that the polynomial T' exists because p is a polynomial, each of $\lambda_{|s|+1}$ and $|\widetilde{\Pi}_{s,R}|$ are of size polynomial in $\lambda, |s|$ and $|\Pi|$, and the self-dependence of $T'(\lambda, |s|, |\Pi|, \log(T))$ on $T'_{|s|+1}$ is only poly-logarithmic.

Remark: *We note that $|\widetilde{\Pi}_{s,R}|$ is always poly$(\lambda, |\Pi|, |s|, \log(T))$. This is because $\widetilde{\Pi}_{s,R}$ is fully described by λ, Π, s, R and T, and the size of each of these is bounded by poly$(\lambda, |\Pi|, |s|, \log(T))$.*

Given this definition of $\widetilde{\Pi}_{s,R}$, we define our indistinguishability obfuscator as follows:

Construction 2 (Indistinguishability Obfuscator)**.** *On input $\lambda \in \mathbb{N}$, Turing machine Π and time bound T, define $\widetilde{\Pi}$, the indistinguishability obfuscation of Π, to be*

$$\widetilde{\Pi} = \boldsymbol{iO}(1^{\lambda}, \Pi, T) = \mathsf{Enc}(1^{l(\lambda,0)}, \widetilde{\Pi}_{\epsilon,R}, T'(\lambda, 0, |\Pi|, \log(T)))$$

Where ϵ is the empty string, and $R \xleftarrow{\$} \{0,1\}^{2l(\lambda,0)}$ and T' a fixed polynomial in $\lambda, |\Pi|$ and $\log(T)$, as described above.

Evaluation: The algorithm to evaluate $\widetilde{\Pi}$ on input $x \in \{0,1\}^d, d < 2^{\lambda}$ proceeds as follows:

1. For every $0 \leq i \leq d$, compute encodings of $\widetilde{\Pi}_{x_{\leq i},R}$ successively, starting with $\widetilde{\Pi}$, an encoding of $\widetilde{\Pi}_{\epsilon,R}$, and subsequently, for every $0 < i \leq d$, computing the encoding of $\widetilde{\Pi}_{x_{\leq i},R}$ by evaluating the encoding of $\widetilde{\Pi}_{x_{<i},R}$, and selecting the encoding of $\widetilde{\Pi}_{x_{\leq i},R}$ from its output.
2. Evaluate the encoding of $\widetilde{\Pi}_{x,R} = \widetilde{\Pi}_{x_{\leq d},R}$ and obtain from its output $(\hat{\Pi}, \hat{x}) = \mathsf{Enc}(1^{l(\lambda,|x|+1)}, \Pi, x, T; R_4)$.
3. Run $\mathsf{Eval}(\hat{\Pi}, \hat{x})$ to obtain $\Pi(x)$.

To analyze the correctness, running time, and compactness of our **iO** construction, we make use of the following lemma:

Lemma 2. *Let Π be a polynomial-time TM, $\lambda \in \mathbb{N}$ be a security parameter, and $T \leq 2^{\lambda}$ be a running time bound. Then, for every $s \in \{0,1\}^*$ with $|s| \leq 2^{\lambda}$ and every $R \in \{0,1\}^{2l(\lambda,|s|)}$, the running time of $\widetilde{\Pi}_{s,R}$ is bounded by $T'(\lambda, |s|, |\Pi|, \log(T))$.*

Proof. We prove the lemma by fixing a $\lambda \in \mathbb{N}$, and inducting on the size of s.

Base case: $|s| = 2^{\lambda}$**.** In this case, the running time T'_s of $\widetilde{\Pi}_{s,R}$ is given by

$$\begin{aligned} T'_s &= \mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,|s|+1)}, \Pi, s, T) \\ &\leq p(l(\lambda, |s|+1), |\Pi|, |s|, \log(T)) \\ &\leq T'(\lambda, |s|, |\Pi|, \log(T)) \end{aligned}$$

This completes the base case.

Inductive step: $|s| < 2^{\lambda}$ By the induction hypothesis, we assume that the lemma holds for all s' with $|s'| = |s| + 1$, in particular, for $s' = s0$ and $s' = s1$.

Then, an execution of $\widetilde{\Pi}_{s,R}$ runs a single evaluation of a PRG, and produces encodings of each of $\widetilde{\Pi}_{s0,R_0}$, $\widetilde{\Pi}_{s1,R_2}$ and (Π, s). The *PRG* runs in time $\mathsf{Time}_{\mathsf{PRG}}(R, 5 \cdot 2l(\lambda, |s|+1))$, while the encoding (Π, s) takes time $\mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,|s|+1)}, \Pi, s, T)$. Further, by the inductive hypothesis, the encodings of $\widetilde{\Pi}_{s0,R_0}$ and $\widetilde{\Pi}_{s1,R_2}$ each take time $\leq T'(\lambda, |s|+1, |\Pi|, \log(T)$. Combining these facts together, we have that the running time T'_s of $\widetilde{\Pi}_{s,R}$ is given by:

$$\begin{aligned}T'_s = & \, p(\lambda_{|s|+1}, |\widetilde{\Pi}_{s0,R}|, 0, \log(T'_{|s|+1})) \\ & + p(\lambda_{|s|+1}, |\widetilde{\Pi}_{s1,R}|, 0, \log(T'_{|s|+1})) \\ & + p(\lambda_{|s|+1}, |\Pi|, |s|, \log(T)) \\ & + \mathsf{Time}_{\mathsf{PRG}}(R, 5 \cdot 2l(\lambda, |s|+1)) \\ & \leq T'(\lambda, |s|, |\Pi|, \log(T))\end{aligned}$$

This concludes the inductive step, and the lemma follows. □

Correctness of iO: The correctness of **iO** applied to any polynomial-length x follows from the correctness of evaluating encodings RE, applied at every level of the evaluation. More concretely, for each index $i \leq |x|$ of the evaluation, except with probability $\mu(l(\lambda, i))$, the evaluation of $\widetilde{\Pi}_{x_{<i},R}$ correctly produces $\widetilde{\Pi}_{x_{\leq i},R}$. Further, the final evaluation of $\hat{\Pi}, \hat{x}$ produces $\Pi^T(x)$ correctly except with probability $\mu(l(\lambda, i+1))$.

Crucially, the correctness at each step relies on the fact that each encoding $\widetilde{\Pi}_{x_{<i},R}$ uses time bound $T'(\lambda, i, |\Pi|, \log(T))$, which, as argued above, is sufficiently large to compute $\widetilde{\Pi}_{x_{\leq i},R}$ for the next level.

Overall, the probability of incorrect evaluation is $\leq \sum_{i=1}^{|x|} \mu(\lambda, i)$, which is negligible for any polynomial-length x.

Running time of iO: Again, considering step i, the evaluation at this step takes time $p(l(\lambda, i)$, $|\Pi_{x_{<i},R}|, 0, T'(\lambda, i, |\Pi|, \log(T))$, which is $\text{poly}(\lambda, i, |\Pi|, \log(T))$. Further, the evaluation of $\hat{\Pi}, \hat{x}$ is $p(l(\lambda, |x|+1), |x|, |\Pi|, T)$, which is $\text{poly}(\lambda, |x|, |\Pi|, T)$. Therefore, overall the running time is $\text{poly}(\lambda, |x|, |\Pi|, T)$.

Efficiency of iO: The size of $\mathbf{iO}(1^\lambda, \Pi, T)$ is the same as the size of $\widetilde{\Pi}_{\epsilon,R}$, which itself is bounded by $\mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,0)}, \Pi_{\epsilon,R}, T'(\lambda, 0, |\Pi|, \log(T)))$, which is $\text{poly}(\lambda, |\Pi|, \log(T))$ by the efficiency of RE.

4.1 Security Proof

Theorem 7. *Let* (Enc, Eval) *be a sub-exponentially-indistinguishability-secure, compact randomized encoding scheme and let* PRG *be a sub-exponentially-secure pseudorandom generator. Then the indistinguishability obfuscator defined in Construction 2 is subexponentially-secure.*

Proof. Consider any pair of ensembles of Turing machines and time bounds $\left\{\Pi^0_\lambda, \Pi^1_\lambda, T_\lambda\right\}$ where for every $\lambda \in \mathbb{N}$, $\Pi^0 = \Pi^0_\lambda$, $\Pi^1 = \Pi^1_\lambda$, $T = T_\lambda$,

$$|\Pi^0| = |\Pi^1| \leq \text{poly}(\lambda) \quad |T| \leq \text{poly}(\lambda)$$
$$\forall x, \Pi^{0,T}(x) = \Pi^{1,T}(x)$$

We first introduce some notation to describe the distributions of randomized encodings generated by $i\mathcal{O}(1^\lambda, \Pi^0_\lambda, T_\lambda)$ and $i\mathcal{O}(1^\lambda, \Pi^1_\lambda, T_\lambda)$. For $\lambda \in \mathbb{N}$, $s \in \{0,1\}^*$, $|s| \leq 2^\lambda$, we define the following distributions

$$D_{\lambda,0,s} = \mathsf{Enc}(1^{l(\lambda,|s|)}, \widetilde{\Pi}^0_{s,R}, T')$$
$$D_{\lambda,1,s} = \mathsf{Enc}(1^{l(\lambda,|s|)}, \widetilde{\Pi}^1_{s,R}, T')$$

where R is uniformly random, T' is as described in Construction 1 and $\widetilde{\Pi}^b_{s,R}$ is defined for the Turing machine Π^b_λ, security parameter λ and time bound T_λ. We will show something stronger than the theorem statement. In particular, we have the following claim.

Claim 2. *There exists $\lambda_0, \epsilon \in \mathbb{N}$ such that for every $\lambda > \lambda_0$, for every $s \in \{0,1\}^*, |s| \leq 2^\lambda$ we have that the distributions $D_{\lambda,0,s}$ and $D_{\lambda,1,s}$ are $S(\lambda)$ indistinguishable where $S(\lambda) \geq 10 \cdot 2^{l(\lambda,|s|-1)^\epsilon}$.*

Using the above claim with s as the empty string and recalling $l(\lambda, 0) = \lambda$, the theorem statement follows. Therefore, in the remainder of the proof, we prove the above claim.

