On Extractability Obfuscation

Elette Boyle[1,⋆], Kai-Min Chung[2], and Rafael Pass[1,⋆⋆]

[1] Cornell University
ecb227@cornell.edu, rafael@cs.cornell.edu
[2] Academica Sinica
kmchung@iis.sinica.edu.tw

Abstract. We initiate the study of *extractability obfuscation*, a notion first suggested by Barak *et al.* (JACM 2012): An extractability obfuscator $e\mathcal{O}$ for a class of algorithms $\mathcal{M}$ guarantees that if an efficient attacker $\mathcal{A}$ can distinguish between obfuscations $e\mathcal{O}(M_1), e\mathcal{O}(M_2)$ of two algorithms $M_1, M_2 \in \mathcal{M}$, then $\mathcal{A}$ can efficiently recover (given M_1 and M_2) an input on which M_1 and M_2 provide different outputs.

- We rely on the recent candidate virtual black-box obfuscation constructions to provide candidate constructions of extractability obfuscators for NC^1; next, following the blueprint of Garg *et al.* (FOCS 2013), we show how to bootstrap the obfuscator for NC^1 to an obfuscator for all non-uniform polynomial-time Turing machines. In contrast to the construction of Garg *et al.*, which relies on indistinguishability obfuscation for NC^1, our construction enables succinctly obfuscating non-uniform *Turing machines* (as opposed to circuits), without turning running-time into description size.
- We introduce a new notion of *functional witness encryption*, which enables encrypting a message m with respect to an instance x, language L, and function f, such that anyone (and only those) who holds a witness w for $x \in L$ can compute $f(m, w)$ on the message and particular known witness. We show that functional witness encryption is, in fact, equivalent to extractability obfuscation.
- We demonstrate other applications of extractability extraction, including the first construction of fully (adaptive-message) indistinguishability-secure functional encryption for an unbounded number of key queries and unbounded message spaces.
- We finally relate indistinguishability obfuscation and extractability obfuscation and show special cases when indistinguishability obfuscation can be turned into extractability obfuscation.

⋆ Supported in part by AFOSR YIP Award FA9550-10-1-0093.

⋆⋆ Pass is 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.

Y. Lindell (Ed.): TCC 2014, LNCS 8349, pp. 52–73, 2014.

1 Introduction

Obfuscation. The goal of *program obfuscation* is to "scramble" a computer program, hiding its implementation details (making it hard to "reverse-engineer"), while preserving its functionality (i.e, input/output behavior). A first formal definition of such program obfuscation was provided by Hada [22]: roughly speaking, Hada's definition—let us refer to it as *strongly virtual black-box*—is formalized using the simulation paradigm. It requires that anything an attacker can learn from the obfuscated code, could be simulated using just black-box access to the functionality.[1] Unfortunately, as noted by Hada, only learnable functionalities can satisfy such a strong notion of obfuscation: if the attacker simply outputs the code it is given, the simulator must be able to recover the code by simply querying the functionality and thus the functionality must be learnable.

An in-depth study of program obfuscation was initiated in the seminal work of Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang [2]. Their central result shows that even if we consider a more relaxed simulation-based definition of program obfuscation—called *virtual black-box* obfuscation—where the attacker is restricted to simply outputting a single bit, impossibility can still be established (assuming the existence of one-way functions). Their result is even stronger, demonstrating the existence of families of functions such that given black-box access to f_s (for a randomly chosen s), not even a *single* bit of s can be guessed with probability significantly better than $1/2$, but given the code of any program that computes f_s, the entire secret s can be recovered. Thus, even quite weak simulation-based notions of obfuscation are impossible.

Barak *et al.* [2] also suggested an avenue for circumventing these impossibility results:[2] introducing the notions of *indistinguishability* and *"differing-inputs" obfuscation.* Roughly speaking, an indistinguishability obfuscator $i\mathcal{O}$ for a class of circuits $\mathcal{C}$ guarantees that given any two *equivalent* circuits C_1 and C_2 (i.e., whose outputs agree on all inputs) from the class, obfuscations $i\mathcal{O}(C_1)$ and $i\mathcal{O}(C_2)$ of the circuits are indistinguishable. In a recent breakthrough result, Garg, Gentry, Halevi, Raykova, Sahai, and Waters [14] provide the first candidate construction of indistinguishability obfuscators for all polynomial-size circuits. Additionally, Garg et al [14] and even more recently, the elegant works of Sahai and Waters [29] and Hohenberger, Sahai and Waters [23], demonstrate several beautiful (and surprising) applications of indistinguishability obfuscation.

In this work, we initiate the study of the latter notion of obfuscation—*"differing-inputs"*, or as we call it, *extractability obfuscation*—whose security guarantees are at least as strong as indistinguishability obfuscation, but weaker than virtual black-box obfuscation. We demonstrate candidate constructions of such extractability obfuscators, and new applications.

[1] Hada actually considered slight distributional weakening of this definition.

[2] Hada also suggested an approach for circumventing his impossibility result, sticking with a simulation-based definition, but instead restricting to particular classes of attacker. It is, however, not clear (to us) what reasonable classes of attackers are.

Extractability Obfuscation. Roughly speaking, an extractability obfuscator $e\mathcal{O}$ for a class of circuits $\mathcal{C}$ guarantees that if an attacker $\mathcal{A}$ can distinguish between obfuscations $i\mathcal{O}(C_1), i\mathcal{O}(C_2)$ of two circuits $C_1, C_2 \in \mathcal{C}$, then $\mathcal{A}$ can efficiently recover (given C_1 and C_2) a point x on which C_1 and C_2 differ: i.e., $C_1(x) \neq C_2(x)$.[3] Note that if C_1 and C_2 are equivalent circuits, then no such input exists, thus requiring obfuscations of the circuits to be indistinguishable (and so extractability obfuscation implies indistinguishability obfuscation).

We may rely on the candidate obfuscator for NC^1 of Brakerski and Rothblum [9] or Barak *et al.* [1] to obtain extractability obfuscation for the same class. We next demonstrate a bootstrapping theorem, showing how to obtain extractability obfuscation for all polynomial-size circuits. Our transformation follows [14], but incurs a somewhat different analysis.

Theorem 1 (Informal). *Assume the existence of an extractability obfuscator for NC^1 and the existence of a (leveled) fully homomorphic encryption scheme with decryption in NC^1 (implied, e.g., by Learning With Errors). Then there exists an extractability obfuscation for $P/poly$.*

Relying on extractability obfuscation, however, has additional advantages: in particular, it allows us to achieve obfuscation of (non-uniform) *Turing machines*. The size of the obfuscated code preserves a polynomial relation to the size of the original Turing machine. In contrast, existing obfuscator constructions [14,9] can achieve this only by first converting the Turing machine to a circuit, turning running time into size.

To achieve this, we additionally rely on the existence of **P**-certificates in the CRS model—namely, succinct non-interactive arguments for **P**.[4]

Theorem 2 (Informal). *Assume the existence of extractability obfuscation for NC^1, fully homomorphic encryption with decryption in NC^1 and P-certificates (in the CRS model). Then there exists extractability obfuscation for polynomial-size Turing machines.*

On a high level, our construction follows the one from [14] but (1) modifies it to deal with executions of Turing machines (by relying on an oblivious Turing machine), and more importantly (2) compresses "proofs" by using **P**-certificates. We emphasize that this approach does *not* work in the setting of indistinguishability obfuscation. Intuitively, the reason for this is that **P**-certificates of false statements *exist*, but are just hard to find; efficiently extracting such **P**-certificates from a successful adversary is thus crucial (and enabled by the extractability property).

We next explore applications of extractability obfuscation.

[3] Pedantically, our notion is a slightly relaxed version of that of [2]; see Section 3.

[4] Such certificates can be either based on knowledge-of-exponent type assumptions [4], or even on *falsifiable* assumptions [12].

Functional Witness Encryption. Consider the following scenario: You wish to encrypt the labels in a (huge) graph (e.g., names of people in a social network) so that no one can recover them, unless there is a clique in the graph of size, say, 100. Then, anyone (and only those) who knows such a clique should be able to recover the labels of the nodes *in the identified clique* (and only these nodes). Can this be done?

