From Cryptomania to Obfustopia through Secret-Key Functional Encryption

Nir Bitansky* Ryo Nishimaki† Alain Passelègue‡ Daniel Wichs§

June 14, 2016

Abstract

Functional encryption lies at the frontiers of current research in cryptography; some variants have been shown sufficiently powerful to yield indistinguishability obfuscation (IO) while other variants have been constructed from standard assumptions such as LWE. Indeed, most variants have been classified as belonging to either the former or the latter category. However, one mystery that has remained is the case of *secret-key functional encryption* with an unbounded number of keys and ciphertexts. On the one hand, this primitive is not known to imply anything outside of minicrypt, the land of secret-key crypto, but on the other hand, we do no know how to construct it without the heavy hammers in obfustopia.

In this work, we show that (subexponentially secure) secret-key functional encryption is powerful enough to construct indistinguishability obfuscation if we additionally assume the existence of (subexponentially secure) plain public-key encryption. In other words, secret-key functional encryption provides a bridge from cryptomania to obfustopia.

On the technical side, our result relies on two main components. As our first contribution, we show how to use secret key functional encryption to get "exponentially-efficient indistinguishability obfuscation" (XIO), a notion recently introduced by Lin et al. (PKC '16) as a relaxation of IO. Lin et al. show how to use XIO and the LWE assumption to build IO. As our second contribution, we improve on this result by replacing its reliance on the LWE assumption with any plain public-key encryption scheme.

Lastly, we ask whether secret-key functional encryption can be used to construct public-key encryption itself and therefore take us all the way from minicrypt to obfustopia. A result of Asharov and Segev (FOCS '15) shows that this is not the case under black-box constructions, even for exponentially secure functional encryption. We show, through a non-black box construction, that subexponentially secure-key functional encryption indeed leads to public-key encryption. The resulting public-key encryption scheme, however, is at most quasi-polynomially secure, which is insufficient to take us to obfustopia.

*MIT, nirbitan@csail.mit.edu Supported by an IBM DARPA grant and an NJIT DARPA grant.

†NTT Secure Platform Laboratories, nishimaki.ryo@lab.ntt.co.jp This work was done in part while the author was visiting Northeastern University.

‡ENS, alain.passelegue@ens.fr This work was done in part while the author was visiting Northeastern University. Supported in part by the *Direction Générale de l'Armement*.

§Northeastern University, wichs@ccs.neu.edu Supported in part by NSF grants CNS-1347350, CNS-1314722, CNS-1413964.

Contents

1 Introduction

The concept of *functional encryption* [BSW11, O'N10] extends that of traditional encryption by allowing the distribution of *functional decryption keys* that reveal specified functions of encrypted messages, but nothing beyond. This concept is one of the main frontiers in cryptography today. It offers tremendous flexibility in controlling and computing on encrypted data, is strongly connected to the holy grail of program obfuscation [AJ15, BV15, LPST16b], and for many problems, may give superior solutions to obfuscation-based ones [GPS15, GPSZ16]. Accordingly, recent years have seen outstanding progress in the study of functional encryption, both in constructing functional encryption schemes and in exploring different notions, their power, and the relationship amongst them (see for instance, [SS10, GVW12, BO13, GKP+13, AGVW13, DIJ+13, GGH+13, BGG+14, GGG+14, BS15, GJKS15, KSY15, BLR+15, GVW15, ABSV15, Wat15, BGJS15, BKS16, AJS15, AS16, GGHZ16, Lin16] and many more).

One striking question that has yet to be solved is *the gap between public-key and secret-key functional encryption schemes. In particular, does any secret-key scheme imply a public-key one?*

The answer to this question is nuanced and seems to depend on certain features of functional encryption schemes, such as the number of functional decryption keys and number of ciphertexts that can be released. For functional encryption schemes that only allow the release of an *a-priori bounded* number of functional keys (often referred to as *bounded collusion*), we know that the above gap is essentially the same as the gap between plain (rather than functional) secret-key encryption and public-key encryption, and should thus be as hard to bridge. Specifically, in the secret-key setting, such schemes supporting an unbounded number of ciphertexts can be constructed assuming low-depth pseudorandom generators (or just one-way functions in the single-key case) [SS10, GVW12]. These secret-key constructions are then converted to public-key ones, relying on (plain) public-key encryption (and this is done quite directly by replacing invocations of a secret-key encryption scheme with invocations of a public-key one.) The same state of affairs holds when reversing the roles and considering a bounded number of ciphertexts and an unbounded number of keys [SS10, GVW12]. In other words, in the terminology of Impagliazzo [Imp95], if the number of keys or ciphertexts is a-priori bounded, then symmetric-key functional encryption lies in minicrypt and public-key functional encryption lies in cryptomania.

For functional encryption schemes supporting an unbounded (polynomial) number of keys and unbounded number of ciphertexts, which will be the default notion throughout the rest of the paper, the question is far less understood. In the public-key setting, such functional encryption schemes with subexponential security are known to imply indistinguishability obfuscation [AJ15, BV15, AJS15]. In contrast, Bitansky and Vaikuntanathan [BV15] show that their construction of indistinguishability obfuscation using functional encryption may be insecure when instantiated with a secret-key functional encryption scheme. In fact, secret-key functional encryption schemes (even exponentially secure ones) are not known to imply any cryptographic primitive beyond those that follow from one-way functions. In the terminology of Impagliazzo [Imp95], as far as we know the two notions of functional encryption may correspond to opposite extremes of the complexity spectrum: on one side public-key schemes inherently correspond to *obfustopia*, the world where indistinguishability obfuscation exists, and on the other side secret-key schemes may lie in *minicrypt*, where one-way functions exist, but there is even no (plain) public-key encryption.

One piece of evidence that may support such a view of the world is given by Asharov and Segev [AS15] who show that there do not exist *fully black-box* constructions of *plain* public-key encryption from secret-key functional encryption, even if the latter is exponentially secure. Still, while we may hope that such secret-key schemes could be constructed from significantly weaker assumptions than needed for public-key schemes, so far no such construction has been exhibited — all known constructions live in obfustopia.

1.1 Our Contributions

In this work, we shed new light on the question of secret-key vs public-key functional encryption (in the multi-key, multi-ciphertext setting). Our main result bridges the two notions based on (plain) public-key encryption.

Theorem 1.1 (Informal)**.** *Assuming secret-key functional encryption and plain public-key encryption that are both subexponentially secure, there exists indistinguishability obfuscation, and in particular, also public-key functional encryption.*

In the terminology of Impagliazzo's complexity worlds: *secret-key functional encryption would turn cryptomania, the land of public-key encryption, into obfustopia.* This puts in new perspective the question of constructing such secret-key schemes from standard assumptions — any such construction would lead to indistinguishability obfuscation from standard assumptions.

The above result still does not settle the question of whether secret-key functional encryption on its own implies (plain) public-key encryption. Here we show that assuming subexponentially-secure secret-key functional encryption and (almost) exponentially-secure one-way functions, there exists (polynomially-secure) public-key encryption.

Theorem 1.2 (Informal)**.** *Assuming subexponentially-secure secret-key functional encryption and $2^{n/\log\log n}$-secure one-way functions, there exists (polynomially-secure) public-key encryption.*

The resulting public-key encryption is not strong enough to take us to obfustopia. Concretely, the constructed scheme is not subexponentially secure as required by our first theorem — it can be *quasi-polynomially* broken. Nevertheless, the result does show that the black-box barrier shown by Asharov and Segev [AS15], which applies even if the underlying secret-key functional encryption scheme and one-way functions are *exponentially secure*, can be circumvented. Indeed, our construction uses the functional encryption scheme in a non-black-box way (see further details in the technical overview section below).

1.2 A Technical Overview

We now provide an overview of the main steps and ideas leading to our results.

Key observation: from SKFE to (strong) exponentially-efficient IO. Our first observation is that secret-key functional encryption (or SKFE in short) implies a weak form of indistinguishability obfuscators termed by Lin, Pass, Seth, and Telang [LPST16a] exponentially-efficient indistinguishability obfuscation (XIO). Like IO, this notion preserves the functionality of obfuscated circuits and guarantees that obfuscations of circuits of the same size and functionality are indistinguishable. However, in terms of efficiency the XIO notion only requires that an obfuscation $\widetilde{C}$ of a circuit $C : \{0,1\}^n \to \{0,1\}^m$ is just mildly smaller than its truth table, namely $|\widetilde{C}| \leq 2^{\gamma n} \cdot \mathrm{poly}(|C|)$, for some compression factor $\gamma < 1$, and a fixed polynomial poly, rather than the usual requirement that *the time to obfuscate*, and in particular the size of $\widetilde{C}$, are polynomial in $|C|$. We show that SKFE implies a slightly stronger notion than XIO where the time to obfuscate C is bounded by $2^{\gamma n} \cdot \mathrm{poly}(|C|)$. We call this notion strong exponentially-efficient indistinguishability obfuscation (SXIO). (We note that, for either XIO or SXIO, we shall typically be interested in circuits over some polynomial size domain, which could be much larger than the circuit itself, e.g., $\{0,1\}^n$ where $n = 100 \log |C|$.)

Proposition 1.1 (Informal)**.**

1. *For any constant $\gamma < 1$, there exists a transformation from SKFE to SXIO with compression factor γ.*

2. *For some subconstant $\gamma = o(1)$, there exists a transformation from subexponentially-secure SKFE to polynomially-secure SXIO with compression factor γ.*

We add more technical details regarding the proof of the above SXIO proposition later on. Both of our theorems stated above rely on the constructed SXIO as a main tool. We next explain, still at a high-level, how the first theorem is obtained. We then dive into further technical details about the proof of this theorem as well as the proof of the second theorem.

From SXIO to IO through public-key encryption. Subexponentially-secure SXIO (or even XIO) schemes with a constant compression factor (as in Proposition 1.1) are already shown to be quite strong in [LPST16a] — assuming subexponential hardness of Learning with Errors (LWE) [Reg05], they imply IO.

Corollary 1.1 (of Proposition 1.1 and [LPST16a])**.** *Assuming SKFE and LWE, both subexponentially secure, there exists IO.*

We go beyond the above corollary, showing that LWE can be replaced with a generic assumption — the existence of (plain) public-key encryption schemes. The transformation of [LPST16a] from LWE and XIO to IO, essentially relies on LWE to obtain a specific type of public-key functional encryption (PKFE) with certain succinctness properties. We show how to construct such PKFE from public-key encryption and SXIO. More details follow.

Concretely, the notion considered is of PKFE schemes that support a *single decryption key*. Furthermore, the time complexity of encryption is bounded by roughly $s^\beta \cdot d^{O(1)}$, where s and d are the size and depth of the circuit computing the function, and $\beta < 1$ is some compression factor. We call such schemes *weakly succinct PKFE schemes*. A weakly succinct PKFE for *boolean* functions (i.e., functions with a single output bit) is constructed by Goldwasser et al. [GKP$^+$13] from (subexponentially-hard) LWE; in fact, the Goldwasser et al. construction has no dependence at all on the circuit size s (namely, $\beta = 0$).

Lin et al. [LPST16a] then show a transformation, relying on XIO, that extends the class of functions also to functions with a long output, rather than just boolean ones. (Their transformation is stated for the case that $\beta = 0$ assuming any constant XIO compression factor $\gamma < 1$, but can be extended to also work for any sufficiently small constant compression factor β for the PKFE.) Such weakly-succinct PKFE schemes can then be plugged in to the transformations of [AJ15, BV15, LPST16b] to obtain full-fledged IO. [1]

We follow a similar blueprint. We first construct weakly-succinct PKFE for functions with a single output bit based on SXIO and PKE, rather than LWE (much of the technical effort in this work lies in this construction). We then bootstrap the construction to deal with multibit functions using (a slightly augmented version of) the transformation from [LPST16a].

Proposition 1.2 (Informal)**.** *For any $\beta = \Omega(1)$, assuming PKE and SXIO with a small enough constant compression factor γ, there exists a single-key weakly-succinct PKFE scheme with compression factor β (for functions with long output).*

A Closer Look into the Techniques

We now provide further details regarding the proofs of the above Propositions 1.1 and 1.2 as well as the proof of Theorem 1.2.

[1] The above is a slightly oversimplified account of [LPST16a]. They also rely on LWE to deduce the existence of puncturable PRFs in $\mathbf{NC}^1$ and show their transformation starting from weakly-succinct PKFE for functions in $\mathbf{NC}^1$. We avoid the reliance on puncturable PRFs in $\mathbf{NC}^1$ by constructing weakly-succinct PKFE for functions with no depth restriction, at the expense of allowing the complexity of encryption to scale polynomially in the depth. This is still sufficient for [BV15, Section 3.2].

SKFE to SXIO: the basic idea. To convey the basic idea behind the transformation, we first describe a construction of SXIO with compression $\gamma = 1/2$. We then explain how to extend it to obtain the more general form of Proposition 1.1.

Recall that in an SKFE scheme, first a master secret key MSK is generated, and can then be used to:

- encrypt (any number of) plaintext messages,
- derive (any number of) functional keys.

The constructed obfuscator $\mathsf{sxi}\mathcal{O}$ is given a circuit C defined on domain $\{0,1\}^n$, where we shall assume for simplicity that the input length is even (this is not essential), and works as follows:

- For every $x \in \{0,1\}^{n/2}$, computes a ciphertext CT_x encrypting the circuit $C_x(\cdot)$ that given input $y \in \{0,1\}^{n/2}$, returns $C(x,y)$.
- For every $y \in \{0,1\}^{n/2}$, derives a functional decryption key SK_y for the function $U_y(\cdot)$ that given as input a circuit D of size at most $\max_x |C_x|$, returns $D(y)$.
- Outputs $\widetilde{C} = \left(\{\mathsf{CT}_x\}_{x\in\{0,1\}^{n/2}}, \{\mathsf{SK}_y\}_{y\in\{0,1\}^{n/2}}\right)$ as the obfuscation.