Proof of claim Let ϵ be the larger of the constants associated with the sub-exponential security of the pseudorandom generator PRG and the indistinguishability security of the encoding scheme $(\mathsf{Enc}, \mathsf{Eval})$ (these constants are also named ϵ in their respective security definitions). Similarly, We consider λ_0 to be large enough so that the security of the encoding scheme $(\mathsf{Enc}, \mathsf{Eval})$ and the pseudorandom generator PRG is applicable. We will actually require a larger λ_0 so that certain asymptotic conditions (depending only on the polynomial size bounds of Π^0_λ, Π^1_λ and T_λ) hold, which we make explicit in the remainder of the proof. For every $\lambda > \lambda_0$, we prove the claim by induction on $|s|$. Our base case will be when $|s| = 2^\lambda$ and in the inductive step we show the claim holds for all s of a particular length d, if it holds for all s of length $d+1$.

Induction statement, for a fixed $\lambda > \lambda_0$: For every $s \in \{0,1\}^{\leq 2^\lambda}$, the distributions $D_{\lambda,0,s}$ and $D_{\lambda,1,s}$ are $10 \cdot 2^{l(\lambda,|s|-1)^\epsilon}$ indistinguishable.

Base case: $|s| = 2^\lambda$. In this case, recall that the output of $\widetilde{\Pi}^b_{s,R}$ is simply $(\hat{\Pi}^{b,T}_\lambda, \hat{s})$. We first claim that, for all s, $(\hat{\Pi}^{0,T}_\lambda \hat{s})$ and $(\hat{\Pi}^{1,T}_\lambda, \hat{s})$ are $2^{\lambda'^\epsilon}$-indistinguishable where $\lambda' = l(\lambda, |s|)$, as follows.

Recall that the output of evaluating $\hat{\Pi}^{b,T}_\lambda, \hat{s}$ is simply $\Pi^{b,T}_\lambda(s)$. Since we have that $\Pi^{0,T}_\lambda(s) = \Pi^{1,T}_\lambda(s)$ for all s, we can apply the security of the randomized encoding scheme. More concretely, since the output (point) distributions are identical, they are $10 \cdot 2^{\lambda'^\epsilon}$-indistinguishable where $\lambda' = l(\lambda, |s|+1)$. Let $B(\cdot)$ be a polynomial such that $B(\lambda')$ bounds from above $|\Pi^b|, |s|$ and T. By the security of the encoding scheme, the encodings $(\hat{\Pi}^{0,T}_\lambda \hat{s})$ and $(\hat{\Pi}^{1,T}_\lambda \hat{s})$ are S' indistinguishable where

$$S' \geq \frac{10 \cdot 2^{l(\lambda,|s|+1)^\epsilon}}{l(\lambda, |s|+1)^c} - B(l(\lambda, |s|+1))^c \geq \frac{10 \cdot 2^{l(\lambda,|s|+1)^\epsilon}}{l(\lambda, |s|+1)^d} \geq 10 \cdot 2^{l(\lambda,|s|)^\epsilon}$$

where the first inequality holds for sufficiently large λ and in the second inequality, we use the fact that $l(\lambda, |s|+1) = l(\lambda, |s|) + \lambda^{d/\epsilon}$. Thus $(\hat{\Pi}^{0,T}_\lambda, \hat{s})$ and $(\hat{\Pi}^{1,T}_\lambda, \hat{s})$ are $10 \cdot 2^{l(\lambda,|s_\lambda|)^\epsilon}$-indistinguishable.

Now, recall that the output of $\widetilde{\Pi}^b_{s,R}$ is simply $(\hat{\Pi}^{b,T}_\lambda, \hat{s})$. By the above argument, we have that, for all s, $(\hat{\Pi}^{0,T}_\lambda \hat{s})$ and $(\hat{\Pi}^{1,T}_\lambda, \hat{s})$ are $2^{\lambda'^\epsilon}$-indistinguishable where $\lambda' = l(\lambda, |s|)$. Let B' be the polynomial such that $B'(l(\lambda, |s|))$ bounds $|\widetilde{\Pi}^b_{s,R}|$ and the running time of $\widetilde{\Pi}^b_{s,R}$ as per Lemma 2. The encodings $D_{\lambda,0,s}$ and $D_{\lambda,1,s}$ are S' indistinguishable where

$$S' \geq \frac{10 \cdot 2^{l(\lambda,|s|)^\epsilon}}{l(\lambda, |s|)^c} - B'(l(\lambda, |s|))^c \geq \frac{10 \cdot 2^{l(\lambda,|s|+1)^\epsilon}}{l(\lambda, |s|)^d} \geq 10 \cdot 2^{l(\lambda,|s|-1)^\epsilon}$$

where, as before, the first inequality holds for sufficiently large λ and in the second inequality, we use the fact that $l(\lambda, |s|+1) = l(\lambda, |s|) + \lambda^{d/\epsilon}$. Hence the claim holds for $|s| = 2^\lambda$.

Inductive step: $|s| < 2^{\lambda}$. By the induction hypothesis, we assume the claim holds for all s' such that $|s'| = |s| + 1$. Recall that the output of $\widetilde{\Pi}^b_{s,R}$ (where $R \xleftarrow{\$} \{0,1\}^{2l(\lambda,|s|)}$) is

$$\begin{gathered}\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}^b_{s0,R_0}, T'; R_1)\\ \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}^b_{s1,R_2}, T'; R_3)\\ \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi^b_{\lambda}, s, T; R_4)\end{gathered}$$

where $(R_0, R_1, R_2, R_3, R_4) \leftarrow \mathsf{PRG}(R, 5 \cdot 2l(\lambda, |s|+1))$. Let H^b denote the above output distribution. We will show H^0 and H^1 are indistinguishable by a hybrid argument as follows.

- Let G_1 be a hybrid distribution exactly as H^0 except that $(R_0, R_1, R_2, R_3, R_4) \xleftarrow{\$} \{0,1\}^{5 \cdot 2l(\lambda,|s|+1)}$. We claim that for both the distributions H^0 and G_1 are $5 \cdot 2^{\lambda'^{\epsilon}}$ indistinguishable where $\lambda' = l(\lambda, |s|)$.

  This follows from the PRG security as follows: any size $5 \cdot 2^{\lambda'^{\epsilon}}$ adversary A that distinguishes H^0 and G_1 can be turned into an adversary A' that can break the PRG security with seed length $2\lambda'$ with the same advantage. A' has Π^0_{λ}, Π^1_{λ}, T_{λ} and s hardcoded in it. Hence, the size of A' is
  $$5 \cdot 2^{\lambda'^{\epsilon}} + \mathrm{poly}(\lambda) + \mathrm{poly}(|s|) \leq 5 \cdot 2^{\lambda'^{\epsilon}} + \mathrm{poly}(\lambda') \leq 2^{(2\lambda')^{\epsilon}}$$
  where the last inequality holds when λ is sufficiently large. Hence, A' breaks the $2^{(2\lambda')^{\epsilon}}$-security of PRG and we have a contradiction.

  Writing out the components of G_1, we have that it is identical to
  $$G_1 \equiv D_{\lambda,0,s0}, D_{\lambda,0,s1}, \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi^0_{\lambda}, s, T_{\lambda}; R)$$

- Let G_2 be a hybrid distribution obtained by modifying the first component of G_1 as follows.
  $$G_2 \equiv D_{\lambda,1,s0}, D_{\lambda,0,s1}, \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi^0_{\lambda}, s, T_{\lambda}; R)$$
  We show that G_1 and G_2 are $5 \cdot 2^{\lambda'^{\epsilon}}$ indistinguishable. This follows from the induction hypothesis as follows: any size $5 \cdot 2^{\lambda'^{\epsilon}}$ adversary A that distinguishes G_1 and G_2 with advantage better than $1/(5 \cdot 2^{\lambda'^{\epsilon}})$ can be turned into an adversary A' that can distinguish $D_{\lambda,0,s0}$ and $D_{\lambda,1,s0}$ with the same advantage. As before, A' has Π^0_{λ}, Π^1_{λ}, T_{λ} and s hardcoded in it, and therefore the size of A' is at most $5 \cdot 2^{\lambda'^{\epsilon}} + \mathrm{poly}(\lambda') \leq 10 \cdot 2^{\lambda'^{\epsilon}}$. Hence, A' breaks the induction hypothesis that says $D_{\lambda,0,s0}$ and $D_{\lambda,1,s0}$ are $10 \cdot 2^{\lambda'^{\epsilon}}$-indistinguishable.

- Similarly, let G_3 be a hybrid distribution obtained by modifying the second component of G_2 as follows.
  $$G_3 \equiv D_{\lambda,1,s0}, D_{\lambda,1,s1}, \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi^0_{\lambda}, s, T_{\lambda}; R)$$
  Similarly as above, we have that G_2 and G_3 are $5 \cdot 2^{\lambda'^{\epsilon}}$-indistinguishable.

- Let G_4 be a hybrid distribution obtained by modifying the third component of G_3 as follows.
  $$G_4 \equiv D_{\lambda,1,s0}, D_{\lambda,1,s1}, \mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi^1_{\lambda}, s, T_{\lambda}; R)$$
  We show G_3 and G_4 are $5 \cdot 2^{\lambda'^{\epsilon}}$-indistinguishable. First, since $\Pi^{0,T}_{\lambda}(s) = \Pi^{1,T}_{\lambda}(s)$, by the security of the encoding scheme, we have that the encodings that form the third component

of G_3 and G_4 are S' indistinguishable where, similar to the base case, $B(l(\lambda, |s|))$ bounds from above $|\Pi^b_\lambda|, |s|$ and T

$$S' \geq \frac{10 \cdot 2^{l(\lambda,|s|)^\epsilon}}{l(\lambda, |s|)^c} - B(l(\lambda, |s|))^c \geq \frac{10 \cdot 2^{l(\lambda,|s|)^\epsilon}}{l(\lambda, |s|)^d} \geq 10 \cdot 2^{l(\lambda,|s|-1)^\epsilon}$$

Hence by a similar argument as before, the hybrid distributions are $5 \cdot 2^{\lambda'^\epsilon}$-indistinguishable.