The question is very related to the notion of *witness encryption*, recently introduced by Garg, Gentry, Sahai, and Waters [15]. Witness encryption makes it possible to encrypt the graph in such a way that anyone who finds any clique in the graph can recover the *whole* graph; if the graph does not contain any such cliques, the graph remains secret. The stronger notion of extractable witness encryption, introduced by Goldwasser, Kalai, Popa, Vaikuntanathan, and Zeldovich [20], further guarantees that the graph can only be decrypted by someone who actually knowns a clique. However, in contrast to existing notions, here we wish to reveal *only the labels associated with the particular known clique.*

More generally, we put forward the notion of *functional witness encryption (FWE).* An FWE scheme enables one to encrypt a message m with respect to an NP-language L, instance x and function f, such that anyone who has (and only those who have) a witness w for $x \in L$ can recover $f(m, w)$. In the above example, m contains the labels of the whole graph, w is a clique, and $f(m, w)$ are the labels of all nodes in w. More precisely, our security definition requires that if you can tell apart encryptions of two messages m_0, m_1, then you must know a witness w for $x \in L$ such that $f(m_0, w) \neq f(m_1, w)$.

We observe that general-purpose FWE and extractability obfuscation actually are equivalent (up to a simple transformation).

Theorem 3 (Informal). *There exists a FWE for NP and every polynomial-size function f if and only if there exists an extractability obfuscator for every polynomial-size circuit.*

The idea is very simple: Given an extractability obfuscator $e\mathcal{O}$, an FWE encryption of the message m for the language L, instance x and function f is the obfuscation of the program that on input w outputs $f(m, w)$ if w is a valid witness for $x \in L$. On the other hand, given a general-purpose FWE, to obfuscate a program Π, let f be the universal circuit that on input (Π, y) runs Π on input y, let L be the trivial language where every witness is valid, and output a FWE of the message Π—since every input y is a witness, this makes it possible to evaluate $\Pi(y)$ on every y.

Other Applications. Functional encryption [6,28] enables the release of "targeted" secret keys sk_f that enable a user to recover $f(m)$, and *only* $f(m)$, given an encryption of m. It is known that strong simulation-based notions of security cannot be achieved if users can request an unbounded number of keys. In contrast, Garg *et al.* elegantly showed how to use indistinguishability obfuscation to satisfy an indistinguishability-based notion of functional encryption (roughly, that encryptions of any two messages m_0, m_1 such that $f(m_0) = f(m_1)$ for all

the requested secret keys sk_f are indistinguishable). The main construction of Garg et al, however, only achieves *selective-message* security, where the attacker must select the two message m_0, m_1 to distinguish before the experiment begins (and it can request decryption keys sk_f). Garg *et al.* observe that if they make subexponential-time security assumptions, use complexity leveraging, and consider a small (restricted) message space, then they can also achieve adaptive-message security.

We show how to use extractability obfuscation to directly achieve full adaptive-message security for any unbounded size message space (without relying on complexity leveraging).

The idea behind our scheme is as follows. Let the public key of the encryption scheme be the verification key for a signature scheme, and let the master secret key (needed to release secret keys sk_f) be the signing key for the signature scheme. To encrypt a message m, obfuscate the program that on input f and a valid signature on f (with respect to the hardcoded public key) simply computes $f(m)$. The secret key sk_f for a function f is then simply the signature on f. (The high-level idea behind the construction is somewhat similar to the one used in [20], which used witness encryption in combination with signature schemes to obtain simulation-based FE for a *single* function f; in contrast, we here focus on FE for an unbounded number of functions).

Proving that this construction works is somewhat subtle. In fact, to make the proof go through, we here require the signature scheme in use to be of a special *delegatable* kind—namely, we require the use of *functional signatures* [7,3] (which can be constructed based on non-interactive zero-knowledge (NIZK) arguments of knowledge), which make it possible to delegate a signing key sk' that allows one to sign only messages satisfying some predicate. The delegation property is only used in the security reduction and, roughly speaking, makes it possible to simulate key queries without harming security for the messages selected by the attacker.

Theorem 4 (Informal). *Assume the existence of NIZK arguments of knowledge for NP and the existence of extractability obfuscators for polynomial-size circuits. Then there exists an (adaptive-message) indistinguishability-secure functional encryption scheme for arbitrary length messages.*

Another interesting feature of our approach is that it directly enables constructions of Hierarchical Functional Encryption (HiFE) (in analogy with Hierarchical Identity-Based encryption [24]), where the secret keys for functions f can be released in a hierarchical way (the top node can generate keys for subsidiary nodes, those nodes can generate keys for its subsidiaries etc.). To do this, simply modify the encryption algorithm to release the $f(m)$ message in case you provide an appropriate chain of signatures that terminates with a signature on f.

From Indistinguishability Obfuscation to Extractability Obfuscation. A natural question is whether we can obtain extractability obfuscation from indistinguishability obfuscation. We address this question in two different settings: first directly

in the context of obfuscation, and second in the language of FWE. (Recall that these two notions are equivalent when dealing with arbitrary circuits and arbitrary functions; however, when considering restricted function classes, there are interesting differences).

- We introduce a weaker form of extractability obfuscation, in which extraction is only required when the two circuits differ on only polynomially many inputs. We demonstrate that any indistinguishability obfuscation in fact implies weak extractability obfuscation.

  Theorem 5 (Informal). *Any indistinguishability obfuscator for P/poly is also a weak extractability obfuscator for P/poly.*

- Mirroring the definition of indistinguishability obfuscation, we may define a weaker notion of FWE—which we refer to as *indistinguishability FWE* (or iFWE)—which only requires that if $f(m_0, w) = f(m_1, w)$ for *all* witnesses w for $x \in L$, then encryptions of m_0 and m_1 are indistinguishable (in contrast, the stronger notion requires that if you can distinguish between encryptions of m_0 and m_1 you must know a witness on which they differ). It follows that iFWE for languages in NP and functions in P/poly is equivalent to indistinguishability obfuscation for P/poly, up to a simple transformation. We show that if restricting to languages with polynomially many witnesses, it is possible to turn any iFWE to an FWE.

  Theorem 6 (Informal). *Assume there exists indistinguishability FWE for every NP language with polynomially many witnesses, and the function f. Then for every language L in NP with polynomially many witnesses, there exists an FWE for L and f.*

  Our proof relies on a local list-decoding algorithm for a large-alphabet Hadamard code due to Goldreich, Rubinfeld and Sudan [19].

Theorems 5 and 6 are incomparable in that Theorem 5 begins with a stronger assumption and yields a stronger conclusion. More precisely, if one begins with iFWE supporting all languages in NP and functions in P/poly, then the equivalence between indistinguishability (respectively, standard) FWE and indistinguishability (resp., extractability) obfuscation, in conjunction with the transformation of Theorem 5, yields a *stronger* outcome in the setting of FWE than Theorem 6: Namely, a form of FWE where (extraction) security holds as long as the function $M(m, w)$ is not "too sensitive" to m: i.e., if for any two messages m_0, m_1 there are only polynomially many witnesses w for which $M(m_0, w) \neq M(m_1, w)$. This captures, for example, functions M that only rarely output nonzero values. Going back to the example of encrypting data m associated with nodes of a social network, we could then allow someone holding clique w to learn whether the nodes in this clique satisfy some chosen rare property (e.g., contains someone with a rare disease, all have the same birthday, etc). Indeed, while there may be many cliques (corresponding to several, even exponentially many, witnesses w), it will be the case that $M(m, w)$ is almost always 0, for all but polynomially many w.

On the other hand, Theorem 6 also provides implications of iFWE for restricted function classes. In particular, Theorem 6 gives a method for transforming indistinguishability FWE for the trivial function $f(m, w) = m$ to FWE for the same function f. It is easy to see that indistinguishability FWE for this particular f is equivalent to the notion of witness encryption [15], and FWE for the same f is equivalent to the notion of extractable witness encryption of [20]. Theorem 6 thus shows how to turn witness encryption to extractable witness encryption for the case of languages with polynomially many witness.