To evaluate $\widetilde{C}$ on input $(x,y) \in \{0,1\}^n$, simply decrypt

$$\mathsf{Dec}(\mathsf{SK}_y, \mathsf{CT}_x) = U_y(C_x) = C_x(y) = C(x,y) \ .$$

Indeed, the required compression factor $\gamma = 1/2$ is achieved. Generating each ciphertext is proportional to the size of the message $|C_x| = \tilde{O}(|C|)$ and some fixed polynomial in the security parameter λ. Similarly the time to generate each functional key is proportional to the size of the circuit $|U_y| = \tilde{O}(|C|)$ and some fixed polynomial in the security parameter λ. Thus overall, the time to generate $\widetilde{C}$ is bounded by $2^{n/2} \cdot \mathrm{poly}(|C|, \lambda)$.

The indistinguishability guarantee follows easily from that of the underlying SKFE. Indeed, SKFE guarantees that for any two sequences $\vec{m} = \{m_i\}$ and $\vec{m}' = \{m'_i\}$ of messages to be encrypted and any sequence of functions $\{f_i\}$ for which keys are derived, encryptions of the $\vec{m}$ are indistinguishable from encryptions of the $\vec{m}'$, provided that the messages are not "separated by the functions", i.e. $f_j(m_i) = f_j(m'_i)$ for every (i,j). In particular, any two circuits C and C' that have equal size and functionality will correspond to such two sequences of messages $\{C_x\}_{x\in\{0,1\}^{n/2}}$ and $\{C'_x\}_{x\in\{0,1\}^{n/2}}$, whereas $\{U_y\}_{y\in\{0,1\}^n}$ are indeed functions such that $U_y(C_x) = C(x,y) = C'(x,y) = U_y(C'_x)$ for all (x,y). (The above argument works even given a very weak selective security definition where all messages and functions are chosen by the attacker ahead of time.)

As said, the above transformation achieves compression factor $\gamma = 1/2$. While such compression is sufficient for example to obtain IO based on LWE, it will not suffice for our two Theorems 1.1, 1.2 (for the first we will need γ to be a smaller constant, and for the second we will need it to even be slightly subconstant). To prove Proposition 1.1 in its more general form, we rely on a result by Brakerski, Komargodski, and Segev [BKS16] that shows how to convert any SKFE into a t-input SKFE. A t-input scheme allows to encrypt a tuple of messages $(m_1, \ldots, m_t)$ each independently, and derive keys for t-input functions $f(m_1, \ldots, m_t)$. In their transformation, starting from a multi-key SKFE results in a multi-key t-input SKFE.

The general transformation then follows naturally. Rather than arranging the input space in a 2-dimensional cube $\{0,1\}^{n/2} \times \{0,1\}^{n/2}$ as we did before with a 1-input scheme, given a t-input scheme we can arrange it in a $(t+1)$-dimensional cube $\{0,1\}^{n/(t+1)} \times \cdots \times \{0,1\}^{n/(t+1)}$, and we will accordingly

get compression $\gamma = 1/(t+1)$. The only caveat is that the BKS transformation incurs a security loss and blowup in the size of the scheme that can grow doubly exponentially in t. As long as t is constant the security loss and blowup are fixed polynomials. The transformation can also be invoked for slightly super-constant t (double logarithmic) assuming subexponential security of the underlying 1-input SKFE (giving rise to the second part of Proposition 1.1).

We remark that previously Goldwasser et al. [GGG+14] showed that t-input SKFE for polynomial t directly gives full-fledged IO. We demonstrate that even when t is small (even constant), t-input SKFE implies a meaningful obfuscation notion such as SXIO.

From SXIO and PKE to weakly succinct PKFE: main ideas. We now describe the main ideas behind our construction of a single-key weakly succinct PKFE. We shall focus on the main step of obtaining such a scheme for functions with a single output bit.[2]

Our starting point is the single-key PKFE scheme of Sahai and Seyalioglu [SS10] based on Yao's garbled circuit method [Yao82]. Their scheme basically works as follows (we assume basic familiarity with the garbled circuit method):

- The master public key MPK consists of L pairs of public keys $\left\{\mathsf{PK}_i^0, \mathsf{PK}_i^1\right\}_{i\in L}$ for a (plain) public-key encryption scheme.
- A functional decryption key SK_f for a function (circuit) f of size L consists of the secret decryption keys $\{\mathsf{SK}_i^{f_i}\}_{i\in L}$ corresponding to the above public keys, according to the bits of f's description.
- To encrypt a message m, the encryptor generates a garbled circuit $\widehat{U}_m$ for the universal circuit U_m that given f, returns $f(m)$. It then encrypts the corresponding input labels $\{k_i^0, k_i^1\}_{i\in L}$ under the corresponding public keys.
- The decryptor in possession of SK_f can then decrypt to obtain the labels $\{k_i^{f_i}\}_{i\in L}$ and decode the garbled circuit to obtain $U_m(f) = f(m)$.

Selective security of this scheme (where the function f and all messages are chosen ahead of time) follows from the semantic security of PKE and the garbled circuit guarantee which says that $\widehat{U}_m, \{k_i^{f_i}\}_{i\in L}$ can be simulated from $f(m)$.

The scheme is indeed *not* succinct in any way. The complexity of encryption and even the size of the ciphertext grows with the complexity of f. Nevertheless, it does seem that the encryption process has a much more succinct representation. In particular, computing a garbled circuit is a *decomposable* process — each garbled gate in $\widehat{U}_m$ depends on a single gate in the original circuit U_m and a small amount of randomness (for computing the labels corresponding to its wires). Furthermore, the universal circuit U_m itself is also decomposable — there exists a small (say, $\mathrm{poly}(|m|, \log L)$-sized) circuit that given i can output the i-th gate in U_m along with its neighbours. The derivation of randomness itself can also be made decomposable using a pseudorandom function. All in all, there exists a small ($\mathrm{poly}(|m|, \log L, \lambda)$-size, for security parameter λ), *decomposition circuit* $U^{\mathsf{de}}_{m,K}$ associated with a key $K \in \{0,1\}^\lambda$ for a pseudorandom function that can produce the ith garbled gate/input-label given input i.

[2]Extending this to functions with multibit output is then done, based on SXIO, using a transformation of [LPST16a]. Concretely, given an m-bit output function $f(x)$ we consider a new single bit function $g_f(x,i)$ that returns the ith bit of $f(x)$. The function key is then derived for the boolean function g_f. The new encryption algorithm, for message x, produces an SXIO obfuscation of a circuit that given $i \in [m]$ uses the old encryption scheme to encrypt (m,i), deriving randomness using a puncturable PRF. The security of the construction is proven as in [LPST16a] based on a probabilistic IO argument [CLTV15]. (Mild) efficiency of the encryption then follows from the mild efficiency of the SXIO and PKFE with related (constant) compression factors.

Yet, the second part of the encryption process, where the input labels $\{k_i^0, k_i^1\}_{i\in L}$ are encrypted under the corresponding public keys $\left\{\mathsf{PK}_i^0, \mathsf{PK}_i^1\right\}_{i\in L}$, may not be decomposable at all. Indeed, in general, it is not clear how to even compress the representation of these $2L$ public-keys. In this high-level exposition, let us make the simplifying assumption that we have at our disposal a *succinct identity-based-encryption (IBE) scheme*. Such a scheme has a single public-key PK that allows to encrypt a message to an identity $\mathsf{id} \in \mathcal{ID}$ taken from an identity space $\mathcal{ID}$. Those in possession of a corresponding secret key $\mathsf{SK}_{\mathsf{id}}$ can decrypt and others learn nothing. Succinctness means that the complexity of encryption may only grow mildly in the size of the identity space. Concretely, by a factor of $|\mathcal{ID}|^\gamma$ for some small constant $\gamma < 1$. In the body, we show that such a scheme can be constructed from (plain) public-key encryption and SXIO (the construction relies on standard "puncturing techniques" and is pretty natural, see Section 5.1 for more details).

Equipped with such an IBE scheme, we can now augment the Sahai-Seyalioglu scheme to make sure that the entire encryption procedure is decomposable. Concretely, we will consider the identity space $\mathcal{ID} = [L] \times \{0, 1\}$, augment the public key to only include the IBE's public key PK, and provide the decryptor with the identity keys $\{\mathsf{SK}_{(i,f_i)}\}_{i\in L}$. Encrypting the input labels $\{k_i^0, k_i^1\}_{i\in L}$ will now be done by simply encrypting to the corresponding identities $\{(i, 0), (i, 1)\}_{i\in L}$. This part of the encryption can now also be described by a small (say $L^\gamma \cdot \mathrm{poly}(\lambda, \log L)$-size) decomposition circuit $E^{\mathsf{de}}_{K,K',\mathsf{PK}}$ that has the PRF key K to derive input labels, the IBE public key PK, and another PRF key K' to derive randomness for encryption. Given an identity (i, b), it generates the corresponding encrypted input label.

At this point, a natural direction is to have the encryptor send a *compressed* version of the Sahai-Seyalioglu encryption, by first using SXIO to shield the two decomposition circuits $E^{\mathsf{de}}_{K,K',\mathsf{PK}}, U^{\mathsf{de}}_{m,K}$ and then sending the two obfuscations. Indeed, decryption can be done just as before by first reconstructing the expanded garbled circuit and input labels and then proceeding as before. Also, in terms of encryption complexity, provided that the IBE compression factor γ is a small enough constant, the entire encryption time will scale only sublinearly in the function's size $|f| = L$ (i.e., with L^β for some constant $\beta < 1$).

The only question is of course security. It is not too hard to see that if the decomposition circuits $E^{\mathsf{de}}_{K,K',\mathsf{PK}}, U^{\mathsf{de}}_{m,K}$ are given as black-boxes then security is guaranteed just as before. The challenge is to prove security relying only on the indistinguishability guarantee of SXIO. A somewhat similar challenge is encountered in the work of Bitansky et al. [BGL$^+$15] when constructing *succinct randomized encodings*. In their setting, they obfuscate (using standard IO rather than SXIO) a decomposition circuit $C^{\mathsf{de}}_{x,K}$ (analogous to our $U^{\mathsf{de}}_{m,K}$) that computes the garbled gates of some succinctly represented long computation.

As already demonstrated in [BGL$^+$15], proving the security of such a construction is rather delicate. As in the standard setting of garbled circuits, the goal is to gradually transition through a sequence of hybrids, from a real garbled circuit (that depends on the actual computation) to a simulated garbled circuit that depends just on the result of the computation. However, unlike the standard setting, here each of these hybrids should be generated by a *hybrid obfuscated decomposition circuit* and the attacker should not be able to tell them apart. As it turns out, "common IO gymnastics" are insufficient here, and we need to rely on the specific hybrid strategy used to transition between the different garbling modes is the proof of security for standard garbled circuits. One feature of the hybrid strategy which is dominant in this context is the amount of information that hybrid decomposition circuits need to maintain about the actual computation. Indeed, as the amount of this information grows so will the size of these decomposition circuits as will the size of the decomposition circuits in the actual construction (that will have to be equally padded to preserve indistinguishability).

Bitansky et al. show a hybrid strategy where the amount of information scales with the *space* of the computation (or *circuit width*). Whereas in their context this is meaningful (as the aim is to save comparing to the *time* of the computation), in our context this is clearly insufficient. Indeed, in our case the space of

the computation given by the universal circuit U_m and the function f can be as large as f's description. Instead, we invoke a different hybrid strategy by Hemenway et al. [HJO$^+$15] that scales only with the *circuit depth*. Indeed, this is the cause for the polynomial dependence on depth in our single-key PKFE construction. Below, we further elaborate on the Hemenway et al. hybrid strategy and how it is imported into our setting.

Decomposable Garbling and Pebbling. The work of Hemenway et al. [HJO$^+$15] provided a useful abstraction for proving the security of Yao's garbled circuits via a sequence of hybrid games. The goal is to transition from a "real" garbled circuit, where each garbled gate is in "RealGate" mode consisting of four ciphertexts encrypting the two labels k_c^0, k_c^1 of the output wire c under the labels of the input wires, to a "simulated" garbled circuit where each garbled gate is in SimGate mode consisting of four ciphertexts that all encrypt the same dummy label k_c^0. As an intermediate step, we can also create a garbled gate in CompDepSimGate mode consisting of four ciphertexts encrypting the same label $k_c^{v(c)}$ where $v(c)$ is the value going over wire c during the computation $C(x)$ and therefore depends on the actual computation.

The transition from a real garbled circuit to a simulated garbled circuit proceeds via a sequence of hybrids where in each subsequent hybrid we can change one gate at a time from RealGate to CompDepSimGate (and vice versa) if all of its predecessors are in CompDepSimGate mode or it is an input gate, or change a gate from CompDepSimGate mode to SimGate mode (and vice versa) if all of its successors are in CompDepSimGate or SimGate modes. The goal of Hemenway et al. was to give a strategy using the least number of gates in CompDepSimGate mode as possible.[3] They abstracted this problem as a pebbling game and show that for circuits of depth d there exists a sequence of $2^{O(d)}$ hybrids with at most $O(d)$ gates in CompDepSimGate mode in any single hybrid.

In our case, we can give a decomposable circuit for each such hybrid game consisting of gates in RealGate, SimGate, CompDepSimGate modes. In particular, the decomposable circuit takes as input a gate index and outputs the garbled gate in the correct mode. We only need to remember which gate is in which mode, and for all gates in CompDepSimGate mode we need to remember the bit $v(c)$ going over the wire c during the computation $C(x)$. It turns out that the configuration of which mode each gate is in can be represented succinctly, and therefore the number of bits we need to remember is roughly proportional to the number of gates in CompDepSimGate mode in any given hybrid. Therefore, for circuits of depth d, the decomposable circuit is of size $O(d)$ and the number of hybrid steps is $2^{O(d)}$.

To ensure that the obfuscations of decomposable circuits corresponding to neighboring hybrids are indistinguishable we also need to rely on standard puncturing techniques. In particular, the gates are garbled using a punctured PRF and we show that in any transition between neighboring hybrids we can even give the adversary the PRF key punctured only on the surrounding of the gate whose mode is changed.