- Finally we observe that G_4 and H^1 are $5 \cdot 2^{\lambda'^\epsilon}$-indistinguishable just as G_1 and H^0 were. By a simple hybrid argument, we have that H^0 and H^1 are $2^{\lambda'^\epsilon}$-indistinguishable.

  Recall that H^0 and H^1 are the distributions of outputs of $\widetilde{\Pi}^0_{s,R}$ and $\widetilde{\Pi}^1_{s,R}$ respectively. By the security of the randomized encoding scheme, the encodings of these machines, *i.e.* $D_{\lambda,0,s}$ and $D_{\lambda,1,s}$ are $S'(\lambda)$-indistinguishable where

$$S'(\lambda) \geq \frac{2^{l(\lambda,|s|)^\epsilon}}{l(\lambda, |s|)^c} - B'(l(\lambda, |s|)^c \geq \frac{2^{l(\lambda,|s|)^\epsilon}}{l(\lambda, |s|)^d} \geq \frac{2^{l(\lambda,|s|-1)^\epsilon} \cdot 2^{(2d\lambda)}}{2^{d\lambda} \cdot (2d\lambda)^{d/\epsilon}} \geq 10 \cdot 2^{l(\lambda,|s|-1)^\epsilon}$$

  where $B'(l(\lambda, |s|))$ bounds from above $|\Pi^b_{s,R}|$ and T' as in Lemma 2. The second inequality holds for sufficiently large λ. In the third inequality, we use the fact that $l(\lambda, |s|) \leq |s|(2d\lambda)^{1/\epsilon} \leq 2^\lambda(2d\lambda)^{1/\epsilon}$ and the last inequality holds for sufficiently large λ.

□

4.2 Nice Distributions

Later in Section 7, we show that compact RE does not exist for general distributions in the plain model. However, here we observe that the above construction of unbounded input IO relies only on compact RE for certain "special purpose" distributions that is not ruled out by the impossibility result in Section 7. We now abstract out the structure of these special purpose distributions. Let $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Dec})$ be a randomized encoding scheme; we define "nice" distributions w.r.t. RE.

0-nice distributions: We say that a pair of distribution ensembles $\{\mathcal{D}_{0,\lambda}\}$ and $\{D_{1,\lambda}\}$ are *0-nice* if $D_{0,\lambda}$ always outputs a fixed tuple (Π_0, x, T) while $D_{1,\lambda}$ always outputs a fixed tuple (Π_1, x, T), satisfying that $\Pi_0^T(x) = \Pi_1^T(x)$.

k-nice distributions: We say that a pair of distribution ensembles $\{\mathcal{D}_{0,\lambda}\}$ and $\{\mathcal{D}_{1,\lambda}\}$ are *k-nice* if there exist some $\ell = \text{poly}(\lambda)$ pairs of distributions $(\{\mathcal{E}^i_{0,\lambda}\}, \{\mathcal{E}^i_{1,\lambda}\})_{i\in[\ell]}$, where the i^{th} pair is k^i-nice with $k^i \leq k-1$, such that, $\mathcal{D}_{b,\lambda}$ samples tuple (Π_b, x_b, T_b) satisfying the following:

- For each $i \in [\ell]$, sample $(\Lambda^i_b, z^i_b, T^i_b) \xleftarrow{\$} \mathcal{E}^i_{b,\lambda}$.
- The output of $\Pi_b(x_b)$ consists of ℓ randomized encodings, where the i^{th} encoding is in the support of $\mathsf{Enc}(1^{\lambda'}, \Lambda^i_b, z^i_b, T^i_b)$, for some $\lambda' = \text{poly}(\lambda)$.

Finally, we say that a pair of distribution ensembles $\{\mathcal{D}_{0,\lambda}\}$ and $\{\mathcal{D}_{1,\lambda}\}$ are *nice* w.r.t. RE if they are *k-nice* w.r.t. RE for some integer k.

Our construction of unbounded input IO and its analysis in previous sections relies only on compact RE for nice distribution ensembles. Hence we can refine Theorem 7 to the following:

Proposition 1. *Assume the existence of a compact randomized encoding scheme* RE *which is sub-exponentially-indistinguishability-secure for every pair of distribution ensemble that are nice w.r.t.* RE*; assume further the existence of sub-exponentially secure one-way functions. Then, there is an unbounded-input indistinguishability obfuscator for Turing machines.*

We stress again that compact RE for nice distributions is not ruled out by the impossibility result in Section 7. Hence, we obtain unbounded input IO from a new assumption different from the extactability assumptions used in previous work [BCP14, ABG$^+$13, IPS15].

Candidate Construction: Finally, we describe a candidate construction of compact RE for nice distributions using the KLW indistinguishability obfuscator for bounded-input Boolean Turing machines: Given input $(1^\lambda, M, x, \mathsf{Time})$, the encoding is an obfuscation, using the KLW scheme, of the program $\Pi_{M,x}$ that on input $i \in [\mathsf{Time}]$ outputs the i^{th} bit of the output $M^{\mathsf{Time}}(x)$. Since $\Pi_{M,x}$ is Boolean, the KLW obfuscator can be applied, and the encoding time is $\text{poly}(\lambda, |M|, |x|, \log \mathsf{Time})$ (hence compact). By the security of indistinguishability obfuscation, for any M_1, x_1 and M_2, x_2 with identical outputs, their encodings are indistinguishable, and thus this construction is a weak compact RE. We here consider it also a candidate construction for compact RE with distributional indistinguishability.

5 Bounded-Input IO from Sublinear RE

In this section, we construct a bounded-input indistinguishability obfuscator for Turing machines, using randomized encoding schemes with sublinear compactness. The construction is very similar to the construction described earlier in Section 4. The main difference is that due to the fact that we only use sublinear compactness, the depth of the tree of encodings we construct must be bounded to some polynomial size given as a parameter, rather than being of depth 2^λ. Further, the correctness and efficiency analysis is slightly different to account for the semi-compactness of the RE scheme used, as opposed to full compactness in the previous construction.

Let $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ be a randomized encoding scheme for Turing machines with sub-exponential indistinguishability security and sublinear efficiency. Let c be the constant associated with the security loss in the security of $(\mathsf{Enc}, \mathsf{Eval})$. We assume that $\mathsf{Enc}(1^\lambda, \cdot, \cdot)$ requires a random tape of length λ, and this is without loss of generality for two reasons: First, one can always apply a PRG to expand the λ-bit random string to a pseudo-random string of arbitrary polynomial length, and second, by Claim 1 in Section 3.4 it is without loss of generality to assume that we start with a RE scheme with a sufficiently small sublinear time complexity and thus even counting the time for PRG expansion, the overall time complexity is still sublinear. Let PRG be a sub-exponentially secure pseudorandom generator and let ϵ be the constant associated with the sub-exponential security of PRG.

For every $\lambda \in \mathbb{N}$, $D \le 2^\lambda$, define

$$l(\lambda, -1) = \lambda$$

$$l(\lambda, D) = l(\lambda, D-1) + (2d\lambda)^{1/\epsilon}$$

where d is any constant strictly greater than c.

Construction 3. *Consider a Turing machine* Π*, security parameter* $\lambda \in \mathbb{N}$*, an input-length bound* n*, and time bound* T *on the running time of* Π*. For every partial input* $s \in \{0,1\}^*$ *with* $|s| \le n$ *and* $R \in \{0,1\}^{2l(\lambda,|s|)}$*, we recursively define a Turing machine* $\widetilde{\Pi}_{s,R}$ *as follows:*

When $|s| < n$:

On the empty input, $\widetilde{\Pi}_{s,R}$ outputs:

$$\begin{aligned}
&\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}_{s0,R_0}, T'(\lambda, |s|+1, |\Pi|, n, T); R_1)\\
&\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}_{s1,R_2}, T'(\lambda, |s|+1, |\Pi|, n, T); R_3)\\
&\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi, s, T; R_4)
\end{aligned}$$

where $(R_0, R_1, R_2, R_3, R_4) \leftarrow \mathsf{PRG}(R, 5 \cdot 2l(\lambda, |s|+1))$ and T' is a fixed polynomial in $\lambda, |s|+1, |\Pi|$, n and T, defined below. In the special case when $|s| = n-1$, the time bound used in the first two encodings is set to T.

On all other inputs, $\widetilde{\Pi}_{s,R}$ outputs $\perp$.

When $|s| = n$:

On the empty input, $\widetilde{\Pi}_{s,R}$ outputs $\mathsf{Enc}(1^{l(\lambda,|s|+1)}, \Pi, s, T; R)$. On all other inputs, $\widetilde{\Pi}_{s,R}$ outputs $\perp$.

Let $p(\cdot)$ and ϵ' respectively be the polynomial and constant corresponding to the sublinear efficiency of RE*. For any λ, n, Π and T, we define the following terms (which are implicitly functions of λ, n, Π and T):*

$$\lambda_n = l(\lambda, n)$$

$$\lambda_{n+1} = l(\lambda, n+1)$$

$$\widetilde{\Pi}_n = \widetilde{\Pi}_{0^n, 0^{2\lambda_n}}$$

$$A = 2 \cdot p(1^{\lambda_n}, |\widetilde{\Pi}_n|, 0)$$

$$B = p(1^{\lambda_{n+1}}, |\Pi|, n) T^{1-\epsilon'}$$

$$C = \mathsf{Time}_{\mathsf{PRG}}(0^{2\lambda_n}, 5 \cdot 2\lambda_{n+1})$$

We note that $\widetilde{\Pi}_n$ is the largest size of any $\widetilde{\Pi}_{s,R}$ we ever need to consider, and further, that it has a description of polynomial size, and hence every $\widetilde{\Pi}_{s,R}$ has polynomial size.

We define the polynomial $T'(\cdot,\cdot,\cdot,\cdot)$ (corresponding to the bound placed on the running time of $\widetilde{\Pi}_{s,R}$), for all λ, $s \in \{0,1\}^{\leq n}$, $R \in \{0,1\}^{2l(\lambda,|s|)}$, Π and T, to be

$$T'(\lambda, |s|, |\Pi|, n, T) = (n - |s| + 1) \cdot (A^{1/\epsilon'} + B + C)$$

where each of A, B and C are defined relative to $\lambda, |\Pi|, n$ and T given as input to T'.