Finally, we leave open whether there are corresponding transformations from indistinguishability obfuscation in the case of many disagreeing inputs, and iFWE in the case of many witnesses. In the latter setting, this is interesting even for the special case of witness encryption (i.e., the function $f(m, w) = m$).

Overview of the Paper. In Section 2, we present definitions and notation for some of the tools used in the paper. In Section 3, we introduce the notion of extractability obfuscation and present a bootstrapping transformation from any extractability obfuscator for NC^1 to one for all poly-time Turing machines. In Section 4, we define functional witness encryption (FWE), and show an equivalence between FWE and extractability obfuscation. In Section 5, we describe an application of extractability obfuscation, in achieving indistinguishability functional encryption with unbounded-size message space. In Section 6, we explore the relationship between indistinguishability and extractability obfuscation, providing transformations from the former to the latter in special cases.

2 Preliminaries

2.1 Fully Homomorphic Encryption

A fully homomorphic encryption scheme $\mathcal{E} = (\mathsf{Gen}_{\mathsf{FHE}}, \mathsf{Enc}_{\mathsf{FHE}}, \mathsf{Dec}_{\mathsf{FHE}}, \mathsf{Eval}_{\mathsf{FHE}})$ is a public-key encryption scheme associated with an additional polynomial-time algorithm $\mathsf{Eval}_{\mathsf{FHE}}$, which enables computation on encrypted data. Formally, we require $\mathcal{E}$ to have the following correctness property:

Definition 1 (FHE correctness). *There exists a negligible $\nu(k)$ s.t.*

$$\Pr_{\mathsf{pk},\mathsf{sk}\leftarrow\mathsf{Gen}(1^k)}\left[\begin{array}{c}\forall \textit{ ciphertexts } c_1,...,c_n \textit{ s.t. } c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(b_i),\\ \forall \textit{ poly-size circuits } f:\{0,1\}^n\to\{0,1\}\\ \mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}_{\mathsf{pk}}(f,c_1,...,c_n)) = f(b_1,...,b_n),\end{array}\right] \geq 1-\nu(k).$$

The size of $c' = \mathsf{Eval}_{\mathsf{FHE}}(\mathsf{pk}, \mathsf{Enc}_{\mathsf{FHE}}(\mathsf{pk}, m), C)$ must depend polynomially on the security parameter and the length of $C(m)$, but be otherwise independent of the size of the circuit C. For security, we require that $\mathcal{E}$ is semantically secure. We also require that Eval is deterministic, and that the decryption circuit $\mathsf{Dec}_{\mathsf{sk}}(\cdot)$ is in NC^1. Most known schemes satisfy these properties. Since the breakthrough of Gentry [17], several fully homomorphic encryption schemes have been constructed with improved efficiency and based on more standard assumptions such as LWE (Learning With Errors) (e.g., [10,8,18,11]), together with a circular security assumption. We refer the reader to these works for more details.

Remark 1 (Homomorphic evaluation of Turing machines). As part of our extractability obfuscation construction for general Turing machines (TM), we require the homomorphic evaluation of an *oblivious* TM with known runtime. Recall that a TM is said to be oblivious if its tape movements are independent of its input. The desired evaluation is done as follows.

Suppose $\hat{x} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_k)$ is an FHE encryption of plaintext message x (where $\hat{x}_\ell$ encrypts the ℓth position of x), $\hat{a} = (\hat{a}_1, \hat{a}_2, \ldots)$ an FHE encryption of the tape values, $\hat{s}$ an FHE ciphertext of the current state, and M an oblivious TM terminating on all inputs within t steps. More specifically, a description of M consists of an initial state s and description of a transition circuit, C_M. In each step $i = 1, \ldots, t$ of evaluation, M accesses some fixed position $\mathsf{pos}_{\mathsf{input}}(i)$ of the input, fixed position $\mathsf{pos}_{\mathsf{tape}}(i)$ of the tape (extending straightforwardly to the multi-tape setting), and the current value of the state, and evaluates C_M on these values.

Homomorphic evaluation of M on the encrypted input $\hat{x}$ then takes place in t steps: In each step i, the transition circuit C_M of M is homomorphically evaluated on the ciphertexts $\hat{x}_{\mathsf{pos}_{\mathsf{input}}}$, $\hat{a}_{\mathsf{pos}_{\mathsf{tape}}}$, and $\hat{s}$, yielding updated values for these ciphertexts. The updated state ciphertext $\hat{s}$ resulting after t steps is the desired output ciphertext. Note that obliviousness of the Turing machine is crucial for this efficient method of homomorphic evaluation, as any input-dependent choices for the head location would only be available to an evaluator in encrypted form.

Overall, homomorphic evaluation of M takes time $O(t(k) \cdot \mathsf{poly}(k))$, and can be described in space $O(|M| \cdot \mathsf{poly}(k))$.

2.2 (Indistinguishability) Functional Encryption

A functional encryption scheme [6,28] enables the release of "targeted" secret keys that enable a user to recover $f(m)$—and only $f(m)$—given an encryption of m. In this work, we will consider the indistinguishability notion of security for functional encryption. Roughly, such a scheme is said to be secure if an adversary who requests and learns secret keys sk_f for a collection of functions f cannot distinguish encryptions of any two messages m_0, m_1 for which $f(m_0) = f(m_1)$ for every requested f. We refer the reader to e.g. [6,28] for a formal definition.

2.3 P-Certificates

P-Certificates are a succinct argument system for **P**. We consider **P** certificates in the CRS model.

For every $c \in \mathbb{N}$, let $L_c = \{(M, x, y) : M(x) = y \text{ within } |x|^c \text{ steps}\}$. Let $T_M(x)$ denote the running time of M on input x.

Definition 2 (P-certificates). *[13] A tuple of probabilistic interactive Turing machines* $(\mathsf{CRSGen}_{\mathsf{cert}}, P_{\mathsf{cert}}, V_{\mathsf{cert}})$ *is a* **P**-certificate system *in the CRS model if there exist polynomials* g_P, g_V, ℓ *such that the following hold:*

- **Efficient Verification:** *On input* $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^k)$, $c \geq 1$, *and a statement* $q = (M, x, y) \in L_c$, *and* $\pi \in \{0,1\}^*$, V_{cert} *runs in time at most* $g_V(k + |q|)$.

- **Completeness by a Relatively Efficient Prover:** *For every $c, d \in \mathbb{N}$, there exists a negligible function μ such that for every $k \in \mathbb{N}$ and every $q = (M, x, y) \in L_c$ such that $|q| \leq k^d$,*
  $\Pr[\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^k); \pi \leftarrow P_{\mathsf{cert}}(\mathsf{crs}, c, q) : V_{\mathsf{cert}}(\mathsf{crs}, c, q, \pi) = 1] \geq 1 - \mu(k)$.
  Furthermore, P_{cert} on input (crs, c, q) outputs a certificate of length $\ell(k)$ in time bounded by $g_P(k + |M| + T_M(x))$.
- **Soundness:** *For every $c \in \mathbb{N}$, and every (not necessarily uniform) PPT P^*, there exists a negligible function μ such that for every $k \in \mathbb{N}$,*
  $\Pr[\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^k); (q, \pi) \leftarrow P^*(\mathsf{crs}, c) : V_{\mathsf{cert}}(\mathsf{crs}, c, q, \pi) = 1 \wedge q \notin L_c] \leq \mu(k)$.

P-certificates are directly implied by any publicly-verifiable succinct non-interactive argument system (SNARG) for **P**. It was shown by Chung et al. [13] that **P**-certificates can be based on *falsifiable* assumptions [27].

Theorem 7. *Assuming that Micali's CS proof [26] is sound, or assuming the existence of publicly-verifiable fully succinct SNARG system for* **P** *[4] (which in turn can be based on any publicly-verifiable SNARG [21,25,16,5]), then there exists a* **P**-certificate system in the CRS model.