From SKFE to PKE: the basic idea. We end our technical exposition by explaining the basic idea behind the construction of public-key encryption (PKE) from SKFE. The construction is rather natural. Using subexponentially-secure SKFE and the second part of Proposition 1.1, we can obtain a $\mathrm{poly}(\lambda)$-secure SXIO with a subconstant compression factor $\gamma = o(1)$; concretely, it can be for example $O(1/\log\log\lambda)$. We can now think about this obfuscator as a plain (efficient) indistinguishability obfuscator for circuits with input length at most $\log\lambda \cdot \log\log\lambda$.

Then, we take a construction of public-key encryption from IO and one-way functions where the input-size of obfuscated circuits can be scaled down at the expense of strengthening the one-way functions. For instance, following the basic *witness encryption paradigm* in [GGSW13], the public key can be a pseudorandom string $\mathsf{PK} = \mathsf{PRG}(s)$ for a $2^{n/\log\log n}$-secure length-doubling pseudorandom generator with

[3] Their aim was proving adaptive security, which is completely orthogonal to our aim. However, for entirely different reasons, the above goal is useful in both their work and ours.

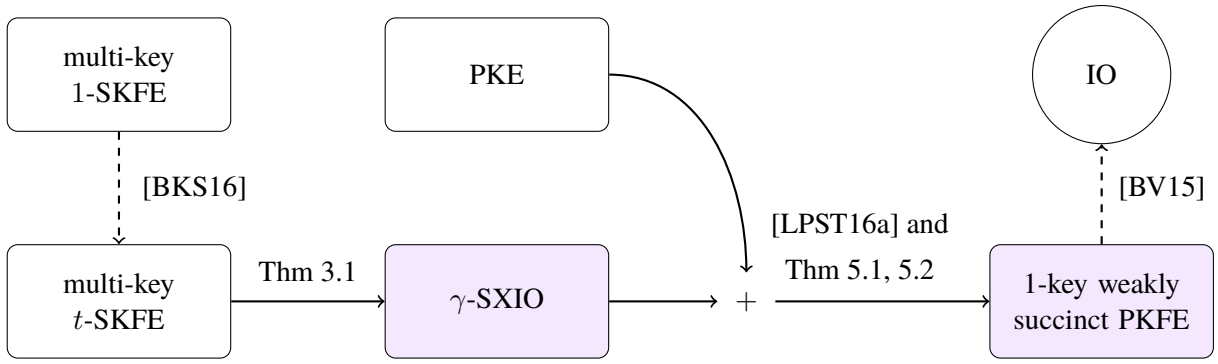

Figure 1: An illustration of our result on IO. Dashed lines denote known results. White backgrounds denote our ingredients or goal. Primitives in rounded rectangles are subexponentially-secure. t-SKFE denotes t-input SKFE. γ-SXIO denotes SXIO with compression factor γ, which is an arbitrary constant less than 1.

seed length $n = \log \lambda \cdot \log \log \lambda$. Here the obfuscator is only invoked for a circuit with inputs in $\{0,1\}^n$. An encryption of m is simply an obfuscation of a circuit that has PK hardwired, and releases m only given a seed s such that $\mathsf{PK} = \mathsf{PRG}(s)$. Security follows essentially as in [GGSW13]. Note that in this construction, we cannot expect more than 2^n security, which is quasi-polynomial in the security parameter λ.

How does the construction circumvent the Asharov-Segev barrier? As noted earlier, Asharov and Segev [AS15] show that even exponentially secure SKFE cannot lead to public-key encryption through a fully black-box construction (see their paper for details about the exact model). The reason that our construction does not fall under their criteria lies in the transformation from SKFE to SXIO with subconstant compression, and concretely in the Brakerski-Komargodski-Segev [BKS16] transformation from SKFE to t-input SKFE that makes non-black-box use in the algorithms of the underlying SKFE scheme.

Organization

In Section 2, we provide preliminaries and basic definitions used throughout the paper. In Section 4, we review Yao's garbled circuits and introduce a notion of decomposable garbling. In Section 3, we introduce the definition of SXIO and present our construction based on SKFE schemes. In Section 5, we present our construction of IO from PKE and SXIO. In Section 6, we present a polynomially-secure PKE scheme from SKFE schemes.

2 Preliminaries

We review basic concepts and definitions used throughout the paper.

2.1 Standard Computational Concepts

We rely on the standard notions of Turing machines and Boolean circuits.

- We say that a (uniform) Turing machine is PPT if it is probabilistic and runs in polynomial time.
- A polynomial-size (or just polysize) circuit family $\mathcal{C}$ is a sequence of circuits $\mathcal{C} = \{C_\lambda\}_{\lambda \in \mathbb{N}}$, such that each circuit C_λ is of polynomial size $\lambda^{O(1)}$ and has $\lambda^{O(1)}$ input and output bits.

- We follow the standard habit of modeling any efficient adversary strategy as a family of polynomial-size circuits. For an adversary $\mathcal{A}$ corresponding to a family of polysize circuits $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$, we often omit the subscript λ, when it is clear from the context.
- We say that a function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for all constants $c > 0$, there exists $N \in \mathbb{N}$ such that for all $n > N$, $f(n) < n^{-c}$.
- If $\mathcal{X}^{(b)} = \{X_\lambda^{(b)}\}_{\lambda\in\mathbb{N}}$ for $b \in \{0, 1\}$ are two ensembles of random variables indexed by $\lambda \in \mathbb{N}$, we say that $\mathcal{X}^{(0)}$ and $\mathcal{X}^{(1)}$ are computationally indistinguishable if for all polysize distinguishers $\mathcal{D}$, there exists a negligible function ν such that for all λ,

$$\Delta = \left|\Pr[\mathcal{D}(X_\lambda^{(0)}) = 1] - \Pr[\mathcal{D}(X_\lambda^{(1)}) = 1]\right| \leq \nu(\lambda).$$

- We write $\mathcal{X}^{(0)} \approx_\delta \mathcal{X}^{(1)}$ to denote that the advantage Δ is bounded by δ.

2.2 Functional Encryption

In this section, we define the different notions of functional encryption (FE) considered in this work.

Definition 2.1 (Multi-input secret-key functional encryption)**.** *Let $t(\lambda)$ be a function, $\overline{\mathcal{M}} = \{\overline{\mathcal{M}}_\lambda = \mathcal{M}_\lambda^{(1)} \times \cdots \times \mathcal{M}_\lambda^{(t(\lambda))}\}_{\lambda\in\mathbb{N}}$ be a product message domain, $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ a range, and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ a class of t-input functions $f : \overline{\mathcal{M}}_\lambda \to \mathcal{Y}_\lambda$. A* t-input secret-key functional encryption *(t-SKFE) scheme for $\mathcal{M}, \mathcal{Y}, \mathcal{F}$ is a tuple of algorithms $\mathsf{SKFE}_t = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ where:*

- $\mathsf{Setup}(1^\lambda)$ *takes as input the security parameter and outputs a master secret key* MSK.
- $\mathsf{KeyGen}(\mathsf{MSK}, f)$ *takes as input the master secret* MSK *and a function $f \in \mathcal{F}$. It outputs a secret key SK_f for f.*
- $\mathsf{Enc}(\mathsf{MSK}, m, i)$ *takes as input the master secret key* MSK*, a message $m \in \mathcal{M}_\lambda^{(i)}$, and an index $i \in [t(\lambda)]$, and outputs a ciphertext CT_i.*
- $\mathsf{Dec}(\mathsf{SK}_f, \mathsf{CT}_1, \ldots, \mathsf{CT}_t)$ *takes as input the secret key SK_f for a function $f \in \mathcal{F}$ and ciphertexts $\mathsf{CT}_1, \ldots, \mathsf{CT}_t$, and outputs some $y \in \mathcal{Y}$, or $\perp$.*

We also require the following property:

Correctness: *For all tuples $\vec{m} = (m_1, \ldots, m_t) \in \overline{\mathcal{M}}_\lambda$ and any function $f \in \mathcal{F}_\lambda$, we have that*

$$\Pr\left[\mathsf{Dec}(\mathsf{SK}_f, \mathsf{CT}_1, \ldots, \mathsf{CT}_t) = f(\vec{m}) : \begin{array}{l} \mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda), \\ \mathsf{SK}_f \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f), \\ \forall i\ \mathsf{CT}_i \leftarrow \mathsf{Enc}(\mathsf{MSK}, m, i) \end{array}\right] = 1$$

Definition 2.2 (Selectively-secure multi-key t-SKFE)**.** *We say that a tuple of algorithms $\mathsf{SKFE}_t = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a* selectively-secure *t-input secret-key functional encryption scheme for $\overline{\mathcal{M}}, \mathcal{Y}, \mathcal{F}$, if it satisfies the following requirement, formalized by the experiment $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{SKFE}_t}(1^\lambda, b)$ between an adversary $\mathcal{A}$ and a challenger:*

1. *The adversary submits challenge message tuples $\left\{(m_{i,1}^0, m_{i,1}^1, i)\right\}_{i\in[t]}, \ldots, \left\{(m_{i,q}^0, m_{i,q}^1, i)\right\}_{i\in[t]}$ for all $i \in [t]$ to the challenger where q is an arbitrary polynomial in λ.*

2. *The challenger runs* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$

3. *The challenger generates ciphertexts* $\mathsf{CT}_{i,j} \leftarrow \mathsf{Enc}(\mathsf{MSK}, m^b_{i,j}, i)$ *for all* $i \in [t]$ *and* $j \in [q]$*, and gives* $\{\mathsf{CT}_{i,j}\}_{i\in[t],j\in[q]}$ *to* $\mathcal{A}$*.*

4. $\mathcal{A}$ *is allowed to make* q *function queries, where it sends a function* $f_j \in \mathcal{F}$ *to the challenger for* $j \in [q]$ *and* q *is an arbitrary polynomial in* λ*. The challenger responds with* $\mathsf{SK}_{f_j} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f_j)$*.*

5. $\mathcal{A}$ *outputs a guess* b' *for* b*.*

6. *The output of the experiment is* b' *if the adversary's queries are valid:*

$$f_j(m^0_{1,j_1}, \ldots, m^0_{t,j_t}) = f_j(m^1_{1,j_1}, \ldots, m^1_{t,j_t}) \text{ for all } j_1, \ldots, j_t, j \in [q] \ .$$

Otherwise, the output of the experiment is set to be $\perp$*.*

We say that the functional encryption scheme is selectively-secure if, for all polysize adversaries $\mathcal{A}$*, there exists a negligible function* $\mu(\lambda)$*, such that*

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

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

We recall the following theorem by Brakerski, Komargodski, and Segev, which states that one can build selectively-secure t-SKFE from any selectively-secure 1-SKFE. The transformation induces a significant blowup and security loss in the number of inputs t. This loss is polynomial as long as t is constant, but in general grows doubly-exponentially in t.

Theorem 2.1 ([BKS16])**.**

1. *For* $t = O(1)$*, if there exists* δ*-selectively-secure single-input secret-key functional encryption, then there exists* δ*-selectively-secure* t*-input secret-key functional encryption.*

2. *There exists a constant* $\varepsilon < 1$*, such that for* $t(\lambda) = \varepsilon \cdot \log\log(\lambda)$*,* $\tilde{\lambda} = 2^{(\log\lambda)^\varepsilon}$*,* $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\varepsilon}$*, if there exists* δ*-selectively-secure single-input secret-key functional encryption, then there exists polynomially-secure selectively-secure* t*-input secret-key functional encryption for functions of size at most* $2^{O((\log\lambda)^\varepsilon)}$*. (Here* $\tilde{\lambda}$ *is the* 1*-SKFE security parameter and* λ *is the* t*-SKFE security parameter.)*

Remark 2.1 (Dependence on circuit size in [BKS16]). The [BKS16] transformation incurs a $(s \cdot \tilde{\lambda})^{2^{O(t)}}$ blowup in parameters, where s is the size of maximal circuit size of supported functions, and $\tilde{\lambda}$ is the security parameter used in the underlying single-input SKFE. In the main setting of parameters considered there, $t = O(1)$, the security parameter λ of the t-SKFE scheme can be identified with $\tilde{\lambda}$ and s can be any polynomial in this security parameter. (Accordingly, the dependence on s is implicit there, and the blowup they address is $\lambda^{2^{O(t)}}$.)

For the second part of the theorem, to avoid superpolynomial blowup in λ, the security parameter $\tilde{\lambda}$ for the underlying SKFE and the maximal circuit size s should be set to $2^{O((\log\lambda)^\varepsilon)}$.

Definition 2.3 (Public-key functional encryption)**.** *Let* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *be a message domain,* $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ *a range, and* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *a class of functions* $f : \mathcal{M} \to \mathcal{Y}$*. A* public-key functional encryption *(PKFE) scheme for* $\mathcal{M}, \mathcal{Y}, \mathcal{F}$ *is a tuple of algorithms* $\mathsf{PKFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *where:*

- $\mathsf{Setup}(1^\lambda)$ *takes as input the security parameter and outputs a master secret key* MSK *and master public key* MPK.
- $\mathsf{KeyGen}(\mathsf{MSK}, f)$ *takes as input the master secret* MSK *and a function* $f \in \mathcal{F}$. *It outputs a secret key* SK_f *for* f.
- $\mathsf{Enc}(\mathsf{MPK}, m)$ *takes as input the master public key* MPK *and a message* $m \in \mathcal{M}$, *and outputs a ciphertext* c.
- $\mathsf{Dec}(\mathsf{SK}_f, c)$ *takes as input the secret key* SK_f *for a function* $f \in \mathcal{F}$ *and a ciphertext* c, *and outputs some* $y \in \mathcal{Y}$, *or* $\perp$.