Given this definition of $\widetilde{\Pi}_{s,R}$ and T', we define our indistinguishability obfuscator as follows:

Construction 4 (Indistinguishability Obfuscator)**.** *On input $\lambda \in \mathbb{N}$, Turing machine Π, input length bound n and time bound T, define $\widetilde{\Pi}$, the indistinguishability obfuscation of Π, to be*

$$\widetilde{\Pi} = \boldsymbol{iO}(1^\lambda, \Pi, n, T) = \mathsf{Enc}(1^{l(\lambda,0)}, \widetilde{\Pi}_{\epsilon,R}, T'(\lambda, 0, |\Pi|, n, T))$$

Where ϵ is the empty string, and $R \xleftarrow{\$} \{0,1\}^{2l(\lambda,0)}$ and T' a fixed polynomial in $\lambda, |\Pi|$, n and T, as described above.

Evaluation: The algorithm to evaluate $\widetilde{\Pi}$ on input $x \in \{0,1\}^d, d \le n$ proceeds as follows:

1. For every $0 \le i \le d$, compute encodings of $\widetilde{\Pi}_{x_{\le i},R}$ successively, starting with $\widetilde{\Pi}$, an encoding of $\widetilde{\Pi}_{\epsilon,R}$, and subsequently, for every $0 < i \le d$, computing the encoding of $\widetilde{\Pi}_{x_{\le i},R}$ by evaluating the encoding of $\widetilde{\Pi}_{x_{<i},R}$, and selecting the encoding of $\widetilde{\Pi}_{x_{\le i},R}$ from its output.

2. Evaluate the encoding of $\widetilde{\Pi}_{x,R} = \widetilde{\Pi}_{x_{\le d},R}$ and obtain from its output $(\hat{\Pi}, \hat{x}) = \mathsf{Enc}(1^{l(\lambda,|x|+1)}, \Pi, x, T)$.

3. Run $\mathsf{Eval}(\hat{\Pi}, \hat{x})$ to obtain $\Pi(x)$.

To analyze the correctness, running time, and compactness of our **iO** construction, we make use of the following lemma:

Lemma 3. *Let Π be a polynomial-time TM, $\lambda \in \mathbb{N}$ be a security parameter, n an input length bound, and $T \le 2^\lambda$ be a running time bound. Then, for every $s \in \{0,1\}^*$ with $|s| \le 2^\lambda$ and every $R \in \{0,1\}^{2l(\lambda,|s|)}$, the running time of $\widetilde{\Pi}_{s,R}$ is bounded by $T'(\lambda, |s|, |\Pi|, n, T)$.*

Proof. We prove the lemma by fixing $\lambda \in \mathbb{N}$, and inducting on the size of s.

Base case: $|s| = n$**.** In this case, the running time T'_s of $\widetilde{\Pi}_{s,R}$ is given by

$$\begin{aligned} T'_s &= \mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,|s|+1)}, \Pi, s, T) \\ &\le p(l(\lambda, |s|+1), |\Pi|, n) T^{1-\epsilon'} \\ &\le B \\ &\le T'(\lambda, |s|, |\Pi|, n, T) \end{aligned}$$

This completes the base case.

Inductive step: $|s| < n$ By the induction hypothesis, we assume that the lemma holds for all s' with $|s'| = |s| + 1$, in particular, for $s' = s0$ and $s' = s1$.

Then, an execution of $\widetilde{\Pi}_{s,R}$ runs a single evaluation of a PRG, and produces encodings of each of $\widetilde{\Pi}_{s0,R_0}$, $\widetilde{\Pi}_{s1,R_2}$ and (Π, s). The *PRG* runs in time $\mathsf{Time}_{\mathsf{PRG}}(R, 5 \cdot 2l(\lambda, |s|+1)) \le C$, while the encoding (Π, s) takes time $\mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,|s|+1)}, \Pi, s, T) \le B$. By the inductive hypothesis, the machines $\widetilde{\Pi}_{s0,R_0}$ and $\widetilde{\Pi}_{s1,R_2}$ each run in time $\le T'(\lambda, |s|+1, |\Pi|, T)$. Then, we have that the encoding time of $\widetilde{\Pi}_{s0,R_0}$ and $\widetilde{\Pi}_{s1,R_2}$ is bounded by:

$$\begin{aligned} &2 \cdot \mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}_{s0,R_0}, 0, T'(\lambda, |s|+1, |\Pi|, T)) \\ &\le 2 \cdot p(1^{l(\lambda,|s|+1)}, \widetilde{\Pi}_{s0,R_0}, 0) \cdot T'(\lambda, |s|+1, |\Pi|, T)^{1-\epsilon'} \\ &\le A \cdot T'(\lambda, |s|+1, |\Pi|, T)^{1-\epsilon'} \\ &\le A \cdot T'(\lambda, |s|+1, |\Pi|, T)^{1} \cdot T'(\lambda, |s|+1, |\Pi|, T)^{-\epsilon'} \\ &\le A \cdot T'(\lambda, |s|+1, |\Pi|, T) \cdot (A^{1/\epsilon'})^{-\epsilon} \\ &\le T'(\lambda, |s|+1, |\Pi|, T) \end{aligned}$$

Combining these facts together, we have that the running time T'_s of $\widetilde{\Pi}_{s,R}$ is given by:

$$\begin{aligned} T'_s &\le T'(\lambda, |s|+1, |\Pi|, T) + B + C \\ &\le (n - (|s|+1) + 1) \cdot (A^{1/\epsilon'} + B + C) + B + C \\ &\le (n - |s| + 1) \cdot (A^{1/\epsilon'} + B + C) \\ &\le T'(\lambda, |s|, |\Pi|, T) \end{aligned}$$

This concludes the inductive step, and the lemma follows. □

Correctness of iO: As in the construction in Section 4, the correctness of **iO** applied to any polynomial-length x follows from the correctness of evaluating encodings RE, applied at every level of the evaluation. More concretely, for each index $i \leq |x|$ of the evaluation, except with probability $\mu(l(\lambda, i))$, the evaluation of $\widetilde{\Pi}_{x_{<i},R}$ correctly produces $\widetilde{\Pi}_{x_{\leq i},R}$. Further, the final evaluation of $\hat{\Pi}, \hat{x}$ produces $\Pi^T(x)$ correctly except with probability $\mu(l(\lambda, i+1))$.

Crucially, the correctness at each step relies on the fact that each encoding $\widetilde{\Pi}_{x_{\leq i},R}$ uses time bound $T'(\lambda, i, |\Pi|, \log(T))$, which, as argued above, is sufficiently large to compute $\widetilde{\Pi}_{x_{\leq i},R}$ for the next level.

Overall, the probability of incorrect evaluation is $\leq \sum_{i=1}^{|x|} \mu(\lambda, i)$, which is negligible for any polynomial-length x.

Running time of iO: Again, considering step i, the evaluation at this step takes time $p(l(\lambda, i), |\Pi_{x_{<i},R}|, 0, T'(\lambda, i, |\Pi|, T)$, which is $\mathrm{poly}(\lambda, i, |\Pi|, T)$. Further, the evaluation of $\hat{\Pi}, \hat{x}$ to produce the final output of the **iO** is $p(l(\lambda, |x|+1), |x|, |\Pi|, T)$, which is $\mathrm{poly}(\lambda, |x|, |\Pi|, T)$. Therefore, overall the running time is $\mathrm{poly}(\lambda,\ |x|, |\Pi|, T)$.

Efficiency of iO: The size of $\mathbf{iO}(1^\lambda, \Pi, T)$ is the same as the size of $\widetilde{\Pi}_{\epsilon,R}$, which itself is bounded by $\mathsf{Time}_{\mathsf{Enc}}(1^{l(\lambda,0)}, \Pi_{\epsilon,R}, T'(\lambda, 0, |\Pi|, T))$, which is $\mathrm{poly}(\lambda, |\Pi|, T)$ by the efficiency of RE.

Security of iO: We note that the security proof for Construction 2 presented in Section 4 carries over exactly to the construction presented in this section. The only difference is that the base case for the induction starts from $|s| = n$ rather than $|s| = 2^\lambda$. Given this change, exactly the same inductive argument can be used to show that subexponential security of RE implies subexponential security of the **iO** construction given above.

6 Bounded-Input IO from Compact RE in the CRS Model

In this section we consider compact RE schemes for Turing machines in the *common reference string* (CRS) model. We show that (1) such encoding schemes can be constructed from compact functional encryption for circuits, and that (2) such encoding schemes suffice to get IO for circuits, which then by [KLW14] suffices to get bounded-input IO for Turing machines.

6.1 Randomized Encoding Schemes in the CRS model

We first formally define a randomized encoding scheme for a class of Turing machines in the CRS model. In this model, a one-time setup is performed which takes (in addition to the security parameter) a bound on machine size, input length, running time and output length. Only computations that respect these bounds can be encoded using this setup. The setup outputs a *long* CRS (the length is polynomial in the aforementioned bounds) and a *short* public encoding key (which depends only on the security parameter). The public encoding key is used by the encoding algorithm, which produces encodings that are *compact* as before. The CRS is used by the evaluation algorithm.

Definition 17 (Randomized Encoding Schemes in the CRS Model)**.** *A Randomized Encoding scheme* RE *for a class of Turing machines* $\{\mathcal{M}_\lambda\}$ *in the CRS model consists of the following algorithms:*

- $(\mathsf{crs}, pk) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l)$*: Setup gets as input (in unary) the security parameter λ, a machine size bound m, input length bound n, time bound T and output length bound l.*

- $\hat{\Pi}_x \xleftarrow{\$} \mathsf{Enc}(pk, \Pi, x)$*: Enc is probabilistic and gets as input a public key pk generated by Setup, Turing machine $\Pi \in \mathcal{M}_\lambda$ and input x. It outputs an encoding $\hat{\Pi}_x$*[6].