3 Extractability Obfuscation

We now present and study the notion of *extractability obfuscation*, which is a slight relaxation of "differing-inputs obfuscation" introduced in [2]. Intuitively, such an obfuscator has the property that if a PPT adversary can distinguish between obfuscations of two programs M_0, M_1, then he must "know" an input on which they differ.

Definition 3 (Extractability Obfuscator). *(Variant of [2][5]) A uniform PPT machine $e\mathcal{O}$ is an* extractability obfuscator *for a class of Turing machines $\{\mathcal{M}_k\}_{k \in \mathbb{N}}$ if the following conditions are satisfied:*

- **Correctness:** *There exists a negligible function $\mathsf{negl}(k)$ such that for every security parameter $k \in \mathbb{N}$, for all $M \in \mathcal{M}_k$, for all inputs x, we have $\Pr[M' \leftarrow e\mathcal{O}(1^k, M) : M'(x) = M(x)] = 1 - \mathsf{negl}(k)$.*
- **Security:** *For every PPT adversary $\mathcal{A}$ and polynomial $p(k)$, there exists a PPT extractor E and polynomial $q(k)$ such that the following holds. For every $k \in \mathbb{N}$, every pair of Turing machines $M_0, M_1 \in \mathcal{M}_k$, and every auxiliary input z,*

$$\Pr\left[\begin{array}{c} b \leftarrow \{0,1\}; \\ M' \leftarrow e\mathcal{O}(1^k, M_b) \end{array} : \mathcal{A}(1^k, M', M_0, M_1, z) = b\right] \geq \frac{1}{2} + \frac{1}{p(k)} \quad (1)$$

$$\implies \quad \Pr\left[w \leftarrow E(1^k, M_0, M_1, z) : M_0(w) \neq M_1(w)\right] \geq \frac{1}{q(k)}. \quad (2)$$

[5] Formally, our notion of extractability obfuscation departs from differing-inputs obfuscation of [2] in two ways: First, [2] require the extractor E to extract a differing input for M_0, M_1 given *any* pair of programs M_0', M_1' evaluating equivalent functions. Second, [2] consider also adversaries who distinguish with negligible advantage $\epsilon(k)$, and require that extraction still succeeds in this setting, but within time polynomial in $1/\epsilon$. In contrast, we restrict our attention only to adversaries who succeed with noticeable advantage.

We remark that we can also consider a distributional-variant of the extraction condition, where instead of requiring the condition to hold with respect to every $M_0, M_1 \in \mathcal{M}_k$ and $z \in \{0,1\}^*$, we consider a distribution $\mathcal{D}$ that samples $(M_0, M_1, z) \leftarrow \mathcal{D}$ and requires that for every distribution $\mathcal{D}$, there exists an extractor such that the extraction condition to hold with respect to $\mathcal{D}$. In applications, it often suffices to require the extraction condition to hold with respect to some specific distribution $\mathcal{D}$. We here focus on the above definition for concrete exposition, but our results hold naturally also for the distributional-variant definition.

We contrast this notion with *indistinguishability obfuscation*:

Definition 4 (Indistinguishability Obfuscator). *[2] A uniform PPT machine $i\mathcal{O}$ is an* indistinguishability obfuscator *for a class of circuits $\{\mathcal{C}_k\}$ if $i\mathcal{O}$ satisfies the Correctness and Security properties as in Definition 3 (for circuit class $\{\mathcal{C}_k\}$ and circuits C_0, C_1 in the place of Turing machines), except with Line (2) replaced with the following:*

$$\implies \quad \exists w : C_0(w) \neq C_1(w). \tag{2'}$$

Note that any *extractability* obfuscator is also directly an *indistinguishability* obfuscator, since existence of an efficient extraction algorithm E finding desired distinguishing input w as in (2) in particular implies that such an input exists, as in (2′).

Remark 2. Note that in the definition of extractability obfuscation, the extractor is given access to the programs M_0, M_1. One could consider an even stronger notion of obfuscation, in which the extractor is given only black-box access to the two programs. As we show in the full version, however, achieving general-purpose obfuscation satisfying this stronger extractability notion is impossible.

We now present specific definitions of extractability obfuscators for special classes of Turing machines.

Definition 5 (Extractability Obfuscator for NC^1). *A uniform PPT machine $e\mathcal{O}_{NC^1}$ is called an* extractability obfuscator for NC^1 *if for constants $c \in \mathbb{N}$, the following holds: Let $\mathcal{M}_k$ be the class of Turing machines corresponding to the class of circuits of depth at most $c \log k$ and size at most k. Then $e\mathcal{O}(c, \cdot, \cdot)$ is an extractability obfuscator for the class $\{\mathcal{M}_k\}$.*

Definition 6 (Extractability Obfuscator for TM). *A uniform PPT machine $e\mathcal{O}_{\mathsf{TM}}$ is called an* extractability obfuscator for the class TM of polynomial-size Turing machines *if it satisfies the following. For each k, let $\mathcal{M}_k$ be the class of Turing machines Π containing a description of a Turing machine M of size bounded by k, such that Π takes two inputs, (t, x), with $|t| = k$, and the output of $\Pi(t, x)$ is defined to be the the output of running the Turing machine $M(x)$ for t steps. Then $e\mathcal{O}_{\mathsf{TM}}$ is an extractability obfuscator for $\{\mathcal{M}_k\}$.*

Applying the properties of extractability obfuscation to this class of Turing machines $\{\mathcal{M}_k\}$ implies that for programs $\Pi_0, \Pi_1 \in \mathcal{M}_k$ defined above

(corresponding to underlying size-k Turing machines M_0, M_1), efficiently distinguishing between obfuscations of Π_0 and Π_1 implies that one can efficiently extract an input *pair* (t', x') for which $\Pi_0(t', x') \neq \Pi_1(t', x')$. In particular, either $M_0(x') \neq M_1(x')$ or $Runtime(M_0, x') \neq Runtime(M_1, x)$. Thus, if restricting attention to a subclass of $\mathcal{M}_k$ for which each pair of programs satisfies $Runtime(M_0, x) = Runtime(M_1, x)$ for each input x, then "standard" extraction is guaranteed (i.e., such that the extracted input contains x' satisfying $M_0(x') \neq M_1(x')$), while achieving input-specific runtime of the obfuscated program. (Indeed, for an input x of unknown runtime, one simply executes the obfuscated program $\tilde{\Pi}$ sequentially with increasing time bounds $t = k, 2k, 2^2k, 2^3k, \cdots$ until a non-$\perp$ output is received). If restricting to a subclass $\mathcal{M}_k$ that has a polynomial runtime bound $t(k)$, then "standard" extraction can be guaranteed by simply defining $Runtime(M, x) = t(k)$ for every $M \in \mathcal{M}_k$ and every input x.

In the sequel, when referring to an extractability obfuscation of a Turing machine M, we will implicitly mean the related program Π_M as above, but will suppress notation of the additional input t.

Definition 7 (Extractability Obfuscator for Bounded-Input TM). *A uniform PPT machine $e\mathcal{O}_{BI}$ is called an extractability obfuscator for bounded-input Turing machines if it satisfies the following. For each k and polynomial $\ell(k)$, let $\mathcal{M}_k^\ell$ be the class of Turing machines Π as in Definition 6, but where the inputs (t, x) of Π are limited by $|t| = k$ and $|x| \leq \ell(k)$. Then there exists a polynomial $p_s(k)$ for which $e\mathcal{O}_{BI}$ is an extractability obfuscator for $\{M_k^\ell\}$, and for every $k \in \mathbb{N}$, and every $M \in \mathcal{M}_k^\ell$, it holds that the obfuscation $M' \leftarrow e\mathcal{O}_{BI}(1^k, M, \ell(k))$ has size bounded by $p_s(\ell(k), k)$.*

3.1 Extractability Obfuscation for NC^1

In this work, we build upon the existence of any extractability obfuscator for NC^1. In particular, this assumption can be instantiated using the candidate obfuscator for NC^1 given by Brakerski and Rothblum [9] or Barak et al. [1]. These works achieve (stronger) *virtual black-box* security within an idealized model, based on certain assumptions. We refer the reader to [9,1] for more details.