We also require the following property:

Correctness: *For any message* $m \in \mathcal{M}$ *and function* $f \in \mathcal{F}$, *we have that*

$$\Pr\left[\mathsf{Dec}(\mathsf{SK}_f, c) = f(m) : \begin{array}{l}(\mathsf{MSK}, \mathsf{MPK}) \leftarrow \mathsf{Setup}(1^\lambda), \\ \mathsf{SK}_f \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f), \\ c \leftarrow \mathsf{Enc}(\mathsf{MPK}, m)\end{array}\right] = 1$$

Definition 2.4 (Selectively-secure single-key PKFE)**.** *We say that a tuple of algorithm* $\mathsf{PKFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a* selectively-secure single-key *public-key functional encryption scheme for* $\mathcal{M}, \mathcal{Y}, \mathcal{F}$, *if it satisfies the following requirement, formalized by the experiment* $\mathsf{Expt}^{\mathsf{PKFE}}_{\mathcal{A}}(1^\lambda, b)$ *between an adversary* $\mathcal{A}$ *and a challenger:*

1. *The adversary submits the challenge message pair* $m^*_0, m^*_1 \in \mathcal{M}$ *and a function* f *to the challenger.*
2. *The challenger runs* $(\mathsf{MSK}, \mathsf{MPK}) \leftarrow \mathsf{Setup}(1^\lambda)$, *generates ciphertext* $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, m^*_b)$ *and a secret key* $\mathsf{SK}_f \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f)$. *The challenger gives* $(\mathsf{MPK}, \mathsf{CT}^*, sk_f)$ *to* $\mathcal{A}$.
3. $\mathcal{A}$ *outputs a guess* b' *for* b.
4. *The output of the experiment is* b' *if* $f(m^*_0) = f(m^*_1)$. *Otherwise the output is* $\perp$.

We say that the public-key functional encryption scheme is selectively-secure if, for all PPT adversaries $\mathcal{A}$, *there exists a negligible function* $\mu(\lambda)$, *such that*

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

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

We now further define two notions of succinctness for functional encryption schemes as above.

Definition 2.5 (Succinct functional encryption)**.** *For a class of functions* $\mathcal{F} = \{\mathcal{F}_\lambda\}$ *over message domain* $\mathcal{M} = \{\mathcal{M}_\lambda\}$, *we let:*

- $n(\lambda)$ *be the input length of the functions in* $\mathcal{F}$,
- $s(\lambda) = \max_{f \in \mathcal{F}_\lambda} |f|$ *be a bound on the circuit size of functions in* $\mathcal{F}_\lambda$,
- $d(\lambda) = \max_{f \in \mathcal{F}_\lambda} \mathsf{depth}(f)$ *a bound on the depth, and*

A functional encryption scheme is

- weakly succinct [BV15] *if the size of the encryption circuit is bounded by* $s^\gamma \cdot \mathrm{poly}(n, \lambda, d)$*, where* poly *is a fixed polynomial, and* $\gamma < 1$ *is a constant. We call* γ *the compression factor.*

The following result from [BV15, Section 3.2] states that one can construct an indistinguishability obfuscator from any single-key weakly succinct public-key functional encryption scheme.

Theorem 2.2 ([BV15])**.** *If there exists a subexponentially secure single-key weakly succinct public-key functional encryption scheme, then there exists an indistinguishability obfuscator.*

2.3 Indistinguishability Obfuscation

We recall the notion of indistinguishability obfuscation (IO) [BGI+01, BGI+12].

Definition 2.6 (Indistinguishability obfuscator (IO))**.** *A PPT machine* $\mathsf{i}\mathcal{O}$ *is an* indistinguishability obfuscator *for a circuit class* $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ *if the following conditions are satisfied:*

- ***Functionality:*** *for all security parameters* $\lambda \in \mathbb{N}$*, for all* $C \in \mathcal{C}_\lambda$*, for all inputs* x*, we have that*

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

- ***Indistinguishability:*** *for any polysize distinguisher* $\mathcal{D}$*, there exists a negligible function* $\mu(\cdot)$ *such that the following holds: for all security parameters* $\lambda \in \mathbb{N}$*, for all pairs of circuits* $C_0, C_1 \in \mathcal{C}_\lambda$ *of the same size and such that* $C_0(x) = C_1(x)$ *for all inputs* x*, then*

$$\left|\Pr\left[\mathcal{D}(\mathsf{i}\mathcal{O}(C_0)) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{i}\mathcal{O}(C_1)) = 1\right]\right| \leq \mu(\lambda) \ .$$

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

2.4 Puncturable Pseudorandom Functions

Puncturable PRFs, defined by Sahai and Waters [SW14], are PRFs with a key-puncturing procedure that produces keys that allow evaluation of the PRF on all inputs, except for a designated polynomial-size set.

Definition 2.7 (Puncturable pseudorandom function)**.** *For sets* $\mathcal{D}, \mathcal{R}$*, a* puncturable pseudorandom function *(PPRF) consists of a tuple of algorithms* $\mathcal{PPRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Ev}, \mathsf{PRF.Punc})$ *that satisfy the following two conditions.*

Functionality preserving under puncturing: *For all polynomial size set* $S \subseteq \mathcal{D}$ *and for all* $x \in \mathcal{D} \setminus S$*, it holds that*

$$\Pr[\mathsf{PRF.Ev}_K(x) = \mathsf{PRF.Ev}_{K\{S\}}(x) : K \leftarrow \mathsf{PRF.Gen}(1^\lambda), K\{S\} \leftarrow \mathsf{PRF.Punc}(K, S)] = 1.$$

Pseudorandom at punctured points: *For all polynomial size set $S \subseteq \mathcal{D}$ with $S = \{x_1, \ldots, x_{k(\lambda)}\}$ and any polysize distinguisher $\mathcal{A}$, there exists a negligible function μ, such that:*

$$|\Pr[\mathcal{A}(\mathsf{PRF.Ev}_{K\{S\}}, \{\mathsf{PRF.Ev}_K(x_i)\}_{i\in[k]}) = 1] - \Pr[\mathcal{A}(\mathsf{PRF.Ev}_{K\{S\}}, U^k) = 1]| \leq \mu(\lambda)$$

where $K \leftarrow \mathsf{PRF.Gen}(1^\lambda)$, $K\{S\} \leftarrow \mathsf{PRF.Punc}(K, S)$ and U denotes the uniform distribution over $\mathcal{R}$. We further say that $\mathcal{PPRF}$ is δ-secure, for some concrete negligible function $\delta(\cdot)$, if for all polysize distinguishers the above indistinguishability gap $\mu(\lambda)$ is smaller than $\delta(\lambda)^{\Omega(1)}$.

The GGM tree-based construction of PRFs [GGM84] from one-way functions are easily seen to yield puncturable PRFs where the size of the punctured key grows polynomially with the size of the set S being punctured, as recently observed by [BW13, BGI14, KPTZ13]. Thus, we have:

Theorem 2.3 ([GGM84, BW13, BGI14, KPTZ13])**.** *If one-way functions exist, then for all efficiently computable functions $n(\lambda)$ and $m(\lambda)$, there exists a puncturable pseudorandom function that maps $n(\lambda)$ bits to $m(\lambda)$ bits (i.e., $\mathcal{D} = \{0,1\}^{n(\lambda)}$ and $\mathcal{R} = \{0,1\}^{m(\lambda)}$).*

2.5 Public-Key Encryption

We recall the notion of plain public-key encryption (PKE).

Definition 2.8 (Plain public-key encryption)**.** *Let $\mathcal{M}$ be some message space. A* public-key encryption *(PKE) scheme for $\mathcal{M}$ is a tuple of algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ where:*

- $\mathsf{KeyGen}(1^\lambda)$ *takes as input the security parameter and outputs a public key* PK *and a secret key* SK.
- $\mathsf{Enc}(\mathsf{PK}, m)$ *takes as input the public key* PK *and a message $m \in \mathcal{M}$ and outputs a ciphertext* CT.
- $\mathsf{Dec}(\mathsf{SK}, \mathsf{CT})$ *takes as input the secret key* SK *and a ciphertext* CT*, and outputs some $m \in \mathcal{M}$, or $\perp$.*

We also require the following property:

Correctness: *For any message $m \in \mathcal{M}$, we have that*

$$\Pr\left[\mathsf{Dec}(\mathsf{SK}, c) = m : \begin{array}{l}(\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{KeyGen}(1^\lambda), \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{PK}, m)\end{array}\right] = 1$$

We also recall the standard notion of security.

Definition 2.9 (Secure public-key encryption)**.** *A tuple of algorithms $PKE = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a* secure *PKE for $\mathcal{M}$ if it satisfies the following requirement, formalized by the experiment $\mathsf{Expt}^{\mathsf{PKE}}_{\mathcal{A}}(1^\lambda, b)$ between an adversary $\mathcal{A}$ and a challenger:*

1. *The challenger runs $(\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and gives PK to $\mathcal{A}$.*
2. *At some point, $\mathcal{A}$ sends two messages m_0^*, m_1^* as the challenge messages to the challenger.*
3. *The challenger generates ciphertext $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{PK}, m_b^*)$ and sends CT^* to $\mathcal{A}$.*
4. *$\mathcal{A}$ outputs a guess b' for b. The experiment outputs 1 if $b' = b$, 0 otherwise.*

We say the PKE scheme is secure if, for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\lambda)$, it holds:

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

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

2.6 Succinct Identity-Based Encryption

We define identity-based encryption (IBE) [Sha84] with certain succinctness properties.

Definition 2.10 (Succinct IBE with γ-compression)**.** *Let $\mathcal{M}$ be some message space and $\mathcal{ID}$ be an identity space. A* succint identity-based encryption scheme with γ-compression *for $\mathcal{M}, \mathcal{ID}$ is a tuple of algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *where:*

- $\mathsf{Setup}(1^\lambda)$ *is takes as input the security parameter and outputs a master secret key* MSK *and a master public key* MPK.
- $\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$ *takes as input the master secret* MSK *and an identity* $\mathsf{id} \in \mathcal{ID}$*. It outputs a secret key* $\mathsf{SK}_{\mathsf{id}}$ *for* id.
- $\mathsf{Enc}(\mathsf{MPK}, \mathsf{id}, m)$ *takes as input the public-parameter* MPK*, an identity* $\mathsf{id} \in \mathcal{ID}$*, and a message* $m \in \mathcal{M}$*, and outputs a ciphertext* c.
- $\mathsf{Dec}(\mathsf{SK}_{\mathsf{id}}, c)$ *takes as input the secret key* $\mathsf{SK}_{\mathsf{id}}$ *for an identity* $\mathsf{id} \in \mathcal{ID}$ *and a ciphertext* c*, and outputs some* $m \in \mathcal{M}$*, or* $\perp$.

We require the following properties:

Correctness: *For any message $m \in \mathcal{M}$ and identity $\mathsf{id} \in \mathcal{ID}$, we have that*

$$\Pr\left[\mathsf{Dec}(\mathsf{SK}_{\mathsf{id}}, c) = m : \begin{array}{l}(\mathsf{MSK}, \mathsf{MPK}) \leftarrow \mathsf{Setup}(1^\lambda), \\ \mathsf{SK}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id}), \\ c \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{id}, m)\end{array}\right] = 1$$

Succinctness: *For any security parameter $\lambda \in \mathbb{N}$, identity space $\mathcal{ID}$, the size of the encryption circuit* Enc*, for messages of size ℓ, is at most $|\mathcal{ID}|^\gamma \cdot \mathrm{poly}(\lambda, \ell)$.*

In this work, we shall consider the following selective-security.

Definition 2.11 (Selectively-secure IBE)**.** *A tuple of algorithms* $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a* selectively-secure *IBE scheme for $\mathcal{M}, \mathcal{ID}$ if it satisfies the following requirement, formalized by the experiment $\mathsf{Expt}^{\mathsf{IBE}}_{\mathcal{A}}(1^\lambda, b)$ between an adversary $\mathcal{A}$ and a challenger:*

1. *The adversary submits the challenge identity $\mathsf{id}^* \in \mathcal{ID}$ and the challenge messages (m_0^*, m_1^*) to the challenger.*
2. *The challenger runs $(\mathsf{MSK}, \mathsf{MPK}) \leftarrow \mathsf{Setup}(1^\lambda)$, generates ciphertext $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, m_b^*)$ and gives $(\mathsf{MPK}, \mathsf{CT}^*)$ to $\mathcal{A}$.*
3. *The adversary is allowed to query (polynomially many) identities $\mathsf{id} \in \mathcal{ID}$ such that $\mathsf{id} \neq \mathsf{id}^*$. The challenger gives $\mathsf{SK}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{MSK}, \mathsf{id})$ to the adversary.*
4. *$\mathcal{A}$ outputs a guess b' for b. The experiment outputs 1 if $b' = b$, 0 otherwise.*

We say the IBE scheme is selectively-secure if, for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\lambda)$, it holds

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

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

3 Strong Exponentially-Efficient Indistinguishability Obfuscation

Lin, Pass, Seth, and Telang [LPST16a] propose a variant of IO that has a weak (yet non-trivial) efficiency, which they call exponentially-efficient IO (XIO). All that this notion requires in terms of efficiency is that the size of an obfuscated circuit is sublinear in the size of the corresponding truth table. They also refer to a stronger notion that requires that also the time to obfuscate a given circuit is sublinear in the size of the truth table. This notion, which we call *strong* exponentially-efficient IO (SXIO), serves as one of the main abstractions in our work.

Definition 3.1 (Strong exponentially-efficient indistinguishability obfuscation (SXIO) [LPST16a])**.** *For a constant $\gamma < 1$, a machine $\mathsf{sxi}\mathcal{O}$ is a γ-compressing* strong exponentially-efficient indistinguishability obfuscator *(SXIO) for a circuit class $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ if it satisfies the functionality and indistinguishability in Definition 2.6 and the following efficiency requirements:*

Non-trivial time efficiency: *for any security parameter $\lambda \in \mathbb{N}$ and circuit $C \in \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ with input length n, the running time of $\mathsf{sxi}\mathcal{O}$ on input $(1^\lambda, C)$ is at most $2^{n\gamma} \cdot \mathrm{poly}(\lambda, |C|)$.*

3.1 SXIO from Single-Input SKFE

In this section, we show that we can construct SXIO from any selectively-secure t-input secret-key functional encryption scheme. We recall that such a t-SKFE scheme can be constructed from any selectively-secure 1-SKFE scheme, as stated in Theorem 2.1.

Theorem 3.1. *For any function $t(\lambda)$, if there exists δ-selectively-secure t-SKFE, then there exists $\frac{1}{t+1}$-compressing δ-secure SXIO.*

The idea of the construction of SXIO from SKFE is explained in the introduction.
We immediately obtain the following corollary from Theorem 2.1 and 3.1.