- $y \leftarrow \mathsf{Eval}(\hat{\Pi}_x, \mathsf{crs})$*: On input $\hat{\Pi}_x$ produced by Enc and crs produced by Setup, Eval outputs y.*

Correctness: *For every security parameters $\lambda \in \mathbb{N}$, $m, n, T, l \in \mathbb{N}$, Turing machine $\Pi \in \mathcal{M}_\lambda$ and input x, such that, $|\Pi| \leq m$, $|x| \leq n$, and $|\Pi^T(x)| \leq l$, we have that*

$$\Pr\left[\begin{array}{c} (\mathsf{crs}, pk) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l) \\ \hat{\Pi}_x \xleftarrow{\$} \mathsf{Enc}(pk, \Pi, x) \end{array} : \mathsf{Eval}(\hat{\Pi}_x, \mathsf{crs}) = \Pi^T(x)\right] = 1$$

The simulation security in the CRS model is essentially the same as that in the plain model (Definition 12), except that simulator can additionally simulate the CRS.

Definition 18. *A randomized encoding scheme RE for a class of Turing machines $\{\mathcal{M}_\lambda\}$ in the CRS model satisfies $(\lambda_0, S(\cdot))$-**simulation security**, if there exists a PPT algorithm Sim and a constant c, such that, for every ensemble $\{\Pi_\lambda, x_\lambda, m_\lambda, n_\lambda, l_\lambda, T_\lambda\}$ where $\Pi_\lambda \in \mathcal{M}_\lambda$ and $|\Pi_\lambda|, |x_\lambda|, m_\lambda, n_\lambda, l_\lambda, T_\lambda \leq B(\lambda)$ for some polynomial B, the following ensembles are $(\lambda_0, S'(\lambda))$ indistinguishable, with $S'(\lambda) = S(\lambda) - B(\lambda)^c$ for all $\lambda \in N$.*

$$\left\{(\mathsf{crs}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l),\ \hat{\Pi}_x \xleftarrow{\$} \mathsf{Enc}(pk, \Pi, x) \ : \ (\mathsf{crs}, pk, \hat{\Pi}_x)\right\}$$
$$\left\{(\mathsf{crs}, pk, \hat{\Pi}_x) \xleftarrow{\$} \mathsf{Sim}(1^\lambda, \Pi^T(x), 1^{|\Pi|}, 1^{|x|}, 1^m, 1^n, 1^T, 1^l) \ : \ (\mathsf{crs}, pk, \hat{\Pi}_x)\right\}$$

where subscripts of security parameter are suppressed.

Indistinguishability-security in the CRS model can be defined similar to that in the plain model, except now we need to work with distributions $D_{b,\lambda}$ that samples $(\Pi_b, x_b, T_b, m_b, n_b, l_b)$.

Definition 19 (Distributional $(\lambda_0, S(\cdot))$-Indistinguishability Security)**.** *A randomized encoding scheme RE for a class of Turing machines $\{\mathcal{M}_\lambda\}$ in the CRS model satisfies $(\lambda_0, S(\cdot))$*-ind-security*, if the following is true w.r.t. some constant $c > 0$: For every ensembles of distributions $\{D_{0,\lambda}\}$ and $\{D_{1,\lambda}\}$ with the following property:*

1. *there exists a polynomial B, such that, for every $b \in \{0, 1\}$, $D_{b,\lambda}$ is a distribution over tuples of the form $(\Pi_b, x_b, T_b, m_b, n_b, l_b)$ with polynomial size and $\lambda, |\Pi_b|, |x_b|, T_b, m_b, n_b, l_b \leq B(\lambda)$.*

2. *there exist an integer $\lambda_0' \geq \lambda_0$, and a function S' with $\leq S'(\lambda) \leq S(\lambda)$ for all λ, such that, the ensembles of output distributions $\{\mathcal{O}_{0,\lambda}\}$ and $\{\mathcal{O}_{1,\lambda}\}$ are $(\lambda_0', S'(\cdot))$-indistinguishable,*

$$\mathcal{O}_{b,\lambda} = \left((\Pi_b, x_b, T_b, m_b, n_b, l_b) \xleftarrow{\$} \mathcal{D}_{b,\lambda} : \Pi_b^{T_b}(x_b), |\Pi_b|, |x_b|, T_b, m_b, n_b, l_b\right)$$

[6]Encoding $\hat{\Pi}_x$ can be viewed as the combination of the program encoding $\hat{\Pi}$ and the input encoding $\hat{x}$ of Definition 11

the ensembles of encoding $\{\mathcal{E}_{0,\lambda}\}$ *and* $\{\mathcal{E}_{1,\lambda}\}$ *defined below is* $(\lambda'_0, S''(\cdot))$*-indistinguishable, where* $S''(\lambda) = \frac{S'(\lambda)}{\lambda^c} - B(\lambda)^c$.

$$\mathcal{E}_{b,\lambda} = \Big((\Pi_b, x_b, T_b, m_b, n_b, l_b) \xleftarrow{\$} \mathcal{D}_{0,\lambda},$$
$$(\mathsf{crs}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^{m_b}, 1^{n_b}, 1^{T_b}, 1^{l_b}),\ \hat{\Pi}_x \xleftarrow{\$} \mathsf{Enc}(pk, \Pi_b, x_b) \ :\ (\mathsf{crs}, pk, \hat{\Pi}_x)\Big)$$

In the CRS model, it is possible to have a compact RE for all Turing machines with simulation security. Therefore, we define compactness in the CRS model independent of security notion.

Definition 20 (Compactness and Sublinear Compactness in the CRS model)**.** *A randomized encoding scheme* $\mathsf{RE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Eval})$ *for Turing machines in the CRS model is* compact *(or sublinear compact) if* Setup *is PPT, and* Enc *and* Eval *have the same efficiency as their counterparts in a compact (or sublinear compact) randomized encoding scheme for Turing machines in the plain model.*

Remark 2. *As shown in Section 3.5, in the plain model* (λ_0, S)*-simulation security implies* (λ_0, S)*-indistinguishability security. We remark that the same holds in the CRS model and the proof is essentially the same. We omit the details here.*

6.2 Randomized encodings with CRS from Compact Functional Encryption

In this section we construct RE schemes in the CRS model from Compact Functional encryption schemes and pseudorandom generators.

Let $(\mathsf{FE.Setup}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be a public key, compact functional encryption scheme for $\mathbf{P}/poly$, and let PRG be a pseudorandom generator. We define a randomized encoding scheme in the CRS model $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Eval})$ as follows.

The setup algorithm $\mathsf{Setup}(1^\lambda, 1^m, 1^n, 1^T, 1^l)$:

- Setup first generates keys for the functional encryption scheme $(mpk, msk) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ and samples a uniformly random string $s \leftarrow \{0,1\}^\lambda$.
- Next, it generates the string $c \leftarrow 0^l \oplus \mathsf{PRG}(s, l)$. That is, it encrypts 0^l using a one-time pad with the key coming from $\mathsf{PRG}(s, l)$
- Let U be the universal circuit that on input (Π, x) where $|\Pi| \le m$ and $|x| \le n$ runs machine Π on x for at most T steps and outputs the first l bits of the tape as output. We define a circuit $C_{U,c}$, that has the string c and circuit U hardcoded in it, as follows.
  1. $C_{U,c}$ takes as input (Π, x, s', b) where (Π, x) satisfies the size constraints as described above, $s' \in \{0,1\}^\lambda$ and $b \in \{0,1\}$.
  2. If $b = 0$ then $C_{U,c}$ outputs $U(\Pi, x)$.
  3. Otherwise $C_{U,c}$ outputs $c \oplus \mathsf{PRG}(s')$.
- Setup runs $sk_C \leftarrow \mathsf{FE.KeyGen}(msk, C_{U,c})$ and outputs sk_C as the common reference string crs and mpk as the public encoding key pk

The encoding algorithm $\mathsf{Enc}(pk, \Pi, x)$**:** Enc parses pk as the functional public key mpk and runs $ct \leftarrow \mathsf{FE.Enc}(mpk, (\Pi, x, 0^\lambda, 0))$. Enc outputs the functional ciphertext ct as the encoding $\hat{\Pi}_x$.

The evaluation algorithm $\mathsf{Eval}(\hat{\Pi}_x, \mathsf{crs})$: Eval parses $\hat{\Pi}_x$ as a functional ciphertext ct and crs as the functional secret key $sk_{C_{U,c}}$. Eval runs $y \leftarrow \mathsf{FE.Dec}(sk_{C_{U,c}}, ct)$ and outputs y.

The correctness of the above encoding scheme follows directly from that of the underlying functional encryption scheme. When a randomized encoding of (Π, x) is evaluated, it outputs the result of running the universal circuit U on (Π, x) that is $\Pi^T(x)$. Also the efficiency properties of the above scheme follow directly from the compactness properties of the functional encryption scheme. Also, note that if the functional encryption scheme we start from has sub-linear compactness (the ciphertext size is sub-linear in the circuit size of the function for which the functional secret keys are generated) then we get an encoding scheme with sub-linear compactness.

We now show the above encoding scheme is secure.

Theorem 8. *Let* $(\mathsf{FE.Setup}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *be a public key functional encryption scheme for* $\mathbf{P}$*/poly with* $(\lambda_0, S(\cdot))$ *selective security, and let* PRG *be a pseudorandom generator with* $(\lambda_0, S(\cdot))$ *security. The randomized encoding scheme defined above is* $(\lambda_0, \frac{S(\cdot)}{4})$*-simulation secure.*

Corollary 1. *If there exists a public key, compact functional encryption scheme with sub-exponential selective security, then there exists a compact randomized encoding scheme in the CRS model that is sub-exponential simulation secure.*

Proof. Let $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{Eval})$ be the randomized encoding scheme as defined above. We need to show there exists a PPT algorithm Sim that, when given the output of a machine on some input (together with the bound parameters of the machine and input), simulates an encoding that is indistinguishable from the real encoding of the machine and input. We define Sim as follows.