Assumption 8 (NC^1 Extractability Obfuscator). *There exists a secure extractability obfuscator $e\mathcal{O}_{NC^1}$ for NC^1, as in Definition 5*

3.2 Amplifying to General Polynomial-Sized Turing Machines

In this section, we demonstrate how to bootstrap from an extractability obfuscator for NC^1 to one for *all* (bounded-input) Turing machines with a polynomial-sized description, by use of fully homomorphic encryption (FHE), in conjunction with a **P**-certificate system (a succinct argument system for statements in **P**).[6] Our construction provides also two corollaries. Relaxing our assump-

[6] **P**-certificates are immediately implied by any succinct non-interactive argument (SNARG) system for NP, but can additionally be based on *falisifiable* assumptions. We refer the reader to Section 2.3 for details.

tions, by using *leveled* FHE, and removing **P**-certificates, we achieve extractability obfuscation for polynomial-size circuits. And strengthening our assumption, replacing **P**-certificates with succinct non-interactive arguments of knowledge (SNARKs), yields extractability obfuscation for all polynomial-size Turing machines. Our construction follows the analogous amplification transformation of Garg et. al [14] in the (weaker) setting of indistinguishability obfuscation.

At a high level, the transformation works as follows. An obfuscation of a Turing machine M consists of two FHE ciphertexts g_1, g_2, each encrypting a description of M under a distinct public key, and an obfuscation of a certain (low-depth) verify-and-decrypt circuit. To evaluate an obfuscation of M on input x, a user will homomorphically evaluate the oblivious[7] universal Turing machine $U(\cdot, x)$ on both ciphertexts g_1 and g_2, and generate a **P**-*certificate* ϕ that the resulting ciphertexts e_1, e_2 were computed correctly. Then, he will provide a low-depth proof π that the certificate ϕ properly verifies (e.g., simply providing the entire circuit evaluation). The collection of e_1, e_2, ϕ, π is then fed into the NC^1-obfuscated program, which checks the proofs, and if valid outputs the decryption of e_1.

Note that the use of computationally sound **P**-certificates enables the size of the obfuscation of M to depend only on the *description size* of M, and not its runtime. This approach is not possible in the setting of *indistinguishability* obfuscation, as certificates of false statements *exist*, but are simply hard to find.

Theorem 9. *There exists a succinct extractability obfuscator $e\mathcal{O}$ for bounded-input* TM*, as in Definition 7, assuming the existence of the following tools:*
- *$e\mathcal{O}_{NC^1}$: an extractability obfuscator for the class of circuits NC^1.*
- *$\mathcal{E} = (\mathsf{Gen}_{\mathsf{FHE}}, \mathsf{Enc}_{\mathsf{FHE}}, \mathsf{Dec}_{\mathsf{FHE}}, \mathsf{Eval}_{\mathsf{FHE}})$: a fully homomorphic encryption scheme with decryption* Dec *in NC^1.*
- *$(\mathsf{CRSGen}_{\mathsf{cert}}, P_{\mathsf{cert}}, V_{\mathsf{cert}})$: a* **P**-*certificate system in the CRS model.*

We remark that by replacing the **P**-certificates with succinct non-interactive arguments of knowledge (SNARKs) and additionally using collision resistant hash functions, then the resulting extractability obfuscator is secure for all polynomial-size Turing machines of possibly unbounded input size.

Corollary 1. *Based on any extractability obfuscator for the class of circuits NC^1, fully homomorphic encryption, succinct non-interactive arguments of knowledge (SNARKs), there exists an extractability obfuscator for* TM*, as in Definition 6.*

We also observe that by using a *leveled* FHE, and removing the **P**-certificates from the construction, we can still achieve extractability obfuscation for P/poly. Namely, instead of generating a **P**-certificate that the homomorphic evaluation of U_k was performed correctly and then computing a low-depth proof that the

[7] A Turing machine is said to be *oblivious* if the tape movements are independent of the input. Without obliviousness, one would be unable to homomorphically evaluate the Turing machine efficiently, as the location of the head of the Turing machine is encrypted.

resulting **P**-certificate properly verifies, simply generate a (large) low-depth proof of correctness of the homomorphic evaluation directly. Further, in the place of FHE, simply sample and utilize keys for a leveled FHE scheme with sufficient levels to support homomorphic evaluation of U_k. The resulting transformation $e\mathcal{O}'$ still satisfies the required correctness and security properties, but no longer achieves succinctness (i.e., the size of the obfuscated Turing machine depends polynomially on its runtime).

Corollary 2. *Based on any extractability obfuscator for the class of circuits NC^1, and* leveled *fully homomorphic encryption, there exists a (non-succinct) extractability obfuscator for P/poly.*

We refer the reader to the full version of this paper for the full construction and analysis of the bootstrapping procedure and associated corollaries.

4 Functional Witness Encryption

We put forth the notion of *functional witness encryption (FWE).* An FWE scheme enables one to encrypt a message m with respect to an NP language L, instance x and a function f, such that anyone that has, and *only* those that have, a witness w for $x \in L$ can recover $f(m, w)$. More precisely, our security definition requires that if you can distinguish encryptions of two messages m_0, m_1, then you must know a witness w for $x \in L$ such that $f(m_0, w) \neq f(m_1, w)$.

For example, an FWE scheme would allow one to encrypt the nodes of a large graph in such a way that anybody (and *only those*) who knows a clique in the graph can decrypt the labels *on the corresponding clique.*

Definition 8 (Functional Witness Encryption). *A* functional witness encryption *scheme for an NP language L (with corresponding witness relation R) and class of Turing machines $\{\mathcal{M}_k\}_{k\in\mathbb{N}}$, consists of the following two polynomial-time algorithms:*

- $\mathsf{Enc}(1^k, x, m, M)$*: On input the security parameter 1^k, an unbounded-length string x, message $m \in MSG$ for some message space MSG, and Turing machine description $M \in \mathcal{M}_k$,* Enc *outputs a ciphertext c.*
- $\mathsf{Dec}(c, w)$*: On input a ciphertext c and an unbounded-length string w,* Dec *outputs an evaluation m' or the symbol $\perp$.*

satisfying the following conditions:

Correctness: *There exists a negligible function* $\mathsf{negl}(k)$ *such that for every security parameter k, for any message $m \in MSG$, for any Turing machine $M \in \mathcal{M}_k$, and for any $x \in L$ such that $R(x, w)$ holds, we have that* $\Pr\left[\mathsf{Dec}(\mathsf{Enc}(1^k, x, m, M), w) = M(m, w)\right] = 1 - \mathsf{negl}(k)$.

Security: *For every PPT adversary $\mathcal{A}$ and polynomials $p(k), \ell(k)$, there exists a PPT extractor E and polynomial $q(k)$ s.t. for every security parameter k, pair of messages $m_0, m_1 \in MSG_k$, Turing machine $M \in \mathcal{M}_k$, string x, and auxiliary input z of length at most $\ell(k)$,*

$$\Pr\left[b \leftarrow \{0,1\}; c \leftarrow \mathsf{Enc}(1^k, x, m_b, M) : \mathcal{A}(1^k, c, z) = b\right] \geq \tfrac{1}{2} + \tfrac{1}{p(k)}$$
$$\Rightarrow \Pr\left[w \leftarrow E(1^k, p(k), x, m_0, m_1, M, z) : M(m_0, w) \neq M(m_1, w)\right] \geq \tfrac{1}{q(k)}.$$

We demonstrate that FWE is, in fact, *equivalent* to extractability obfuscation, up to a simple transformation.

Theorem 10 (Equivalence of FWE and Extractability Obfuscation). *The existence of the following two primitives is equivalent:*

1. *Succinct functional witness encryption for NP and P/poly.*
2. *Succinct extractability obfuscation for P/poly.*

Roughly, given an extractability obfuscator $e\mathcal{O}$, an FWE encryption of the message m, for the language L, instance x and function f will be the obfuscation of the program that on input w outputs $f(m, w)$ if w is a valid witness for $x \in L$. On the other hand, given a general-purpose FWE, to obfuscate a program Π, let f be the universal circuit that on input (Π, y) runs Π on input y, let L be the trivial language where every witness is valid, and output a FWE of the message Π. We defer the proof of Theorem 10 to the full version of this paper.