Corollary 3.1.

1. *If there exists δ-selectively-secure single-input secret-key functional encryption, then there exists γ-compressing δ-secure SXIO where $\gamma < 1$ is an arbitrary constant.*
2. *Let $\varepsilon < 1$ be a constant and $\tilde{\lambda} = 2^{(\log\lambda)^\varepsilon}$. If there exists $2^{-\tilde{\lambda}^{\Omega(1)}}$-selectively-secure single-input secret-key functional encryption, then there exists polynomially-secure SXIO with compression factor $\gamma(\lambda) = O(1/\log\log\lambda)$ for circuits of size at most $2^{O((\log\lambda)^\varepsilon)}$. (Here $\tilde{\lambda}$ is the 1-SKFE security parameter and λ is the SXIO security parameter.)*

3.2 The Construction and Proof of SXIO

We start by describing the SXIO construction and next argue its security. In what follows, given a circuit C, we identify its input space with $[N] = \{1, \ldots, N\}$ (so in particular, $N = 2^n$ if C takes n-bit strings as input). Let $\mathsf{SKFE}_t = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a selectively-secure t-input secret-key functional encryption scheme.

Construction. We construct an SXIO scheme $\mathsf{sxi}\mathcal{O}$ as follows.

$\mathsf{sxi}\mathcal{O}(1^\lambda, C)$**:** For every $j \in [N^{1/(t+1)}]$:

- let U_j be the t-input universal circuit that given $j_1, \ldots, j_{t-1} \in [N^{1/(t+1)}]$ and a t-input circuit D, returns $D(j_1, \ldots, j_{t-1}, j)$.
- let C_j be the t-input circuit that given $j_1, \ldots, j_t \in [N^{1/(t+1)}]$ returns $C(j_1, \ldots, j_t, j)$.

1. Generate $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$.
2. Generate $\mathsf{CT}_{t,j} \leftarrow \mathsf{Enc}(\mathsf{MSK}, C_j, t)$ for $j \in [N^{1/(t+1)}]$.
3. Generate $\mathsf{CT}_{i,j} \leftarrow \mathsf{Enc}(\mathsf{MSK}, j, i)$ for $i \in [t-1]$ and $j \in [N^{1/(t+1)}]$.
4. Generate $\mathsf{SK}_{U_j} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, U_j)$ for $j \in [N^{1/(t+1)}]$
5. $\mathsf{sxi}\mathcal{O}(C) = (\{\mathsf{CT}_{i,j}\}_{i\in[t],j\in[N^{1/(t+1)}]}, \{\mathsf{SK}_{U_j}\}_{j\in[N^{1/(t+1)}]})$

$\mathsf{Eval}(\mathsf{sxi}\mathcal{O}, x)$**:** To evaluate the obfuscated circuit, convert $x \in [N]$ into $(j_1, \ldots, j_t, j_{t+1}) \in [N^{1/(t+1)}]^{(t+1)}$ and output $\mathsf{Dec}(\mathsf{SK}_{U_{j_{t+1}}}, \mathsf{CT}_{1,j_1}, \ldots, \mathsf{CT}_{t,j_t})$.

Proof of Theorem 3.1. We first note that $\mathsf{sxi}\mathcal{O}$ indeed satisfies the non-trivial time efficiency requirement. The obfuscated circuit consists of $t \cdot N^{1/(t+1)}$ ciphertexts, each computable in time $\mathrm{poly}(|C|, \lambda)$, and $N^{1/(t+1)}$ secret-keys of SKFE_t, each computable in time $\mathrm{poly}(|C|, \lambda)$. Thus, for $t \leq \lambda$, the overall running-time of the obfuscator is bounded by $N^{1/(t+1)} \cdot \mathrm{poly}(|C|, \lambda)$, as required.

Security. We show that if there exists a distinguisher $\mathcal{D}$ against $\mathsf{sxi}\mathcal{O}$, then there exists an adversary $\mathcal{A}$ that breaks the security of the underlying scheme SKFE_t (it even beaks a weaker security notion than the one from Definition 2.2, where function queries are also fixed in advance).

The reduction is straightforward. If $\mathcal{D}$ distinguishes $\mathsf{sxi}\mathcal{O}$ obfuscations of circuits C, C' (of same size and functionality), then $\mathcal{A}$ can invoke $\mathcal{D}$ to distinguish

$$\left\{(j_1, \ldots, j_{t-1}, C_{j_t}) : j_i \in [N^{1/(t+1)}]\right\} \text{ and } \left\{(j_1, \ldots, j_{t-1}, C'_{j_t}) : j_i \in [N^{1/(t+1)}]\right\},$$

given secret keys for all functions $\left\{U_{j_{t+1}} : j_{t+1} \in [N^{1/(t+1)}]\right\}$. Since the two circuits are functionally equivalent, all these queries are valid queries, since $U_{j_{t+1}}(j_1, \ldots, j_{t-1}, C_{j_t}) = C(j_1, \ldots, j_t, j_{t+1}) = C(j_1, \ldots, j_t, j_{t+1}) = U_{j_{t+1}}(j_1, \ldots, j_{t-1}, C'_{j_t})$.

This completes the proof of Theorem 3.1. □

Remark 3.1 (SXIO from succinct single-key SKFE). To get t-input SKFE as required above from 1-input SKFE, via the [BKS16] transformation, the original SKFE indeed has to support an unbounded polynomial number of functional keys. We note that a similar SXIO construction is possible from a 1-input SKFE that supports a functional key for a single function f, but is *succinct* in the sense that the encryption circuit does not grow with the complexity of f.

In more detail, assume a (1-input) single-key SKFE with succinctness as above, where the time to derive a key for a function f is bounded by $|f|^c \cdot \mathrm{poly}(\lambda)$ for some constant $c \geq 1$. The SXIO will consist of a single key for the function f that given as input C_j, as defined above, returns $C_j(1), \ldots, C_j(N^{\frac{1}{c+1}})$, and encryptions of $C_1, \ldots, C_{N^{c/c+1}}$. Accordingly we still get SXIO with compression factor $\gamma = 1 - \frac{1}{c+1}$. This does not lead to arbitrary constant compression (in contrast with the theorem above), since $\frac{1}{2} \leq \gamma < 1$. Yet, it already suffices to obtain IO, when combined with LWE (as in Corollary 1.1).

4 Yao's Garbled Circuits are Decomposable

In this section, we define the notion of decomposable garbled circuits and prove that the classical Yao's garbled circuit construction satisfies our definition of decomposability (in some parameter regime). We use a decomposable garbling scheme as a building block to construct a PKFE scheme in Section 5.2.

4.1 Decomposable Garbling

Circuit garbling schemes [Yao82, BHR12] typically consist of algorithms $(\mathsf{Gar.CirEn}, \mathsf{Gar.InpEn}, \mathsf{Gar.De})$. $\mathsf{Gar.CirEn}(C, K)$ is a circuit garbling algorithm that given a circuit C and secret key K, produces a garbled circuit $\widehat{C}$. $\mathsf{Gar.InpEn}(x, K)$ is an input garbling algorithm that takes an input x and the same secret key K, and produces a garbled input $\widehat{x}$. $\mathsf{Gar.De}(\widehat{C}, \widehat{x})$ is a decoder that given the garbled circuit and input decodes the result y.

In this work, we shall particularly be interested in garbling *decomposable circuits*. A decomposable circuit C can be represented by a smaller circuit C_{de} that can generate each of the gates in the circuit C (along with pointers to their neighbours). When garbling such circuits, we shall require that the garbling process will also be decomposable and will admit certain *decomposable security* properties. We next formally define the notion of decomposable circuits and decomposable garbling schemes.

Definition 4.1 (Decomposable Circuit)**.** *Let $C : \{0,1\}^n \to \{0,1\}$ be a boolean circuit with L binary gates and W wires. Each gate $g \in [L]$ has an associated tuple (f, w_a, w_b, w_c) where $f : \{0,1\}^2 \to \{0,1\}$ is the binary function computed by the gate, $w_a, w_b \in [W]$ are the incoming wires, and $w_c \in [W]$ is the outgoing wire. A wire w_c can be the outgoing wire of at most a single gate, but can be used as an incoming wire to several different gates and therefore this models a circuit with fan-in 2 and unbounded fan-out. We define the* predecessor gates *of g to be the gates whose outgoing wires are w_a, w_b (at most 2 of them). We define the* successor gates *of g to be the gates that have w_c as an incoming wire. The gates are topologically ordered and labeled by $1, \ldots, L$ so that if j is a successor of i then $i < j$. A wire w is an* input wire *if it is not the outgoing wire of any gate. We assume that the wires $1, \ldots, n$ are the input wires. There is a unique* output wire *w which is not an incoming wire to any gate.*

We say that C is decomposable if there exists a smaller circuit C_{de}, called the decomposition circuit*, that given a gate label $g \in [L]$ as input, outputs the associated tuple $C_{\mathsf{de}}(g) = (f, w_a, w_b, w_c)$.*

Definition 4.2 (Decomposable Garbling)**.** *A decomposable garbling scheme consists of three deterministic polynomial-time algorithms $(\mathsf{Gar.CirEn}, \mathsf{Gar.InpEn}, \mathsf{Gar.De})$ that work as follows:*

- *$\widehat{b}_i \leftarrow \mathsf{Gar.InpEn}(i, b; K)$: takes as an input label $i \in [n]$, a bit $b \in \{0,1\}$, and secret key $K \in \{0,1\}^\lambda$, and outputs a garbled input bit $\widehat{b}_i$.*
- *$\widehat{G}_g \leftarrow \mathsf{Gar.CirEn}(C_{\mathsf{de}}, g; K)$: takes as input a decomposition circuit $C_{\mathsf{de}} : \{0,1\}^L \to \{0,1\}^*$, a gate label $g \in [L]$, and secret key $K \in \{0,1\}^\lambda$, and outputs a garbled gate $\widehat{G}_g$.*
- *$y \leftarrow \mathsf{Gar.De}(\widehat{C}, \widehat{b})$: takes as input garbled gates $\widehat{C} = \left\{\widehat{G}_g\right\}_{g \in [L]}$, and garbled input bits $\widehat{b} = \left\{\widehat{b}_i\right\}_{i \in [n]}$, and outputs $y \in \{0,1\}^m$.*

The scheme should satisfy the following requirements:

1. ***Correctness:*** *for every decomposable circuit C with decomposition circuit C_{de} and any input $b_1, \ldots, b_n \in \{0,1\}^n$, the decoding procedure $\mathsf{Gar.De}$ produces the correct output $y = C(b_1, \ldots, b_n)$.*

2. (σ, τ, δ)***-Decomposable Indistinguishability:*** *There are functions $\sigma(\Phi, s, \lambda), \tau(\Phi) \in \mathbb{N}$, $\delta(\lambda) \leq 1$ such that for any security parameter λ, any input $x \in \{0,1\}^n$, and any two circuits (C, C') that:*

   - *have the same topology Φ, and in particular the same size L and input-output lengths (n, m),*
   - *have decomposition circuits $(C_{\mathsf{de}}, C'_{\mathsf{de}})$ of the same size s*
   - *and agree on x: $C(x) = C'(x)$,*

   there exist hybrid circuits $\left\{ \mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)} \;\middle|\; t \in [\tau] \right\}$, each being of size at most σ, as well as (possibly inefficient) hybrid functions $\left\{ \mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in [\tau] \right\}$ with the following syntax:

   - $(K^{(t)}_{\mathsf{pun}}, g^{(t)}_{\mathsf{pun}}, i^{(t)}_{\mathsf{pun}}) \leftarrow \mathsf{Gar.HPunc}^{(t)}(K)$*, given a key $K \in \{0,1\}^\lambda$ and an index $t \in [\tau]$, outputs a* punctured key $K^{(t)}_{\mathsf{pun}}$*, a gate label $g^{(t)}_{\mathsf{pun}} \in [L]$, and an input label $i^{(t)}_{\mathsf{pun}} \in [n]$.*
   - $\widehat{G}_g \leftarrow \mathsf{Gar.HCirEn}^{(t)}(g; K)$*, given a gate label $g \in [L]$, and a (possibly punctured) key K, outputs a* fake garbled gate $\widehat{G}_g$*.*
   - $\widehat{b}_i \leftarrow \mathsf{Gar.HInpEn}^{(t)}(i, b; K)$*, given an input label $i \in [n]$, and a (possibly punctured) key K, outputs a* fake garbled input bit $\widehat{b}_i$*.*

   We require that the following properties hold:

   (a) ***The hybrids transition from C to C':*** *For any $K \in \{0,1\}^\lambda$, $g \in [L]$, $i \in [n]$, $b \in \{0,1\}$,*

$$\mathsf{Gar.CirEn}(C_{\mathsf{de}}, g; K) = \mathsf{Gar.HCirEn}^{(1)}(g; K) \;\; , \mathsf{Gar.InpEn}(i, b; K) = \mathsf{Gar.HInpEn}^{(1)}(i, b; K),$$
$$\mathsf{Gar.CirEn}(C'_{\mathsf{de}}, g; K) = \mathsf{Gar.HCirEn}^{(\tau)}(g; K) \;\; , \mathsf{Gar.InpEn}(i, b; K) = \mathsf{Gar.HInpEn}^{(\tau)}(i, b; K).$$

   (b) ***Punctured keys preserve functionality:*** *For any $K \in \{0,1\}^\lambda$, and $t \in [\tau - 1]$, and letting $(K^{(t)}_{\mathsf{pun}}, g^{(t)}_{\mathsf{pun}}, i^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HPunc}^{(t)}(K)$, it holds that:*

   - *For any $g \neq g^{(t)}_{\mathsf{pun}}$, we have*

$$\mathsf{Gar.HCirEn}^{(t)}(g; K) = \mathsf{Gar.HCirEn}^{(t)}(g; K^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HCirEn}^{(t+1)}(g, K).$$

   - *For any $i \neq i^{(t)}_{\mathsf{pun}}$ and $b \in \{0,1\}$, we have*

$$\mathsf{Gar.HInpEn}^{(t)}(i, b; K) = \mathsf{Gar.HInpEn}^{(t)}(i, b; K^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HInpEn}^{(t+1)}(i, b, K).$$

   (c) ***Indistinguishability on punctured inputs:*** *For any polysize distinguisher $\mathcal{D}$, security parameter $\lambda \in \mathbb{N}$, and circuits (C, C') as above,*

$$\left| \Pr\left[\mathcal{D}\left(\hat{g}^{(t)}_{\mathsf{pun}}, \hat{i}^{(t)}_{\mathsf{pun}}, \mathsf{Gar.HPunc}^{(t)}(K) \right) = 1 \right] - \Pr\left[\mathcal{D}\left(\hat{g}^{(t+1)}_{\mathsf{pun}}, \hat{i}^{(t+1)}_{\mathsf{pun}}, \mathsf{Gar.HPunc}^{(t)}(K) \right) = 1 \right] \right| \leq \delta(\lambda) \;\; ,$$

   where, for $t \geq 0$ we denote by $\hat{g}^{(t)}_{\mathsf{pun}}$ the value $\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K)$ and by $\hat{i}^{(t)}_{\mathsf{pun}}$ the value $\mathsf{Gar.HInpEn}^{(t)}(i^{(t)}_{\mathsf{pun}}, x_{i^{(t)}_{\mathsf{pun}}}; K)$, with x being the input on which the two circuits C and C' agree on. The probability is over $K \leftarrow \{0,1\}^\lambda$, and $(K^{(t)}_{\mathsf{pun}}, g^{(t)}_{\mathsf{pun}}, i^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HPunc}^{(t)}(K)$.