The simulator $\mathsf{Sim}(1^\lambda, 1^m, 1^n, 1^T, y, 1^{|\Pi|}, 1^{|x|})$**:** Here we denote

- Sim first generates functional keys $(mpk, msk) \leftarrow \mathsf{FE.Setup}(1^\lambda)$ and a random string $s \leftarrow \{0,1\}^\lambda$.
- Next, Sim generates the string $c \leftarrow y \oplus \mathsf{PRG}(s, |y|)$. That is, it encrypts the output y under a one-time pad with the key coming from $\mathsf{PRG}(s, |y|)$.
- Sim runs $sk_C \leftarrow \mathsf{FE.KeyGen}(msk, C_{U,c})$ where $C_{U,c}$ is same as in the construction of the encoding scheme, except with c being the string generated in the above step.
- Sim generates $ct \leftarrow \mathsf{FE.Enc}(mpk, (0^{|\Pi|}, 0^{|x|}, s, 1))$.
- Sim outputs sk_C as the simulated crs, mpk as the public encoding key pk and ct as simulated machine encoding $\hat{\Pi}_x$.

Now we formally prove that the above simulator is secure. Consider any ensemble $\{\Pi_\lambda, x_\lambda, m_\lambda, n_\lambda, l_\lambda, T_\lambda\}$ where $\Pi_\lambda \in \mathcal{M}_\lambda$ and $|\Pi_\lambda|, |x_\lambda|, m_\lambda, n_\lambda, l_\lambda, T_\lambda \leq B(\lambda)$ for some polynomial B. Subsequently, we suppress the security parameter in the subscript.

We need to show that for every $\lambda > \lambda_0$,

$$D_0 = \left(\begin{array}{c} (pk, \mathsf{crs}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, 1^{m(\lambda)}, 1^{n(\lambda)}, 1^{T(\lambda)}) \\ (\hat{\Pi}_x) \xleftarrow{\$} \mathsf{Enc}(pk, \Pi, x) \end{array} : \mathsf{crs}, pk, \hat{\Pi}_x \right)$$

$$D_1 = \left((pk, \mathsf{crs}, \hat{\Pi}_x) \xleftarrow{\$} \mathsf{Sim}(1^\lambda, 1^{m(\lambda)}, 1^{n(\lambda)}, 1^{T(\lambda)}, \Pi^T(x), 1^{|\Pi|}, 1^{|x|}) : \mathsf{crs}, pk, \hat{\Pi}_x \right)$$

are $\frac{S(\lambda)}{4} - B(\lambda)^d$ indistinguishable, where $\lambda_0, S(\cdot)$ is the security of functional encryption scheme, and d is some constant. We show this by a hybrid argument as follows.

- Let H_0 be the distribution of the real encoding D_0. Rewriting H_0 in terms of the underlying primitives we have

$$H_0 = \left(\begin{array}{c} (mpk, msk) \leftarrow \mathsf{FE.Setup}(1^\lambda) \\ s \leftarrow \{0,1\}^\lambda \\ c \leftarrow 0^l \oplus \mathsf{PRG}(s,l) \\ sk_{C_{U,c}} \leftarrow \mathsf{FE.KeyGen}(msk, C_{U,c}) \\ ct \leftarrow \mathsf{FE.Enc}(mpk, (\Pi, x, 0^\lambda, 0) \end{array} \quad : mpk, sk_{C_{U,c}}, ct \right)$$

- Let H_1 be a hybrid distribution exactly as above, except that c is generated as $c \leftarrow 0^l \oplus R$ where R is uniformly random from $\{0,1\}^l$. We claim H_0 and H_1 are $S(\lambda) - B(\lambda)^{d'}$ indistinguishable, where d' is some constant. This follows from the security of the pseudorandom generator. Any adversary A that distinguishes H_0 and H_1 can be turned into an adversary A' that breaks the security of the pseudorandom generator with the same advantage. A' has Π, x, T hard-coded and needs to run the setup and generation algorithms of the functional encryption schemes. Therefore the size of A' is

$$\mathsf{size}(A) + \text{poly}(B(\lambda)) = (S(\lambda) - B(\lambda)^{d'}) + B(\lambda)^{d'} = S(\lambda)$$

  Hence A' breaks the $(\lambda_0, S(\cdot))$ security of the pseudorandom generator and we have a contradiction.

- Let H_2 be a hybrid distribution just as above except that c is generated from the output y as $c \leftarrow y \oplus R$ where, as before, R is uniformly random from $\{0,1\}^l$. Since R is uniformly random, the distributions H_2 and H_1 are identical.

- Let H_3 be a hybrid distribution just as above except that the one-time pad key is generated using the pseudorandom generator. That is, $c \leftarrow y \oplus \mathsf{PRG}(s,l)$. As before, the distributions H_3 and H_2 are $S(\lambda) - B(\lambda)^{d'}$ indistinguishable, where d' is some constant.

- Let H_4 be a hybrid distribution just as above except that ct is generated as

$$ct \leftarrow \mathsf{FE.Enc}(mpk, (0^{m(\lambda)}, 0^{n(\lambda)}, s, 1))$$

  This is exactly the distribution as generated by the simulator Sim. We claim H_3 and H_4 are $S(\lambda) - B(\lambda)^{d''}$ indistinguishable where d' is some constant. This follows from the security of the functional encryption scheme. Any adversary A that distinguishes H_3 and H_4 can be turned into an adversary A' that breaks the security of the functional encryption scheme with the same advantage. A' selects the challenge messages as $(0^{m(\lambda)}, 0^{n(\lambda)}, s, 1)$ and $(\Pi, x, 0^\lambda, 0)$ and secret key query $C_{U,c}$ which has the same output y on both messages. A' needs to additionally run the pseudorandom generator. Hence the size of A' is $\mathsf{size}(A) + B(\lambda)^{d'}$ for some constant d', which as before results in size $S(\lambda)$. Hence A' breaks the $(\lambda_0, S(\cdot))$ security of the functional encryption scheme and we have a contradiction.

Each of the hybrid distribution pairs above are $S(\lambda) - B(\lambda)^{d'}$ indistinguishable. By a simple hybrid argument we have that the distributions $D_0 = H_0$ and $D_1 = H_4$ are $\frac{S(\lambda) - B(\lambda)^{d'}}{4} \geq \frac{S(\lambda)}{4} - B(\lambda)^d$ indistinguishable, for some constant d, hence completing the proof.

□

6.3 IO for circuits from RE in the CRS model

In this section we show that compact RE schemes for Turing machines in the CRS model implies **iO** for circuits; combining with the result of [KLW14] that **iO** for circuits implies **iO** for (bounded-input) Turing machines, we obtain the following theorem:

Theorem 9. *Assume the existence of sub-exponentially secure one-way functions. If there exists a compact randomized encoding scheme in the CRS model with sub-exponential simulation security, then there exists an bounded-input indistinguishability obfuscator for Turning machines.*

We note that in the proof of the theorem, it suffices to have a compact randomized encoding scheme in the CRS model satisfying distributional indistinguishability security, *with auxiliary inputs* (i.e., Definition 19 w.r.t. distributions $\{D_{b,\lambda}\}$ that additionally samples an auxiliary input z_b, and the security requirement is that if the output distributions together with the auxiliary inputs are indistinguishable, then the encodings together with the auxiliary inputs are also indistinguishable, with appropriate security loss). Since this security notion is implied by simulation security, for simplicity, we directly state and prove the theorem from compact randomized with simulation security below.

The construction and proof is very similar to that of unbounded-input **iO** from compact RE schemes in the plain model presented in Section 4. We thus highlight the major differences below.

Proof sketch for Theorem 9. We first briefly recall the main ideas behind the unbounded-input **iO** construction. The unbounded-input **iO** generates an encoding of a recursively defined Turing machine $\Pi_{\epsilon,R}$ (where ϵ is the empty string and R is uniformly random). $\Pi_{s,R}$ generates encodings of $\Pi_{s0,R_0}, \Pi_{s1,R_1}$ and of the machine to be obfuscated with input s, where R_0 and R_1 are pseudorandom strings derived from R. For every such machine $\Pi_{s,R}$, we refer to $|s|$ as the *level* of the machine. Evaluating the obfuscation on an input of length n involves evaluating an encoding of a machine at every level $i \in \{0, \ldots, n\} = [n]$.

Construction: Our circuit obfuscator $i\mathcal{O}$ gets as input the security parameter 1^λ and the circuit to obfuscate C. Let n be the input length of C. To use RE schemes in the CRS model, $i\mathcal{O}$ first generates a RE setup (pk_i, crs_i) for every level $i \in [n]$. The obfuscation consists of $\vec{\mathsf{crs}} = \{\mathsf{crs}_i\}_{i=0}^n$, and an encoding of the machine $\Pi_{\vec{pk}_1,C,\epsilon,R}$ which has public keys $\vec{pk}_1 = \{pk_i\}_{i=1}^n$ hardcoded (more generally $\vec{pk}_i$ denotes $\{pk_j\}_{j=i}^n$). For every level $i < n$ and $s \in \{0,1\}^i$, the machine $\Pi_{\vec{pk}_{i+1},C,s,R}$ uses pk_{i+1} to generate encodings of $\Pi_{\vec{pk}_{i+2},C,s0,R_0}$ and $\Pi_{\vec{pk}_{i+2},C,s1,R_1}$. When $i = n$, the machine $\Pi_{\vec{pk}_{i+1},C,s,R}$ simply outputs $C(s)$. To evaluate the obfuscation on an input x, one evaluates an encoding of the machine $\Pi_{\vec{pk}_{i+1},C,x[1\ldots i],R}$ at every level $i \in [n]$, using $\mathsf{crs}_i \in \vec{\mathsf{crs}}$.

Note that, just as in the unbounded-input **iO** construction (Lemma 2), the compactness of the RE ensures that the size of the encodings at each level is some fixed polynomial in the security parameter and the circuit size. The obfuscation additionally contains a crs_i for every level $i \in [n]$ where the length of crs_i is polynomial in the running time of machines at that level (*i.e.* the time taken to encode machines at the next level), which by the same argument (Lemma 2) is some fixed polynomial in the security parameter and the circuit size. All in all, the size of the obfuscated circuit is polynomial in the security parameter and the circuit size.