5 Applications to Functional Encryption

We show how to use extractability obfuscation to directly achieve (indistinguishability) functional encryption for unbounded number of key queries and with full adaptive-message security for any unbounded size message space, without relying on complexity leveraging.

The intuition behind our scheme is simple. Let the public key of the FE scheme be the verification key for a signature scheme, and let the master secret key (needed to release secret keys sk_f) be the signing key for the signature scheme. To encrypt a message m, obfuscate the program that on input f and a valid signature on f (given the public key) simply computes $f(m)$. The secret key sk_f for a function f is then simply the signature on f. (The high-level idea behind the construction is somewhat similar to the one used in [20], which uses witness encryption in combination with signature schemes to obtain simulation-based FE for a *single* function f; in contrast, we here focus on FE for an unbounded number of functions).

Proving that this construction works is somewhat subtle. In fact, to make the proof go through, we require the signature scheme in use to be of a special *delegtable* kind—namely, we require the use of *functional signatures* [7,3] (which can be constructed based on non-interactive zero-knowledge arguments of knowledge), which make it possible to delegate a signing key sk' that enables one to sign only messages that satisfy some predicate.[8] The delegation property is only used in the security reduction and, roughly speaking, makes it possible

[8] Note that functional signatures were not needed in [20], as they only consider a single key query. In our case, functional signatures are needed to answer "CCA"-type key queries.

to simulate key queries without harming security for the messages selected by the attacker.

We defer the full construction of the functional encryption scheme and proof of security to the full version.

Theorem 11. *Assume the existence of non-interactive zero-knowledge arguments of knowledge (NIZKAoK) for NP and the existence of a extractability obfuscators for P/poly. Then there exists a (fully) indistinguishability-secure functional encryption scheme for arbitrary length messages.*

6 Relating Extractability and Indistinguishability Obfuscation

A natural question is whether we can obtain extractability obfuscation from indistinguishability obfuscation. We address this question in two different settings: first directly in the context of obfuscation, and second in the language of FWE. (Recall that these two notions are equivalent when dealing with arbitrary circuits and arbitrary functions; however, when considering restricted function classes, there are interesting differences).

In Section 6.1, we demonstrate that any indistinguishability obfuscation in fact implies a weak version of extractability obfuscation, in which extraction is only guaranteed when the two circuits differ on only polynomially many inputs. In Section 6.2, we define a weaker notion of FWE mirroring the definition of indistinguishability obfuscation, and provide a transformation from any such indistinguishability FWE to standard FWE for languages with polynomially many witnesses.

The two results are incomparable, in that the former transformation (in Section 6.1) starts with a stronger assumption and yields a stronger result. Indeed, if one begins with indistinguishability FWE for all NP and P/poly, then by the equivalence of FWE and obfuscation, the former transformation yields a stronger outcome in the setting of FWE, guaranteeing indistinguishability of encryptions of messages m_0, m_1 with respect to a function f and NP statement x with potentially exponentially many witnesses, as long as only *polynomially many such witnesses w produce differing outputs $f(m_0, w) \neq f(m_1, w)$*. On the other hand, the FWE transformation (in Section 6.2) also treats the case of restricted function classes. For example, it provides a method for transforming indistinguishability FWE for the trivial function $f(m, w) = m$ to FWE for the same function f. It is easy to see that indistinguishability FWE for this particular f is equivalent to the notion of witness encryption [15], and FWE for the same f is equivalent to the notion of extractable witness encryption of [20]. The transformation in Section 6.2 thus shows how to turn witness encryption to extractable witness encryption for the case of languages with polynomially many witness.

6.1 From Indistinguishability Obfuscation to Extractability Obfuscation for Circuits with Polynomial Differing Inputs

We show that indistinguishability obfuscation directly implies a weak version of extraction obfuscation, where extraction is successful for any pair of circuits C_0, C_1 that vary on polynomially many inputs.

Definition 9 (Weak Extractability Obfuscation). *A uniform transformation $\mathcal{O}$ is a* weak extractability obfuscator *for a class of Turing machines $\mathcal{M} = \{\mathcal{M}_k\}$ if for every PPT adversary $\mathcal{A}$ and polynomial $p(k)$, there exists a PPT algorithm E and polynomials $p_E(k), t_E(k)$ for which the following holds. For every polynomial $d(k)$, for all sufficiently large k, and every pair of circuits $M_0, M_1 \in \mathcal{M}_k$* differing on at most $d(k)$ inputs*, and every auxiliary input z,*

$$\Pr\left[b \leftarrow \{0,1\}; \tilde{M} \leftarrow \mathcal{O}(1^k, C_b) : \mathcal{A}(1^k, \tilde{M}, M_0, M_1, z) = b\right] \geq \tfrac{1}{2} + \tfrac{1}{p(k)}$$
$$\implies \Pr\left[x \leftarrow E(1^k, M_0, M_1, z) : M_0(x) \neq M_1(x)\right] \geq \tfrac{1}{p_E(k)},$$

and the runtime of E is $t_E(k, d(k))$.

Theorem 12. *Let $\mathcal{O}$ be an indistinguishability obfuscator for P/poly. Then $\mathcal{O}$ is also a* weak extractability obfuscator *for P/poly.*

Denote by $n = n(k)$ the (polynomial) input length of the circuits in question. At a high level, the extractor E associated with an adversary $\mathcal{A}$ performs a form of binary search over $\{0,1\}^n$ for a desired input by considering a sequence of intermediate circuits lying "in between" C_0 and C_1. The goal is that after n iterations, E will reach a pair of circuits $C^{\mathsf{Left}}, C^{\mathsf{Right}}$ for which: (1) $\mathcal{A}$ can still distinguish between obfuscations $\{\mathcal{O}(C^{\mathsf{Left}})\}$ and $\{\mathcal{O}(C^{\mathsf{Right}})\}$, and yet (2) C^{Left} and C^{Right} are identical on all inputs except a single known x, for which $C^{\mathsf{Left}}(x) = C_0(x)$ and $C^{\mathsf{Right}}(x) = C_1(x)$. Thus, by the indistinguishability security of $\mathcal{O}$, it must be that E has extracted an input x for which $C_0(x) \neq C_1(x)$.

To demonstrate, consider the case where the circuits C_0, C_1 differ on a single unknown input x^*. In the first step, the extractor algorithm E will consider an intermediate circuit C^{Mid} equal to C_0 on the first half of its inputs, and equal to C_1 on the second half of its inputs. Then since $C^{\mathsf{Mid}}(x^*) \in \{C_0(x^*), C_1(x^*)\}$ and all three circuits agree on inputs $x \neq x^*$, it must be that C^{Mid} is *equivalent* to either C_0 or C_1. By the security of the indistinguishability obfuscator, it follows that the obfuscations of such equivalent circuits are indistinguishable. But, if an adversary $\mathcal{A}$ distinguishes between obfuscations of C_0 and C_1 with non-negligible advantage ϵ, then $\mathcal{A}$ must successfully distinguish between obfuscations of C_0 & C^{Mid} or C^{Mid} & C_1. Namely, it must be the case that $\mathcal{A}$'s distinguishing advantage is very small between one of these pairs of distributions (corresponding to the case $C^{\mathsf{Mid}} \equiv C_b$) and is nearly ϵ for the other pair of distributions (corresponding to $C^{\mathsf{Mid}} \not\equiv C_{1-b}$). Thus, by generating samples from these distributions and estimating $\mathcal{A}$'s distinguishing advantages for the two distribution pairs, E can determine whether $C^{\mathsf{Mid}} \equiv C_0$ or $C^{\mathsf{Mid}} \equiv C_1$ and, in turn, has learned whether x^* lies in the first or second half of the input space. This

process is then repeated iteratively within a smaller window (i.e., considering a new intermediate circuit lying "in between" C^{Mid} and C_b for which $C^{\mathsf{Mid}} \not\equiv C_b$). In each step, we decrease the input space by a factor of two, until x^* is completely determined.