We show that Yao's garbled circuit scheme, in fact, gives rise to a decomposable garbling scheme where the security loss and size of the hybrid circuits scales with the depth of the garbled circuits.

Theorem 4.1. *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a class of boolean circuits where each $C \in \mathcal{C}_\lambda$ has circuit size at most $L(\lambda)$, input size at most $n(\lambda)$, depth at most $d(\lambda)$, fan-out at most $\varphi(\lambda)$, and decomposition circuit of size at most $\Delta(\lambda)$. Then assuming the existence of δ-secure one-way functions, $\mathcal{C}$ has a decomposable garbling scheme with (σ, τ, δ)-decomposable indistinguishability where the bound on the size of hybrid circuits is $\sigma = \mathrm{poly}(\lambda, d, \log L, \varphi, \Delta)$, the number of hybrids is $\tau = L \cdot 2^{O(d)}$, and the indistinguishability gap is $\delta^{\Omega(1)}$.*

The proof of the above theorem spans the rest of this section. We rely heavily on the ideas of Hemenway et al. [HJO+15] which considered an orthogonal question of adaptively secure garbling schemes but (for entirely different reasons) developed ideas that are useful for decomposable garbling.

4.2 Yao's Garbled Circuits: Construction

Let $(\mathsf{SEnc}, \mathsf{SDec})$ be a CPA-secure symmetric key encryption scheme with key space $\{0,1\}^\lambda$. Furthermore assume it satisfies *special correctness* so that for all messages m we have:

$$\Pr[\mathsf{SDec}_k(\mathsf{SEnc}_{k'}(m)) \neq \perp \ : \ k, k' \leftarrow \{0,1\}^\lambda] = \mathrm{negl}(\lambda).$$

Such encryption schemes can be based on one-way functions.

Let $\mathcal{PPRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Ev}, \mathsf{PRF.Punc})$ be a PPRF with:

- domain $\mathcal{D} = ([W] \times \{0,1\}) \cup [L]$, where $[W]$ is the set of wires and $[L]$ is the set of gates
- range $\mathcal{R} = \{0,1\}^\lambda$

and furthermore assume that the keys output by $\mathsf{PRF.Gen}$ are just random values $K \in \{0,1\}^\lambda$. Such PPRFs can be constructed based on one-way functions.

Yao's Garbled Circuit Construction. The key K of the garbling scheme is just a key for the PPRF. We define the two functions $\mathsf{Gar.InpEn}, \mathsf{Gar.CirEn}$ as follows.

- $\widehat{b}_i \leftarrow \mathsf{Gar.InpEn}(i, b; K)$: Output $\widehat{b}_i = \mathsf{PRF.Ev}_K((i,b))$.
- $\widehat{G}_g \leftarrow \mathsf{Gar.CirEn}(C_{\mathsf{de}}, g; K)$: Let $C_{\mathsf{de}}(g) = (f, w_a, w_b, w_c)$. Compute the 6 wire labels $k_\alpha^\beta = \mathsf{PRF.Ev}_K(w_\alpha, \beta)$ for $\alpha \in \{a, b, c\}$ and $\beta \in \{0,1\}$, unless w_c is an output wire in which case set $k_c^0 = 0$, $k_c^1 = 1$. Compute 4 ciphertexts:

  $$\begin{aligned}
  c_{0,0} &\leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^0}(k_c^{f(0,0)}))\\
  c_{0,1} &\leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^1}(k_c^{f(0,1)}))\\
  c_{1,0} &\leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^0}(k_c^{f(1,0)}))\\
  c_{1,1} &\leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^0}(k_c^{f(1,1)}))
  \end{aligned}$$

  where the encryption randomness is derived using $\mathsf{PRF.Ev}_K(g)$. Set $\widehat{G}_g = [c_1, c_2, c_3, c_4]$ to be a lexicographic ordering of the above ciphertexts.
- $y \leftarrow \mathsf{Gar.De}(\widehat{C}, \widehat{b})$: On input garbled gates $\widehat{C} = \left\{\widehat{G}_g\right\}_{g\in[L]}$ and garbled input bits $\widehat{b} = \left\{\widehat{b}_i\right\}_{i\in[n]}$, iteratively compute a wire-label for every wire in the circuit.

For the input wires, the labels these are given by $\widehat{b}$. For any gate g with incoming wires w_a, w_b and outgoing wire w_c such that the labels k_a, k_b of w_a, w_b have already been computed, derive the label k_c of w_c as follows. Parse $\widehat{G}_g = [c_1, c_2, c_3, c_4]$ and set $k_c = \mathsf{SDec}_{k_b}(\mathsf{SDec}_{k_a}(c_i))$ for the first one of $i = 1, 2, 3, 4$ for which the outcome is not $\perp$.

Finally, output the value y which is the wire label of the output wire.

The correctness of this construction follows by inspection and from the special correctness of the symmetric key encryption scheme.

4.3 Hybrid Circuits and Pebbling

We now show that Yao's construction is decomposable by defining the hybrid functions as in definition 4.2.

Half-Sequence. We now define the sequence of hybrid functions

$$\left\{\mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)}, \mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in [\tau]\right\}.$$

We will actually only define a *half-sequence* which ensures that:

- $\mathsf{Gar.CirEn}(C_{\mathsf{de}}, g; K) = \mathsf{Gar.HCirEn}^{(1)}(g; K)$,
  $\mathsf{Gar.InpEn}(i, b; K) = \mathsf{Gar.HInpEn}^{(1)}(i, b; K) = \mathsf{Gar.HInpEn}^{(\tau)}(i, b; K)$,
- $\mathsf{Gar.HCirEn}^{(\tau)}(g; K)$ does not depend on C_{de} but only on its topology Φ and the output bit $C(x) = C'(x)$.

By adding a symmetric sequence of additional hybrid functions

$$\left\{\mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)}, \mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in \{\tau+1, \ldots, 2\tau-1\}\right\}$$

where we define the functions with $t = \tau + i$ the same way as those with $t = \tau - i$ but with C'_{de} instead of C_{de} we then get the full hybrid sequence satisfying the definition.

Pebbling. We define the following pebbling game over a circuit with some topology Φ. Each gate of the circuit can either have no pebble, a black pebble, or a gray pebble. The rules of the game are as follows:

I. We can place or remove a black pebble on a gate as long as both predecessors of that gate have black pebbles on them (or the gate is an input gate).

II. We can replace a black pebble with a gray pebble on any non-output gate as long as all successors of that gate have black or gray pebbles on them.

A pebbling of the topology Φ is a sequence of steps where we start with no pebbles on any gate and sequentially apply rules I and II so as to end up with a gray pebble on every non-output gate and a black pebble on the output. Let us assume that the topology Φ admits a pebbling consisting of steps $\mathsf{pstep}_1, \ldots, \mathsf{pstep}_{\tau'}$.

Pebbling and Circuit Configurations. We will define the hybrids, each of which being parametrized by a circuit configuration conf. For every gate g, the configuration conf specifies if the gate is in one of four possible modes

$$\{\mathsf{RealGate}, \mathsf{SimGate}, \mathsf{CompDepSimGate}, \frac{1}{2}\mathsf{CompDepSimGate}\}.$$

Furthermore, for every wire w which is an incoming or outgoing wire of some gate in mode $\mathsf{CompDepSimGate}$ or $\frac{1}{2}\mathsf{CompDepSimGate}$, the configuration specifies a bit $v(w)$.

Assume we are given an input x and a circuit C having topology Φ that admits a pebbling consisting of steps $\mathsf{pstep}_1, \ldots, \mathsf{pstep}_{\tau'}$. We first define a sequence of configurations $\mathsf{conf}'_1, \ldots, \mathsf{conf}'_{\tau'}$ where every configuration conf'_i corresponds to the pebble placement after step i of the pebbling: every gate that has no pebble is in $\mathsf{RealGate}$ mode, every gate that has a black pebble is in $\mathsf{CompDepSimGate}$ mode, every gate that has a gray pebble is in $\mathsf{SimGate}$ mode. Furthermore, for every wire w which is an incoming or outgoing wire of some gate in mode $\mathsf{CompDepSimGate}$ we set $v(w)$ to be the bit going over the wire w during the computation $C(x)$. Next we add several intermediate configurations as follows: if the transition from $\mathsf{conf}'_i, \mathsf{conf}'_{i+1}$ corresponds to an application of rule I in the pebbling, meaning that we either add or remove a black pebble on some gate g, then we add an intermediate configuration in between these, which is identical to $\mathsf{conf}'_i, \mathsf{conf}'_{i+1}$ except that the gate g is in $\frac{1}{2}\mathsf{CompDepSimGate}$ mode. This results in a sequence $\mathsf{conf}_1, \ldots, \mathsf{conf}_\tau$ where $\tau \leq 2\tau'$.

Configurations and Hybrids. We now define the hybrid functions

$$\left\{ \mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)}, \mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in [\tau] \right\}.$$

First, for a configuration conf we define the functions $\mathsf{Gar.HInpEn}^{(\mathsf{conf})}$ and $\mathsf{Gar.HCirEn}^{(\mathsf{conf})}$ parameterized by conf. For every $t \in [\tau]$ we then define $\mathsf{Gar.HInpEn}^{(t)} = \mathsf{Gar.HInpEn}^{(\mathsf{conf}_t)}$ and $\mathsf{Gar.HCirEn}^{(t)} = \mathsf{Gar.HCirEn}^{(\mathsf{conf}_t)}$ where $\mathsf{conf}_1, \ldots, \mathsf{conf}_\tau$ is the sequence of configuration given by the pebbling game as above.

- $\widehat{G}_g \leftarrow \mathsf{Gar.HCirEn}^{(\mathsf{conf})}(g; K)$: Let $C_{\mathsf{de}}(g) = (f, w_a, w_b, w_c)$.

  If the key K is a standard PPRF key then proceed as follows. Compute the 6 wire labels $k^\beta_\alpha = \mathsf{PRF.Ev}_K(w_\alpha, \beta)$ for $\alpha \in \{a, b, c\}$ and $\beta \in \{0, 1\}$, unless w_c is an output wire in which case set $k^0_c = 0$, $k^1_c = 1$. Compute 4 ciphertexts $c_{0,0}$, $c_{0,1}$, $c_{1,0}$, $c_{1,1}$ as in Figure 2 depending on the mode of g, where the encryption randomness is derived from $\mathsf{PRF.Ev}_K(g)$. Let $\widehat{G}_g = [c_1, c_2, c_3, c_4]$ to be a lexicographic ordering of the above ciphertexts.

  If the key K is not a standard PPRF key then parse $K = (K\{S\}, H)$. If there is a tuple of the form $(g, \widehat{G}_g)$ in H then output $\widehat{G}_g$. Else attempt to compute $\widehat{G}_g$ as above using the punctured PPRF key $K\{S\}$ and output $\perp$ if this is not possible.

- $\mathsf{Gar.HInpEn}^{(\mathsf{conf})}(i, b; K)$: If the key K is a standard PPRF key then output $\widehat{b}_i = \mathsf{PRF.Ev}_K((i, b))$. Else parse $K = (K\{S\}, H)$ and output $\widehat{b}_i = \mathsf{PRF.Ev}_{K\{S\}}((i, b))$ or $\perp$ if this computation fails.

We define $\mathsf{Gar.HPunc}^{(t)}(K)$ as follows. Assume that the transition from conf_t to conf_{t+1} changes the mode of a gate $g^{(t)}_{\mathsf{pun}}$ such that $g^{(t)}_{\mathsf{pun}}$ has incoming wires w_a, w_b and outgoing wire w_c.

- If the transition changes $g^{(t)}_{\mathsf{pun}}$ from $\mathsf{RealGate}$ to $\frac{1}{2}\mathsf{CompDepSimGate}$ (or vice versa):
  $\mathsf{Gar.HPunc}^t(K) := (K^t_{\mathsf{pun}} = ((K\{S\}, H), g^{(t)}_{\mathsf{pun}}, i_a)$ where $S = \{(w_a, 1 - v(w_a)), g^{(t)}_{\mathsf{pun}}\}$, $K\{S\} = \mathsf{PRF.Punc}(K, S)$, $i_a = \perp$ if w_a is not an input wire, else i_a is the index of input wire w_a. For every gate $g \neq g^{(t)}_{\mathsf{pun}}$ that has w_a as an input wire compute $\widehat{G}_g \leftarrow \mathsf{Gar.HCirEn}^{(\mathsf{conf}_t)}(g; K)$ and add $\widehat{G}_g$ to H.