Security: We need to show that, for any pair of functionally equivalent circuits C_0 and C_1, the joint distribution $(\vec{\mathsf{crs}}, \tilde{\Pi}_{\vec{pk}_1,\epsilon,C_0,R})$ is indistinguishable from $(\vec{\mathsf{crs}}, \tilde{\Pi}_{\vec{pk}_1,\epsilon,C_1,R})$. Just as in the proof of Theorem 7, we prove a stronger statement by induction. We claim that for every level $i \in [n]$, and every $s \in \{0,1\}^i$, the joint distribution $(\tilde{\Pi}_{\vec{pk}_{i+1},s,C_0,R}, \vec{\mathsf{crs}}_i, \vec{pk}_i)$ is indistinguishable from $(\tilde{\Pi}_{\vec{pk}_{i+1},s,C_1,R}, \vec{\mathsf{crs}}_i, \vec{pk}_i)$.

When $i = n$ (the base case), the above distributions are indistinguishable since the output of $\Pi_{\vec{pk}_{i+1},s,C_0,R}$ (in this case $C_0(s)$) is identical to the output of $\Pi_{\vec{pk}_{i+1},s,C_1,R}$. Hence by the indistinguishability security of the RE scheme (which is implied by simulation security, see Remark 2), $(\tilde{\Pi}_{\vec{pk}_{n+1},s,C_0,R}, pk_n, \mathsf{crs}_n)$ is indistinguishable from $(\tilde{\Pi}_{\vec{pk}_{n+1},s,C_1,R}, pk_n, \mathsf{crs}_n)$.

For the inductive step, we show the distributions are indistinguishable at any level $i < n$ assuming they are indistinguishable at level $i+1$. We do this by a hybrid argument as follows. Let H_0 be the joint distribution of the encoding at level i with C_0 hardcoded, with $\vec{\mathsf{crs}}_i$ and $\vec{pk}_i$. Writing this a bit differently, we have,

$$H_0 = ((\tilde{\Pi}_{\vec{pk}_{i+1},s,C_0,R}, \mathsf{crs}_i, pk_i), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$$

Next, we define hybrid distribution H_1 as follows.

$$H_1 = (\mathsf{Sim}(out(\tilde{\Pi}_{\vec{pk}_{i+1},s,C_0,R})), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$$

where Sim is the simulator for the RE scheme (for the sake of brevity in this proof sketch, we omit the other inputs to the simulator). By simulation security, H_0 and H_1 are indistinguishable.

Next, we define hybrid distribution H_2 by changing the underlying circuit to C_1.

$$H_2 = (\mathsf{Sim}(out(\tilde{\Pi}_{\vec{pk}_{i+1},s,C_1,R})), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$$

To show H_2 and H_1 are indistinguishable, we show that the following distributions are indistinguishable.

$$(out(\Pi_{\vec{pk}_{i+1},s,C_0,R}), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1}) \approx (out(\Pi_{\vec{pk}_{i+1},s,C_1,R}), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$$

This follows from the induction hypothesis as follows. Recall that the output of $\Pi_{\vec{pk}_{i+1},s,C_0,R}$ is a pair of level $i+1$ encodings $(\tilde{\Pi}_{\vec{pk}_{i+2},s0,C_0,R_0}, \tilde{\Pi}_{\vec{pk}_{i+2},s1,C_0,R_1})$. By the induction hypothesis, each of these encodings is indistinguishable from one with C_1 hardcoded, in the presence of $(\vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$. By a simple hybrid argument, the above indistinguishability holds, and hence H_2 is indistinguishable from H_1.

Finally, we define hybrid distribution H_3 which contains the encoding at level i with C_1 hardcoded.

$$H_3 = ((\tilde{\Pi}_{\vec{pk}_{i+1},s,C_1,R}, \mathsf{crs}_i, pk_i), \vec{\mathsf{crs}}_{i+1}, \vec{pk}_{i+1})$$

By the simulation security of the RE scheme H_3 is indistinguishable from H_2. Hence, by a hybrid argument H_0 is indistinguishable from H_3 hence completing the inductive step. □

7 Impossibility of Compact RE

In this section, we show several impossibility results related to sublinear (and hence compact) RE with different security:

Theorem 10. *The following impossibility results hold in the* plain *model:*

1. *Sublinear randomized encoding schemes with (polynomial) simulation security do not exist, assuming one-way functions.*

2. *Sublinear randomized encoding schemes with* sub-exponential *indistinguishability security do not exist, assuming sub-exponentially secure one-way functions.*

3. *Sublinear randomized encoding schemes with (polynomial) indistinguishability security do not exist, assuming bounded-input* **iO** *for Turning machines and one-way functions.*

Next we proceed to prove Theorem 10.

Impossibility 1. The impossibility of sublinear RE with simulation security follows from standard techniques that leverages the sublinear-size of the encoding to derive a contradiction to the imcompressibility of pseudo-random strings; below, we provide a proof sketch.

Proof Sketch. We argue that assuming one-way functions, compact RE with simulation security does not exist in the plain model. Suppose not and there is a compact RE that admits a simulator Sim which on input the output $y = \Pi(x)$ can simulate an encoding $(\widetilde{\Pi}, \tilde{x})$ that is indistinguishable from an honestly generated encoding $(\hat{\Pi}, \hat{x})$. By the indistinguishability, it follows that evaluating $(\widetilde{\Pi}, \tilde{x})$ yields the output y. Now, consider the specific computation of evaluating a PRG G on a short random seed s, that is, $\Pi = G$ and $x = s$. The simulator Sim on the pseudo-random output $y = G(s)$, produces $(\tilde{G}, \tilde{s})$, that can be evaluated to generate y. By the pseudo-randomness of PRG, it follows that Sim on input a truly random string y', can also output a tuple $(\tilde{G}', \tilde{s}')$ that evaluates to y', and the length of the tuple is sublinear in the length of y'. However, this contradicts the imcompressibility of random strings. □

Impossibility 2. The impossibility of sub-exponentially (ind-)secure sublinear RE is implied by impossibility 3, and the fact that bounded-input **iO** for Turing machines can be constructed from sub-exponentially (ind-)secure sublinear RE. More precisely, by our construction in Section 5, sub-exp secure sublinear-RE (and one-way functions) imply sub-exp secure **iO** for circuits; combined with the result of [KLW14] that sub-exp secure **iO** for circuits (and one-way functions) imply (polynomially secure) bounded-input **iO** for Turing machines, we have that assuming sub-exp secure one-way functions,

Sub-exp (ind-)secure Sublinear RE $\Longrightarrow$ Bounded-input **iO** for TMs

On the other hand, impossibility 3 states that

Bounded-input **iO** for TMs $\Longrightarrow$ **NO** (ind-)secure Sublinear RE

Therefore, if impossibility 3 is true, sub-exponentially secure sublinear RE is impossible assuming sub-exp secure one-way functions.

Impossibility 3. We show that (poly-secure) sublinear RE does not exist assuming the following two primitives.

- A bounded-input **iO** for Turing machines $i\mathcal{O}$, which on input $(1^\lambda, 1^n, R, T)$ runs in time
$$\mathsf{Time}_{i\mathcal{O}}(1^\lambda, 1^n, R, T) \leq \mathrm{poly}(\lambda, n, |R|, \log T) \ .$$

- A pseudo-random generator PRG that on input a seed s of length λ and a length k outputs a string of length k in time $\mathrm{poly}(\lambda, k)$. This is implied by the existence of one-way functions.

The efficiency of $i\mathcal{O}$ and PRG means that there is a constant d, such that, the following holds:

- For every $\lambda \in N$, $\ell = \ell(\lambda)$, program R with size $|R| \leq \lambda + \ell$, input length $n = \ell/2$, and $T = 2^\lambda$, the run-time of $i\mathcal{O}$ is

$$\mathsf{Time}_{i\mathcal{O}}(1^\lambda, 1^{\ell/2}, R, 2^\lambda) \leq (\lambda\ell)^d \ , \tag{1}$$

- and for every string $s \in \{0,1\}^\lambda$, and $k \in N$, the run-time of the PRG is

$$\mathsf{Time}_{\mathsf{PRG}}(s, k) \leq (\lambda k)^d \ . \tag{2}$$

Assuming such $i\mathcal{O}$ and PRG, we first show that there does not exist sublinear RE that are sufficiently compact in the following sense:

Claim 3. *Let d be a constant defined as above w.r.t. $i\mathcal{O}$ and PRG. There is no sublinear RE with time complexity satisfying the following:*

$$\mathsf{Time}_{\mathsf{Enc}}(1^\lambda, \Pi, x, T) \leq \mathrm{poly}(\lambda, |\Pi|, |x|)T^\alpha, \quad \textit{for } \alpha \leq 1/2d \ . \tag{3}$$

Before proving the claim, we argue that the above claim suffices for ruling out the existence of any sublinear RE schemes. By Claim 1 in Section 3.4, given any sublinear RE with complexity scaling with T^β for an arbitrary constant $\beta > 0$, one can reduce the time complexity to scaling with T^α for an arbitrary smaller constant α, by recursively composing it for some constant times. Combining this fact with the above claim, we conclude that assuming bounded-input **iO** for TMs and one-way function, sublinear RE do not exist.

Proof of Claim 3. Assume for contradiction that there is a sublinear RE scheme $\mathsf{RE} = (\mathsf{Enc}, \mathsf{Eval})$ whose time complexity depends on T^α with a sufficiently small $\alpha \leq 1/2d$ (and some multiplicative polynomial factor $\mathrm{poly}(\lambda, |\Pi|, |x|)$). Below, we derive a contradiction by constructing two ensembles of distributions $\{D_{0,\lambda}\}$, $\{D_{1,\lambda}\}$ that both sample triplets of form (Π, x, T); we show that (1) the outputs of the program and input sampled from these two distributions are indistinguishable, yet (2) the encoding of the program and input are distinguishable with probability close to 1. This violates the ind-security of RE, and gives a contradiction.

Let $\ell = \ell(\lambda)$ be a sufficiently large polynomial in λ, whose magnitude will become clear in the description below.