The picture becomes more complicated, however, when there are several inputs on which C_0 and C_1 disagree. Here the intermediate circuit C^{Mid} need not agree with either C^{Left} or C^{Right} on all inputs. Thus, whereas above $\mathcal{A}$'s distinguishing advantage along one of the two paths was guaranteed to drop no more than a negligible amount, here in each step $\mathcal{A}$'s advantage could split by as much as half. At this rate, after only $\log k$ iterations, $\mathcal{A}$'s advantage will drop below usable levels, and the binary search approach will fail. Indeed, if C_0, C_1 differ on superpolynomially may inputs $d(k) \in k^{\omega(1)}$, there may not even *exist* a pair of adjacent circuits C^{Left} and C^{Right} satisfying the desired properties (1) and (2) described above. (Intuitively, for example, it could be the case that each time one evaluation is changed from $C_0(x)$ to $C_1(x)$, the adversary's probability of outputting 1 increases by the negligible amount $1/d$).

We show, however, that if there are polynomially many differing inputs $D \subset \{0,1\}^n$ for which $C_0(x) \neq C_1(x)$, then this issue can be overcome. The key insight is that, in any step of the binary search where the adversary's distinguishing advantage may split noticeably among the two possible continuing paths, this step must also split the *set of differing inputs* into two subsets: that is, the number of points d' on which C^{Left} and C^{Right} disagree is equal to the *sum* of the number of points d^L on which C^{Mid} and C^{Left} disagree and the number of points d^R on which C^{Mid} and C^{Right} disagree. Then even though the adversary's distinguishing advantage may split as $\epsilon' = \epsilon^L + \epsilon^R$, for at least one of the two paths $b \in \{L, R\}$, it must be that the ratio of $\epsilon^b/d^b \geq \epsilon'/d'$ is roughly maintained (up to a negligible amount). Since there are only polynomially many total disagreeing inputs $d(k) \in k^{O(1)}$ to start, and assuming $\mathcal{A}$ begins with non-negligible distinguishing advantage, the original ratio ϵ/d at the root node begins as a non-negligible amount. And so we are guaranteed that there exists a path down the tree for which ϵ'/d' (and, in particular, the intermediate distinguishing advantage ϵ') stays above this non-negligible amount ϵ/d. Our extractor E will find this path by simply following *all paths* which maintain distinguishing advantage above this value. By the security of the indistinguishability obfuscation scheme, there will be at most polynomially many such paths (corresponding to those terminating at the inputs $x \in D$), and all other paths in the tree will be pruned.

More specifically, our extractor E runs as follows. At the beginning of execution, it sets a fixed threshold $\mathsf{thresh} = \epsilon/dk$ based on the original (signed) distinguishing advantage ϵ of $\mathcal{A}$ and the number of inputs d on which the circuits differ (note that if $d = d(k)$ is unknown, E will repeat the whole extraction procedure with guesses k, k^2, k^{2^2}, k^{2^3}, etc, for this value). At each step it considers three circuits $C^{\mathsf{Left}}, C^{\mathsf{Mid}}, C^{\mathsf{Right}}$, and estimates $\mathcal{A}$'s (signed) distinguishing advantage between obfuscations of C^{Left} & C^{Mid} and of C^{Mid} & C^{Right}, using repeated sampling with sufficiently low error ($\mathsf{err} = \epsilon/dk^2$). For each pair that yields

distinguishing probability above thresh (possibly neither, one, or both pairs), E recurses by repeating this process at a circuit lying between the relevant window. More explicitly, if the left pair yields sufficient distinguishing advantage, then E will repeat the process for the triple of circuits $C^{\mathsf{Left}}, C', C^{\mathsf{Mid}}$ for the circuit C' "halfway between" $C^{\mathsf{Left}}, C^{\mathsf{Mid}}$; analogous for the right pair; if both surpass threshold, E repeats for both; and if neither surpass threshold, then E will not continue down this path of the binary search.

In the full version, we prove that for appropriate choice of threshold, E will only ever visit polynomially many nodes in the binary search tree, and will be guaranteed to find a complete path for which $\mathcal{A}$'s distinguishing advantage maintains above threshold through all n steps down the tree (thus specifying a desired n-bit distinguishing input).

Note that Theorem 12 implies, for example, that for the class of polynomial multipoint locker functions (i.e., functions evaluating to nonzero bit strings at polynomially many hidden points), indistinguishability obfuscation is *equivalent* to extractability obfuscation.

6.2 From Indistinguishability FWE to FWE for Languages with Polynomial Witnesses

We now address this question in the language of FWE.

Mirroring the definition of indistinguishability obfuscation, we define a weaker notion of FWE—which we refer to as *indistinguishability FWE*—which only requires that if $f(m_0, w) = f(m_1, w)$ for *all* witnesses w for $x \in L$, then encryptions of m_0 and m_1 are indistinguishable. Recall that, in contrast, the stronger notion requires that if you can distinguish between encryptions of m_0 and m_1 you must know a witness on which they differ.

Definition 10 (Indistinguishability FWE). *An* indistinguishability functional witness encryption *(iFWE) scheme for an NP language L and class of functions $\mathcal{F} = \{F_k\}$ consists of encryption and decryption algorithms* $\mathsf{Enc}, \mathsf{Dec}$ *with the same syntax as standard FWE, satisfying the same correctness property, and the following (weaker) security property:*

(Indistinguishability) security: *For every PPT adversary $\mathcal{A}$ and polynomial $\ell(\cdot)$, there exists a negligible function $\nu(\cdot)$ such that for every security parameter k, every function $f \in F_k$, messages $m_0, m_1 \in MSG_k$, string x, and auxiliary information z of length at most $\ell(k)$ for which $f(m_0, w) = f(m_1, w)$ for* every *witness w of $x \in L$,*

$$\Big|\Pr\left[\mathcal{A}(1^k, \mathsf{Enc}(1^k, x, m_0, f), z) = 1\right] - \Pr\left[\mathcal{A}(1^k, \mathsf{Enc}(1^k, x, m_1, f), z) = 1\right]\Big| < \nu(k).$$

Using the same transformation as in the Extractability Obfuscation-FWE equivalence (see Theorem 10), it can be seen that iFWE for P/poly and NP is directly equivalent to indistinguishability obfuscation for P/poly. We now consider the question of whether we can turn any iFWE into an FWE. We provide an affirmative answer for two restricted cases.

The first result is derived from the transformation from the previous section, combined with the simple extractability obfuscation-to-FWE equivalence transformation (see Theorem 10). Loosely, it says that from iFWE for P/poly, we can obtain a weak form of FWE where (extraction) security holds as long as the function $f(m, w)$ is not "too sensitive" to m: i.e., if for any two messages m_0, m_1 there are only polynomially many witnesses w for which $f(m_0, w) \neq f(m_1, w)$. For example, this captures functions f that rarely output nonzero values. Returning to the example of encrypting data m associated with nodes of a social network, we could allow someone holding clique w to learn whether the nodes in this clique satisfy some chosen rare property (e.g., contains someone with a rare disease, all have the same birthday, etc). Then, while there may be many cliques (corresponding to several, even exponentially many, witnesses w), it will hold that $f(m, w) = 0$ for all but polynomially many w.

As a special case, if the language has only polynomially many witnesses for each statement, then this property holds for any function class.

Definition 11. *We say a class of functions $\mathcal{F} = \{F_k\}$ has* t-bounded sensitivity *w.r.t. message space MSG and* NP *language L (with relation R), if for every $f \in F_k$, every $m_0, m_1 \in MSG$, and every $x \in \{0,1\}^*$ there are at most $t(|x|)$ witnesses w s.t. $R(x, w) = 1$ and $f(m_0, w) \neq f(m_1, w)$.*

Corollary 3. *Suppose there exists iFWE for* NP *and P/poly. Then for any polynomial $t(\cdot)$, there exist FWE schemes for any class of functions $\mathcal{F} = \{F_k\}$, message space MSG, and* NP *language L, for which $\mathcal{F}$ has t-bounded sensitivity with respect to MSG and L.*

The second result considers iFWE for general function classes (instead of just P/poly), but restricts to NP languages with polynomial witnesses. In the encrypted social network example, this allows basing on a weaker assumption (not requiring the iFWE scheme to support all P/poly), but would restrict to social networks with only polynomially many cliques. The transformation preserves the supported function class: For example, given iFWE for the singleton function class $\{f(m, w) = m\}$ (corresponding to standard witness encryption), one obtains standard FWE for the same class (i.e., *extractable* witness encryption). This result requires a new approach, and makes use of techniques in error-correcting codes.