- If the transition changes $g^{(t)}_{\mathsf{pun}}$ from $\frac{1}{2}\mathsf{CompDepSimGate}$ to $\mathsf{CompDepSimGate}$ (or vice versa):
  $\mathsf{Gar.HPunc}^t(K) := (K^t_{\mathsf{pun}} = ((K\{S\}, H), g^{(t)}_{\mathsf{pun}}, i_b)$ where $S = \{(w_b, 1 - v(w_b)), g^{(t)}_{\mathsf{pun}}\}$, $K\{S\} =$

RealGate	SimGate	CompDepSimGate
$c_{0,0} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^0}(k_c^{f(0,0)}))$	$c_{0,0} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^0}(k_c^0))$	$c_{0,0} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^0}(k_c^{v(w_c)}))$
$c_{0,1} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^1}(k_c^{f(0,1)}))$	$c_{0,1} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^1}(k_c^0))$	$c_{0,1} \leftarrow \mathsf{SEnc}_{k_a^0}(\mathsf{SEnc}_{k_b^1}(k_c^{v(w_c)}))$
$c_{1,0} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^0}(k_c^{f(1,0)}))$	$c_{1,0} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^0}(k_c^0))$	$c_{1,0} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^0}(k_c^{v(w_c)}))$
$c_{1,1} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^1}(k_c^{f(1,1)}))$	$c_{1,1} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^1}(k_c^0))$	$c_{1,1} \leftarrow \mathsf{SEnc}_{k_a^1}(\mathsf{SEnc}_{k_b^1}(k_c^{v(w_c)}))$

$\frac{1}{2}$CompDepSimGate
$c_{1-v(w_a),0} \leftarrow \mathsf{SEnc}_{k_a^{1-v(w_a)}}(\mathsf{SEnc}_{k_b^0}(k_c^{v(w_c)})$
$c_{1-v(w_a),1} \leftarrow \mathsf{SEnc}_{k_a^{1-v(w_a)}}(\mathsf{SEnc}_{k_b^1}(k_c^{v(w_c)}))$
$c_{v(w_a),0} \leftarrow \mathsf{SEnc}_{k_a^{v(w_a)}}(\mathsf{SEnc}_{k_b^0}(k_c^{f(v(w_a),0)}))$
$c_{v(w_a),1} \leftarrow \mathsf{SEnc}_{k_a^{v(w_a)}}(\mathsf{SEnc}_{k_b^1}(k_c^{f(v(w_a),1)}))$

Figure 2: Garbling Gate modes: RealGate, SimGate, CompDepSimGate, $\frac{1}{2}$CompDepSimGate .

$\mathsf{PRF.Punc}(K, S)$, $i_b = \perp$ if w_b is not an input wire, else i_b is the index of input wire w_b. For every gate $g \neq g^{(t)}_{\mathsf{pun}}$ that has w_b as an input wire compute $\widehat{G}_g \leftarrow \mathsf{Gar.HCirEn}^{(\mathsf{conf}_t)}(g; K)$ and add $\widehat{G}_g$ to H.

- If the transition changes $g^{(t)}_{\mathsf{pun}}$ from CompDepSimGate to SimGate (or vice versa):
  $\mathsf{Gar.HPunc}^t(K) := (K^t_{\mathsf{pun}} = ((K\{S\}, H), g^{(t)}_{\mathsf{pun}}, \perp)$ with S being the set $\{(w_c, 0), (w_c, 1), g^{(t)}_{\mathsf{pun}}\}$, $K\{S\} = \mathsf{PRF.Punc}(K, S)$. For every gate $g \neq g^{(t)}_{\mathsf{pun}}$ that has w_c as an input wire compute $\widehat{G}_g \leftarrow \mathsf{Gar.HCirEn}^{(\mathsf{conf}_t)}(g; K)$ and add $\widehat{G}_g$ to H.

4.4 Proof of Security

We now show that the above hybrid functions satisfy properties (a), (b), (c) of Definition 4.2.

Property (a). Firstly, $\left\{\mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)}, \mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in [\tau]\right\}$ represents a valid half-sequence, meaning that:

- $\mathsf{Gar.CirEn}(C_{\mathsf{de}}, g; K) = \mathsf{Gar.HCirEn}^{(1)}(g; K)$,
  $\mathsf{Gar.InpEn}(i, b; K) = \mathsf{Gar.HInpEn}^{(1)}(i, b; K) = \mathsf{Gar.HInpEn}^{(\tau)}(i, b; K)$,
- $\mathsf{Gar.HCirEn}^{(\tau)}(g; K)$ does not depend on C_{de} but only on its topology Φ and the output bit $C_{\mathsf{de}}(x)$.

In particular, $\mathsf{Gar.HCirEn}^{(\tau)}(g; K) = \mathsf{Gar.HCirEn}^{(\mathsf{conf}_\tau)}(g; K)$ corresponds to a configuration where all non-output gates are in SimGate mode and the output gate is in CompDepSimGate mode, meaning that $\mathsf{Gar.HCirEn}^{(\mathsf{conf}_\tau)}(g; K)$ does not depend on the boolean function f implemented by any of the gates. Furthermore, the configuration conf_τ only specifies the output value $v(w_{out}) = C(x)$. Therefore, these functions are a valid half-sequence. As we discussed, by adding a symmetric half-sequence in the reverse direction for the circuit C'_{de} we get a sequence that satisfies property (a).

Property (b). This property follows by inspection. In particular,

- For any $g \neq g^{(t)}_{\mathsf{pun}}$, we have

$$\mathsf{Gar.HCirEn}^{(t)}(g; K) = \mathsf{Gar.HCirEn}^{(t)}(g; K^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HCirEn}^{(t+1)}(g, K) \ .$$

  Since $\mathsf{Gar.HCirEn}^{(t)}(g; K)$ and $\mathsf{Gar.HCirEn}^{(t+1)}(g, K)$ only differ on the mode of the gate $g^{(t)}_{\mathsf{pun}}$ and the punctured key $K^{(t)}_{\mathsf{pun}}$ hardwires the value of all other gates affected by PRF puncturing to match those of $\mathsf{Gar.HCirEn}^{(t)}(g; K)$

- For any $i \neq i^{(t)}_{\mathsf{pun}}$ and $b \in \{0, 1\}$, we have $\mathsf{Gar.HInpEn}^{(t)}(i, b; K) = \mathsf{Gar.HInpEn}^{(t)}(i, b; K^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HInpEn}^{(t+1)}(i, b, K)$. Actually, we have $\mathsf{Gar.HInpEn}^{(t)}(i, b; K) = \mathsf{Gar.InpEn}(i, b; K)$ as well as $\mathsf{Gar.HInpEn}^{(t)}(i, b; K^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.InpEn}(i, b; K)$, for all $i \neq i^{(t)}_{\mathsf{pun}}$.

Property (c). For property (c), we must show that

$$\begin{aligned}&\left(\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.HInpEn}^{(t)}(i^{(t)}_{\mathsf{pun}}, x_{i^{(t)}_{\mathsf{pun}}}; K), K^{(t)}_{\mathsf{pun}}\right)\\ \approx &\left(\mathsf{Gar.HCirEn}^{(t+1)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.HInpEn}^{(t+1)}(i^{(t)}_{\mathsf{pun}}, x_{i^{(t)}_{\mathsf{pun}}}; K), K^{(t)}_{\mathsf{pun}}\right)\end{aligned}$$

are computationally indistinguishable, where the probability is over $K \leftarrow \{0, 1\}^\lambda$, and $(K^{(t)}_{\mathsf{pun}}, g^{(t)}_{\mathsf{pun}}, i^{(t)}_{\mathsf{pun}}) = \mathsf{Gar.HPunc}^{(t)}(K)$.

Since in our case $\mathsf{Gar.HInpEn}^{(t)}(i, b; K) = \mathsf{Gar.HInpEn}^{(t+1)}(i, b; K) = \mathsf{Gar.InpEn}(i, b; K)$ for all i, b we can simplify the above to:

$$\begin{aligned}&\left(\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.InpEn}(i^{(t)}_{\mathsf{pun}}, x_{i^{(t)}_{\mathsf{pun}}}; K), K^{(t)}_{\mathsf{pun}}\right)\\ \approx &\left(\mathsf{Gar.HCirEn}^{(t+1)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.InpEn}(i^{(t)}_{\mathsf{pun}}, x_{i^{(t)}_{\mathsf{pun}}}; K), K^{(t)}_{\mathsf{pun}}\right)\end{aligned}$$

We consider several cases. Assume that the transition from conf_t to conf_{t+1} changes the mode of a gate $g^{(t)}_{\mathsf{pun}}$ such that $g^{(t)}_{\mathsf{pun}}$ has incoming wires w_a, w_b and outgoing wire w_c.

Case 1: If the transition changes $g^{(t)}_{\mathsf{pun}}$ from $\mathsf{RealGate}$ to $\frac{1}{2}\mathsf{CompDepSimGate}$ (or vice versa):

The difference between $\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.HCirEn}^{(t+1)}(g^{(t)}_{\mathsf{pun}}; K)$ is only in the value encrypted by the ciphertexts:

$$\begin{aligned}c_{1-v(w_a),0} &\leftarrow \mathsf{SEnc}_{k_a^{1-v(w_a)}}(\cdots)\\ c_{1-v(w_a),1} &\leftarrow \mathsf{SEnc}_{k_a^{1-v(w_a)}}(\cdots)\end{aligned}$$

The key $k_a^{1-v(w_a)} = \mathsf{PRF.Ev}_K(w_a, 1 - v(w_a))$ is computed using a PPRF and the encryption randomness is derived using $r = \mathsf{PRF.Ev}_K(g^{(t)}_{\mathsf{pun}})$. Furthermore $K^{(t)}_{\mathsf{pun}} = (K\{S\}, H)$ where $K\{S\}$ is punctured at $S = \{(w_a, 1 - v(w_a)), g^{(t)}_{\mathsf{pun}}\}$. The set H only contains other ciphertexts created under the key $k_a^{1-v(w_a)}$ but otherwise does not contain any other information related to $k_a^{1-v(w_a)}$ or r. Therefore, we can first rely on punctured PRF security to switch $k_a^{1-v(w_a)}$ and r to random, then on CPA security to switch the value encrypted by the ciphertexts $c_{1-v(w_a),0}, c_{1-v(w_a),1}$ from that of $\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K)$ to that of $\mathsf{Gar.HCirEn}^{(t+1)}(g^{(t)}_{\mathsf{pun}}; K)$ and then on punctured PRF security again to switch $k_a^{1-v(w_a)}$ and the encryption randomness back to PRF outputs. This proves indistinguishability.

Case 2: If the transition changes $g^{(t)}_{\mathsf{pun}}$ from $\frac{1}{2}\mathsf{CompDepSimGate}$ to the mode $\mathsf{CompDepSimGate}$ (or vice versa):

This case is identical to the previous one with w_b instead of w_a.

Case 3: If the transition changes $g^{(t)}_{\mathsf{pun}}$ from $\mathsf{CompDepSimGate}$ to $\mathsf{SimGate}$ (or vice versa):

The difference between $\mathsf{Gar.HCirEn}^{(t)}(g^{(t)}_{\mathsf{pun}}; K), \mathsf{Gar.HCirEn}^{(t+1)}(g^{(t)}_{\mathsf{pun}}; K)$ is only a switch from $k_c^{v(w_c)}$ to k_c^0 in the garbled gate $g^{(t)}_{\mathsf{pun}}$.

The keys $k_c^0 = \mathsf{PRF.Ev}_K(w_c, 0)$, $k_c^1 = \mathsf{PRF.Ev}_K(w_c, 1)$ are computed using the PPRF. Furthermore $K^{(t)}_{\mathsf{pun}} = (K\{S\}, H)$ where $K\{S\}$ is punctured at $S = \{(w_c, 0), (w_c, 1)\}$. The keys k_c^0, k_c^1 are used to compute some values in H but they are used in a completely symmetric manner (this is because all successors of $g^{(t)}_{\mathsf{pun}}$ are in $\mathsf{SimGate}, \mathsf{CompDepSimGate}$ modes and therefore the 0 and 1 keys are used identically). Therefore, we first rely on PPRF security to switch k_c^0, k_c^1 to random values, then by symmetry we can exchange $k_c^{v(w_c)}$ for k_c^0 in the garbled gate $g^{(t)}_{\mathsf{pun}}$, and then we rely on PPRF security again to switch them back to PRF outputs.

This completes the proof of property (c).

4.5 Pebble Complexity and Parameters

We now analyze the parameters $\sigma(\Phi, s, \lambda), \tau(\Phi) \in \mathbb{N}, \delta(\lambda)$ as defined in Definition 4.2 that our construction achieves. Assume that the topology Φ has maximum fan-out $\varphi(\Phi)$ and can be pebbled in $\tau'(\Phi)$ steps using at most $p(\Phi)$ black pebbles. Furthermore assume that the pebbling is *succinct* meaning that in any step t of the pebbling there is an index $g^{(t)}_{gray} \in [L]$ such that a gate g has a gray pebble on it if and only if g is not the output gate and $g \geq g^{(t)}_{gray}$. Lastly assume that the size of the compact circuits $C_{\mathsf{de}}, C'_{\mathsf{de}}$ is at most Δ and that the symmetric-key encryption scheme and the PPRF have security $\delta_{ENC}(\lambda)$ and $\delta_{PPRF}(\lambda)$ respectively, meaning that the advantage of any polynomial-time attacker against these schemes is bounded by $\delta_{ENC}(\lambda)$ and $\delta_{PPRF}(\lambda)$ respectively.

Lemma 4.1. *The construction given achieves* $\tau(\Phi) \leq 4\tau'(\Phi)$, $\sigma(\Phi, s, \lambda) \leq \mathrm{poly}(\lambda, \varphi(\Phi), p(\Phi), \log L) + \Delta$, *and* $\delta(\lambda) \leq O(\delta_{ENC}(\lambda) + \delta_{PPRF}(\lambda))$.