Distribution $D_{b,\lambda}$ samples triplet $(\Pi_b, 0, T)$ as follows:

- Sample two seeds $s \xleftarrow{\$} \{0,1\}^\lambda$, and $u \xleftarrow{\$} \{0,1\}^\lambda$.
- Turing machine $\Pi_b[\lambda, s, u]$ (with (λ, s, u) hardwired in), on input 0, proceeds in two steps
  1. Compute $\mathsf{PRG}(s, \ell) = y$, and $\mathsf{PRG}(u, \Gamma) = r$ where $\Gamma = (\lambda\ell)^d$.
  2. Obfuscate the program $R_b[b, y]$ described in Figure 1 to obtain

     $$\hat{R}_b = i\mathcal{O}(1^\lambda, 1^{\ell/2}, R_b, 2^\lambda \ ; \ r) \ .$$

     (The input length of R_b is bounded by $\ell/2$ and its run time is bounded by 2^λ.)
  3. Output $(y, \hat{R}_b)$.
- Set $T = 2(\lambda\Gamma)^d$.

  Note that T is an upper bound on the run-time of Π_b, as the first step of Π_b takes at most $(\lambda\ell)^d + (\lambda\Gamma)^d$ (according to condition (2)), and the second step takes at most $(\lambda\ell)^d$ steps (according to condition (1)).

The program $R_b[b, y]$

Hardwired Values [b, y]: b is in $\{0, 1, \perp\}$, and y an ℓ-bit length string.
Input ($\hat{\Pi}$, $\hat{x}$): encoding of a program and an input, s.t. $|\hat{\Pi}| + |\hat{x}| \leq \ell/2$.
The program proceeds as follow:

- Compute $z \leftarrow \mathsf{Eval}(\hat{\Pi}, \hat{x})$.
- Output b if the first ℓ bits of z equals to y, otherwise output $\perp$.

Figure 1: The program R_b used in the proof of impossibility of sublinear RE in the plain model.

We first show that the distributions, $out_{0,\lambda}$ and $out_{1,\lambda}$, of outputs of the program and input sampled from $D_{0,\lambda}$ and $D_{1,\lambda}$ are indistinguishable

$$\begin{aligned} out_{b,\lambda} = &\Big((\Pi_b, 0, T) \xleftarrow{\$} D_{b,\lambda} : \Pi_b^T(0), T, |\Pi_b|, |0|\Big) \\ &= \Big(s, u \xleftarrow{\$} \{0,1\}^\lambda, y = \mathsf{PRG}(s, \ell), r = \mathsf{PRG}(u, \Gamma), \\ &\qquad \hat{R}_b = i\mathcal{O}(1^\lambda, 1^{\ell/2}, R_b[b, y], 2^\lambda \ ; \ r) \ : \ (y, \hat{R}_b), T, |\Pi_b|, |0|\Big) \end{aligned}$$

Towards this, consider the following hybrid distributions.

Distribution $H_{b,\lambda}$ samples output tuple in the same way as $out_{b,\lambda}$ does, except that the pseudo-random strings y and r are replaced with truly random strings $\tilde{y} \xleftarrow{\$} \{0,1\}^\ell$ and $\tilde{r} \xleftarrow{\$} \{0,1\}^\Gamma$. More precisely,

$$\begin{aligned} H_{b,\lambda} = &\Big(\underline{\tilde{y} \xleftarrow{\$} \{0,1\}^\ell, \tilde{r} \xleftarrow{\$} \{0,1\}^\Gamma}, \\ &\qquad \tilde{R}_b = i\mathcal{O}(1^\lambda, 1^{\ell/2}, R_b[b, \underline{\tilde{y}}], 2^\lambda \ ; \ \underline{\tilde{r}}) \ : \ (\underline{\tilde{y}}, \tilde{R}_b), T, |\Pi_b|, |0|\Big) \end{aligned}$$

By the security of PRG, the above distribution is indistinguishable from $out_{b,\lambda}$.

Distribution $G_{b,\lambda}$ samples output tuple in the same way as $H_{b,\lambda}$ does, except that, instead of obfuscating the program R_b, it obfuscate the program $R_\perp$, which always outputs $\perp$ (and has the same run time as R_b). More precisely,

$$\begin{aligned} G_{b,\lambda} = &\Big(\tilde{y} \xleftarrow{\$} \{0,1\}^\ell, \tilde{r} \xleftarrow{\$} \{0,1\}^\Gamma, \\ &\qquad \underline{\tilde{R}_\perp = i\mathcal{O}(1^\lambda, 1^{\ell/2}, R_\perp[\perp, \tilde{y}], 2^\lambda \ ; \ \tilde{r})} \ : \ (\tilde{y}, \underline{\tilde{R}_\perp}), T, |\Pi_b|, |0|\Big) \end{aligned}$$

We claim that it follows from the security of $i\mathcal{O}$ that $G_{b,\lambda}$ and $H_{b,\lambda}$ are indistinguishable. Towards this, it suffices to show that with probability $1 - 2^{-\ell/2}$ over the random choice of $\tilde{y}$, the program $R_b[b, \tilde{y}]$ (obfuscated in $H_{b,\lambda}$) always outputs $\perp$, in which case $R_b[b, \tilde{y}]$ and $R_\perp[\perp, \tilde{y}]$ agree on all inputs and their obfuscation is indistinguishable. By construction, R_b outputs a value that is not $\perp$ only when the input $(\hat{\Pi}, \hat{x})$ satisfies that the first ℓ bits of the string z evaluated from it matches $\tilde{y}$. However, the encoding length is bounded by $\ell/2$, there are at most $2^{\ell/2}$ possible string z; when $\tilde{y}$ is chosen at random, the probability that z agrees with $\tilde{y}$ is at most $2^{-\ell/2}$. Except with this probability, R_b always outputs $\perp$.

On the other hand, we show that the distributions, $enc_{0,\lambda}$ and $enc_{1,\lambda}$, of the encoding of the program and input sampled from $D_{0,\lambda}$ and $D_{1,\lambda}$ are efficiently *distinguishable.*

$$enc_{b,\lambda} = \Big((\Pi_b, 0, T) \xleftarrow{\$} D_{b,\lambda} : (\hat{\Pi}_b, \hat{0}) \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \Pi_b, 0, T)\Big)$$

Towards this, we first show that the length of the encoding $(\hat{\Pi}_b, \hat{0})$ is bounded by $\ell/2$. This is because the run-time of Enc satisfies

$$\begin{aligned}
\mathsf{Time}_{\mathsf{Enc}}(1^\lambda, \Pi_b, 0, T) &\leq \mathrm{poly}(\lambda, |\Pi_b|, |0|)T^\alpha \text{ for } \alpha \leq 1/2d \\
&\leq \mathrm{poly}(\lambda)(2(\lambda\Gamma)^d)^{1/2d^2} = \mathrm{poly}(\lambda)\Gamma^{1/2d} \\
&= \mathrm{poly}(\lambda)((\lambda\ell)^d)^{1/2d} \\
&\leq \lambda^c\sqrt{\ell} \leq \ell/2
\end{aligned}$$

where the second line follows from that $|\Pi_b| = O(\lambda)$ and $T = 2(\lambda\Gamma)^d$, the third line follows from that $\Gamma = (\lambda\ell)^d$, and in the last line, c is a constant independent of d and the last inequality holds for sufficiently large $\ell = \ell(\lambda)$.

□

Acknowledgment: We are extremely grateful to Nir Bitansky and Omer Paneth for informing us of their impossibility result for compact RE assuming differing-input obfuscation, SNARKs and collision-resistant hash functions; this results was the inspiration behind our main impossibility result. We are also very grateful to them for many delightful and insightful discussions.

References

[ABG+13] Prabhanjan Ananth, Dan Boneh, Sanjam Garg, Amit Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. *IACR Cryptology ePrint Archive*, 2013:689, 2013.

[AIK04] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc^0. In *FOCS*, pages 166–175, 2004.

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

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. *IACR Cryptology ePrint Archive*, 2015:173, 2015.

[App11] Benny Applebaum. Randomly encoding functions: A new cryptographic paradigm - (invited talk). In *Information Theoretic Security - 5th International Conference, ICITS 2011, Amsterdam, The Netherlands, May 21-24, 2011. Proceedings*, pages 25–31, 2011.

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, volume 0, pages 106–115, 2001.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In *TCC*, pages 52–73, 2014.

[BCPR13] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. Indistinguishability obfuscation vs. auxiliary-input extractable functions: One must fall. *IACR Cryptology ePrint Archive*, 2013:641, 2013.

[BGI$^+$01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology CRYPTO 2001*, pages 1–18. Springer, 2001.

[BGL$^+$15] Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Sidharth Telang. Succinct randomized encodings and their applications. *IACR Cryptology ePrint Archive*, 2015:356, 2015.

[BKS15] Zvika Brakerski, Ilan Komargodski, and Gil Segev. From single-input to multi-input functional encryption in the private-key setting. *IACR Cryptology ePrint Archive*, 2015:158, 2015.

[BP15] Nir Bitansky and Omer Paneth. Zaps and non-interactive witness indistinguishability from indistinguishability obfuscation. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 401–427, 2015.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *IACR Cryptology ePrint Archive*, 2015:163, 2015.

[CHJV14] Ran Canetti, Justin Holmgren, Abhishek Jain, and Vinod Vaikuntanathan. Indistinguishability obfuscation of iterated circuits and RAM programs. *IACR Cryptology ePrint Archive*, 2014:769, 2014.

[GGH$^+$13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *Proc. of FOCS 2013*, 2013.

[GGHW13] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In *CRYPTO'14*, 2013.

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

[GKP$^+$13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 555–564, 2013.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 545–554, 2013.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 294–304, 2000.

[IK02a] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *ICALP*, pages 244–256, 2002.

[IK02b] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, pages 244–256, 2002.

[IPS15] Yuval Ishai, Omkant Pandey, and Amit Sahai. Public-coin differing-inputs obfuscation and its applications. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 668–697, 2015.

[KLW14] Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. Technical report, Cryptology ePrint Archive, Report 2014/925, 201 4. http://eprint. iacr. org, 2014.

[KMN$^+$14] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im)perfect obfuscation. 2014.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. *Proc. of STOC 2014*, 2014.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164, 1982.