Definition 12. *Let L be an* NP *language with corresponding relation R. We say that L has t-bounded witness if for every $x \in \{0,1\}^*$, there are at most $t(|x|)$ distinct witnesses w such that $R(x, w) = 1$.*

Theorem 13. *For every function class $\mathcal{F} = \{F_k\}$ and polynomial $t(\cdot)$, if there exist indistinguishability functional witness encryption schemes for $\mathcal{F}$ and every t-bounded witness* NP *language, then for every t-bounded witness* NP *language L (with corresponding relation R), there exists a functional witness encryption schemes for $\mathcal{F}$ and L.*

Proof. Let L be a t-bounded witness NP language with corresponding relation R for some polynomial $t(\cdot)$. Define $q(\cdot)$ such that for every $k \in \mathbb{N}$, $q(k)$ is the smallest prime $\geq 8t(k)$. Assume without loss of generality (by padding) that any witness of any $x \in L$ has length $u(|x|)$ for some polynomial u. To construct a functional witness encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ for L and $\mathcal{F}$, we consider the following NP language L'.

$$L' = \{(x, r, a) : \exists w \in \{0,1\}^{u(|x|)} \text{ s.t. } (R(x,w) = 1) \wedge (r \in \mathbb{F}_{q(|x|)}^{u(|x|)}) \wedge (\langle r, w \rangle = a)\},$$

where $\mathbb{F}_q = \{0, \ldots, q-1\}$ is the prime field of size q and $\langle \cdot, \cdot \rangle$ denotes inner product over $\mathbb{F}_q^u$.

Let $(\mathsf{Enc}', \mathsf{Dec}')$ be a indistinguishability FWE scheme for L' and $\mathcal{F}$. We construct a FWE scheme $(\mathsf{Enc}, \mathsf{Dec})$ for L and $\mathcal{F}$ as follows.

- $\mathsf{Enc}(1^k, x, m, f)$: On input security parameter 1^k, statement $x \in \{0,1\}^*$, message $m \in MSG_k$, and function $f \in \mathcal{F}_k$, Enc generates c as:
  - Let $q = q(|x|)$ and $u = u(|x|)$. Sample $r \leftarrow \mathbb{F}_q^u$ uniformly random.
  - For every $a \in \mathbb{F}_q$, compute $c_a = \mathsf{Enc}'(1^k, (x, q, r, a), m, f)$.
  - Output $c = \{c_a\}_{a \in \mathbb{F}_q}$.
- $\mathsf{Dec}(c, w)$: On input a ciphertext $c = \{c_a\}_{a \in \mathbb{F}_q}$ and a witness $w \in \{0,1\}^*$, Dec runs $\mathsf{Dec}'(c_a, w)$ for every $a \in \mathbb{F}_q$. If there exists some a such that $\mathsf{Dec}'(c_a, w) \neq \perp$, then output the first non-$\perp$ $\mathsf{Dec}'(c_a, w)$. Otherwise, output $\perp$.

It is not hard to see that correctness of $(\mathsf{Enc}', \mathsf{Dec}')$ implies correctness of $(\mathsf{Enc}, \mathsf{Dec})$: For every k, x, m, f, w, if w is a witness for $x \in L$, then there exists some $a \in \mathbb{F}_q$ such that w is a witness for $(x, q, r, a) \in L'$, and for the first such a, by the correctness of $(\mathsf{Enc}', \mathsf{Dec}')$, $\mathsf{Dec}'(c_a, w) = f(m, w)$ with $1 - \mathsf{negl}(k)$ probability, which implies that $\mathsf{Dec}(\mathsf{Enc}(1^k, x, m, f), w)$ output $f(m, w)$ with $1 - \mathsf{negl}(k)$ probability as well.

We refer the reader to the full version of this paper for the proof of security of $(\mathsf{Enc}, \mathsf{Dec})$. At a high level, we show that if an adversary $\mathcal{A}$ can distinguish $\mathsf{Enc}(1^k, x, m_0, f)$ and $\mathsf{Enc}(1^k, x, m_1, f)$ with a non-negligible advantage, then there is a non-negligible fraction of $r \in \mathbb{F}_q^u$ such that we learn non-trivial information about the value of $\langle r, w \rangle$ for some witness w such that $f(m_0, w) \neq f(m_1, w)$. Note that a linear function $g_w(r) := \langle r, w \rangle$ can be viewed as a q-ary Hadamard code of w. The non-trivial information allows us to obtain a (randomized) function $h(r)$ that agree with $g_w(r)$ on non-negligibly more than $1/q$ fraction of points. We can then apply the local list-decoding algorithm of Goldreich, Rubinfield, and Sudan [19] to recover w.

References

1. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. Cryptology ePrint Archive, Report 2013/631 (2013)
2. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM 59(2), 6 (2012)
3. Bellare, M., Fuchsbauer, G.: Policy-based signatures. Cryptology ePrint Archive, Report 2013/413 (2013)

4. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for snarks and proof-carrying data. In: STOC, pp. 111–120 (2013)
5. Bitansky, N., Chiesa, A., Ishai, Y., Paneth, O., Ostrovsky, R.: Succinct non-interactive arguments via linear interactive proofs. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 315–333. Springer, Heidelberg (2013)
6. Boneh, D., Sahai, A., Waters, B.: Functional encryption: a new vision for public-key cryptography. Commun. ACM 55(11), 56–64 (2012)
7. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. Cryptology ePrint Archive, Report 2013/401 (2013)
8. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. Electronic Colloquium on Computational Complexity (ECCC) 18, 111 (2011)
9. Brakerski, Z., Rothblum, G.: Virtual black-box obfuscation for all circuits via generic graded encoding. Cryptology ePrint Archive, Report 2013/563 (2013)
10. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106 (2011)
11. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. Cryptology ePrint Archive, Report 2013/541 (2013)
12. Canetti, R., Lin, H., Paneth, O.: Public-coin concurrent zero-knowledge in the global hash model. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 80–99. Springer, Heidelberg (2013)
13. Chung, K.-M., Lin, H., Pass, R.: Constant-round concurrent zero knowledge from falsifiable assumptions. Cryptology ePrint Archive, Report 2012/563 (2012)
14. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS (2013)
15. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: STOC, pp. 467–476 (2013)
16. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct nizks without pcps. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (2013)
17. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)
18. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013)
19. Goldreich, O., Rubinfeld, R., Sudan, M.: Learning polynomials with queries: The highly noisy case. SIAM J. Discrete Math. 13(4), 535–570 (2000)
20. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to run turing machines on encrypted data. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 536–553. Springer, Heidelberg (2013)
21. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 321–340. Springer, Heidelberg (2010)
22. Hada, S.: Zero-knowledge and code obfuscation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 443–457. Springer, Heidelberg (2000)
23. Hohenberger, S., Sahai, A., Waters, B.: Replacing a random oracle: Full domain hash from indistinguishability obfuscation. Cryptology ePrint Archive, Report 2013/509 (2013), `http://eprint.iacr.org/`

24. Horwitz, J., Lynn, B.: Toward hierarchical identity-based encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)
25. Lipmaa, H.: Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 169–189. Springer, Heidelberg (2012)
26. Micali, S.: Computationally sound proofs. SIAM J. Comput. 30(4), 1253–1298 (2000)
27. Naor, M.: On cryptographic assumptions and challenges. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 96–109. Springer, Heidelberg (2003)
28. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010), http://eprint.iacr.org/
29. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: Deniable encryption, and more. Cryptology ePrint Archive, Report 2013/454 (2013), http://eprint.iacr.org/