Lemma 4.1. It's easy to see that $\tau(\Phi) \leq 4\tau'(\Phi)$ since we added at most one intermediate configuration in between any pebbling steps to define the half-sequence of hybrid functions of length at most $2\tau'(\Phi)$, and therefore the length of the full sequence is of length at most $4\tau'(\Phi)$.

We now compute $\sigma(\Phi, s, \lambda)$ which is a bound on the circuit size of

$$\left\{ \mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)} \;\middle|\; t \in [\tau] \right\}.$$

First, we note that the size of the punctured key K^t_{pun} is $|K^t_{\mathsf{pun}}| \leq \varphi(\Phi)\mathrm{poly}(\lambda)$ since it hardwires at most $\varphi(\Phi)$ garbled gates. Secondly we note that the size of any configuration conf_t can be described by at most $O(p(\Phi)\log L)$ bits since it only needs to specify the $p(\Phi)$ gates that are in $\mathsf{CompDepSimGate}$ mode and in $\frac{1}{2}\mathsf{CompDepSimGate}$ mode, the bits $v(w_c)$ for each outgoing wire w_c of such gates, and the index $g^{(t)}_{gray}$ to be able to decide if a gate is in $\mathsf{SimGate}$ or $\mathsf{RealGate}$ mode. Lastly, the function $\mathsf{Gar.HCirEn}^{(t)}$ needs to run the circuit C_{de} or C'_{de} once and then apply some processing on the output. Therefore, by observation, the circuit size is bounded by $\mathrm{poly}(\lambda, \varphi(\Phi), p(\Phi), \log L) + \Delta$.

Finally, the bound on $\delta(\lambda)$ follows directly from the security proof. □

In [HJO+15] (Section 6, Strategy 2), it is shown that any circuit topology of depth d and circuit size L can be pebbled using $p(\Phi) \leq 2d$ black pebbles and in $\tau'(\Phi) \leq L4^d$ steps. Furthermore, it is easy to see that this pebbling is succinct.[4] Therefore, by plugging these parameters into Lemma 4.1 we get the proof of Theorem 4.1.

5 Single-Key Succinct PKFE from SXIO and PKE

This section consists of four subsections. The main part is constructing a weakly succinct PKFE scheme for boolean functions in Section 5.2. First, in Section 5.1, we construct a succinct IBE scheme with γ-compression, which we shall use as a building block in Section 5.2. In Section 5.3, we present a transformation from weakly succinct PKFE schemes for boolean functions into ones for non-boolean functions. Lastly, we explain how the pieces come together to give IO from SKFE in Section 5.4.

5.1 A Succinct IBE Scheme

In this section, we construct a succinct IBE scheme from SXIO and PKE. Here by *succinct*, we mean that the size of the encryption circuit (and then also of the master public key) is sublinear in the size of the identity space (roughly, $|\mathcal{ID}|^\gamma$, with $\gamma < 1$, see Section 2.6 for the complete definition). In the following subsections, we will use the constructed IBE as a building block in our construction of succinct PKFE.

Theorem 5.1. *For any $\beta < \gamma < 1$, assuming there exists a β-compressing SXIO scheme, a puncturable PRF, and a plain PKE scheme, there exists a succinct IBE scheme with γ-compression. Moreover, assuming the underlying primitives are δ-secure so is the resulting IBE scheme.*

We start by describing the IBE construction and then argue its security. Let $\mathcal{PPRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Ev}, \mathsf{PRF.Punc})$ be a puncturable PRF, $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ a PKE scheme, and $\mathsf{sxi}\mathcal{O}$ a β-compressing SXIO scheme. Let $[s]$ denote the identity space.

Construction. The scheme $\mathsf{IBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for an identity space $\mathcal{ID} = [s]$ is given by the following algorithms.

$\mathsf{Setup}(1^\lambda)$**:**

- Choose puncturable PRF key $S \leftarrow \mathsf{PRF.Gen}(1^\lambda)$.
- Construct a circuit $\mathsf{KG}_{\mathsf{ibe}}[S]$ as described in Figure 3 that computes $r_i = \mathsf{PRF.Ev}_S(i)$ for an input $i \in [s]$ and generates a pair of keys $(\mathsf{PKE.PK}_i, \mathsf{PKE.SK}_i) \leftarrow \mathsf{PKE.Gen}(1^\lambda; r_i)$.
- Output $\mathsf{MPK} = \mathsf{sxi}\mathcal{O}(\mathsf{KG}_{\mathsf{ibe}}[S])$ and $\mathsf{MSK} = \{\mathsf{PKE.SK}_i\}_{i\in[s]}$.

$\mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$**:**

- Parse $\mathsf{MSK} = \{\mathsf{PKE.SK}_i\}_{i\in[s]}$.
- Output $\mathsf{SK}_{\mathsf{id}} = \mathsf{PKE.SK}_{\mathsf{id}}$.

[4] Indeed, if the gates $1, \ldots, L$ are topologically sorted so that L is the output gate, then the pebbling can be made to place gray pebbles in according to this ordering, meaning that the first gray pebble is placed on gate $L-1$ then $L-2$ and finally continuing down to 1. One minor difference between our version and the one in [HJO+15] is that the latter allows replacing a black pebble with a gray pebble at the output gate whereas our does not. This only requires us to slightly modify the pebbling strategy to keep the pebble at the output gate black.

$\mathsf{Enc}(\mathsf{MPK}, \mathsf{id}, x)$**:**

- Compute $\mathsf{PKE.PK}_{\mathsf{id}}$ by running $\mathsf{MPK}(\mathsf{id}) = \mathsf{sxi}\mathcal{O}(\mathsf{KG}_{\mathsf{ibe}}[S])(\mathsf{id})$.
- Output $\mathsf{CT} \leftarrow \mathsf{PKE.Enc}(\mathsf{PKE.PK}_{\mathsf{id}}, x)$.

$\mathsf{Dec}(\mathsf{SK}_{\mathsf{id}}, \mathsf{CT})$**:**

- Output $\mathsf{PKE.Dec}(\mathsf{SK}_{\mathsf{id}}, \mathsf{CT})$.

Key Generation Circuit $\mathsf{KG}_{\mathsf{ibe}}[S]$

Hardwired: a puncturable PRF key $S \in \{0,1\}^\lambda$.
Input: an index $i \in [s]$.
Padding: the circuit is padded to size $\mathsf{pad} = \mathsf{pad}(s, \lambda)$, which is determined in the analysis.

1. Compute $r_i \leftarrow \mathsf{PRF.Ev}_S(i)$.
2. Generate a key pair $(\mathsf{PKE.PK}_i, \mathsf{PKE.SK}_i) \leftarrow \mathsf{PKE.Gen}(1^\lambda; r_i)$.
3. Output the public key $\mathsf{PKE.PK}_i$.

Figure 3: Circuit $\mathsf{KG}_{\mathsf{ibe}}[S]$

Proof of Theorem 5.1. We now prove Theorem 5.1, starting by analyzing succinctness, and then moving on to the security proof.

Padding Parameter

The proof of security relies on indistinguishability of obfuscations of circuits $\mathsf{KG}_{\mathsf{ibe}}$ and $\mathsf{KG}^*_{\mathsf{ibe}}$ defined in Figures 3 and 4. Accordingly, we set $\mathsf{pad} = \max(|\mathsf{KG}_{\mathsf{ibe}}|, |\mathsf{KG}^*_{\mathsf{ibe}}|)$. The circuits $\mathsf{KG}_{\mathsf{ibe}}$ and $\mathsf{KG}^*_{\mathsf{ibe}}$ compute a puncturable PRF over domain $[s]$, a PKE key pair, and may have punctured PRF keys and public keys hardwired. Thus,

$$\mathsf{pad} \leq \mathrm{poly}(\lambda, \log(s)) \ .$$

Succinctness

The input space for $\mathsf{KG}_{\mathsf{ibe}}$ is $[s]$. Therefore, by the SXIO guarantee, the size of the encryption circuit (dominated by running the obfuscated $\mathsf{KG}_{\mathsf{ibe}}$) is

$$s^\beta \cdot \mathrm{poly}(\lambda, \ell, \log s) \leq s^\gamma \cdot \mathrm{poly}(\lambda, \ell) \ ,$$

where ℓ is a bound on the length of encrypted messages.

Security Proof

Let us assume that the underlying primitives are δ-secure. We define a sequence of hybrid games.

Hyb_0**:** The first game is the original selective security experiment for $b = 0$, $\mathsf{Expt}^{\mathsf{IBE}}_{\mathcal{A}}(1^\lambda, 0)$. In this game, the adversary first selects the challenge identity id^* and messages (m^*_0, m^*_1) and then gets an encryption of m^*_0 for identity id^* and the master public key. It can also query polynomially many secret keys for identities different from id^* (See Definition 2.11 for more details).

Hyb_1**:** We change $\mathsf{KG}_{\mathsf{ibe}}$ into $\mathsf{KG}^*_{\mathsf{ibe}}$ described in Figure 4. In this hybrid game, we set $r^* = \mathsf{PRF.Ev}_S(\mathsf{id}^*)$ and $(\mathsf{PKE.PK}^*, \mathsf{PKE.SK}^*) \leftarrow \mathsf{PKE.Gen}(1^\lambda; r^*)$. Thus, The behaviors of $\mathsf{KG}_{\mathsf{ibe}}$ and $\mathsf{KG}^*_{\mathsf{ibe}}$ are totally the same, and so are their size since we pad circuit $\mathsf{KG}_{\mathsf{ibe}}$ to have the same size as $\mathsf{KG}^*_{\mathsf{ibe}}$. Then, we can use the indistinguishability guarantee of $\mathsf{sxi}\mathcal{O}$ and it holds $\mathsf{Hyb}_0 \approx_\delta \mathsf{Hyb}_1$.

Key generation circuit $\mathsf{KG}^*_{\mathsf{ibe}}[S\{\mathsf{id}^*\}]$

Hardwired: a punctured PRF key $S\{\mathsf{id}^*\}$, a key $\mathsf{PKE.PK}^*$.
Input: an index $i \in [s]$.

1. If $i = \mathsf{id}^*$, then output $\mathsf{PKE.PK}^*$.
2. Else if compute $r_i \leftarrow \mathsf{PRF.Ev}_{S\{\mathsf{id}^*\}}(i)$.
3. Generate a key pair $(\mathsf{PKE.PK}_i, \mathsf{PKE.SK}_i) \leftarrow \mathsf{PKE.Gen}(1^\lambda; r_i)$.
4. Output PKE public key $\mathsf{PKE.PK}_i$.

Figure 4: Circuit $\mathsf{KG}^*_{\mathsf{ibe}}[S\{\mathsf{id}^*\}]$

Hyb_2**:** We change $r^* = \mathsf{PRF.Ev}_S(\mathsf{id}^*)$ into uniformly random r^*. Due to the pseudorandomness at punctured points, it holds $\mathsf{Hyb}_1 \approx_\delta \mathsf{Hyb}_2$.

Hyb_3**:** We change CT^* from $\mathsf{PKE.Enc}(\mathsf{id}^*, m^*_0)$ to $\mathsf{PKE.Enc}(\mathsf{id}^*, m^*_1)$. In Hyb_2 and Hyb_3, we do not need randomness used to generate $\mathsf{PKE.PK}^*$. We just use the hardwired $\mathsf{PKE.PK}^*$. Therefore, $\mathsf{Hyb}_2 \approx_\delta \mathsf{Hyb}_3$ follows directly from the semantic security of the PKE scheme.

Hyb_4**:** This is $\mathsf{Expt}^{\mathsf{IBE}}_{\mathcal{A}}(1^\lambda, 1)$. We can show the indistinguishability between Hyb_3 and Hyb_4 in a reverse manner.

This completes the proof of Theorem 5.1. □

5.2 Weakly Succinct PKFE for Boolean Functions

We now construct a single-key weakly succinct PKFE scheme for the class of boolean functions. The construction is based on succinct IBE, decomposable garbling, and SXIO.

Theorem 5.2. *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of circuits with a single output bit and let $n(\lambda), s(\lambda), d(\lambda)$ be bounds on their input length, size, and depth (respectively). For any constants β, γ such that $3\beta < \gamma < 1$,*

assuming a δ-secure, β-compressing SXIO, there exists a constant α, such that given any δ-secure, α-compressing IBE, and δ-secure one-way functions, there exists a $2^d s\delta$-secure succinct PKFE for $\mathcal{C}$ with compression factor γ.

Depth preserving universal circuits. To prove the above theorem, we recall the existence of depth preserving universal circuits [CH85]. Concretely, any family of circuits $\mathcal{C}$ as considered in Theorem 5.2 has a uniform family of universal circuits $\{U_\lambda\}_{\lambda\in\mathbb{N}}$ with fan-out 2,[5] depth $d \cdot \mathrm{polylog}(s)$, and size $s^3 \cdot \mathrm{polylog}(s)$, for some fixed polynomial poly. Each such circuit takes as input a description $(f_1, \ldots, f_s)$ of a function in $\mathcal{C}$ and an input $(x_1, \ldots, x_n)$ and outputs $f(x)$. Furthermore, uniformity here means that each circuit has a decomposition circuit of size $\mathrm{polylog}(s)$.

Ingredients and notation used in the construction.

- We denote by $U^{(x)} : \{0,1\}^s \to \{0,1\}$ the universal circuit, with $x \in \{0,1\}^n$ being a hardwired bitstring, such that on input $(f_1, \ldots, f_s)$, the circuit $U^{(x)}$ outputs $f(x)$. This circuit has a decomposition circuit of size $\mathrm{poly}(n, \log(s))$, which we denote by $U^{(x)}_{\mathsf{de}}$. We also denote by L the number of gates in the circuit $U^{(x)}$.
- Let $\mathsf{sxi}\mathcal{O}$ be a δ-secure, β-compressing SXIO scheme.
- Let $\mathsf{IBE} = (\mathsf{IBE.Setup}, \mathsf{IBE.KeyGen}, \mathsf{IBE.Enc}, \mathsf{IBE.Dec})$ be δ-secure, succinct, IBE scheme with α-compression for the identity space being $\mathcal{ID} = [s] \times \{0,1\}$.
- Let $(\mathsf{Gar.CirEn}, \mathsf{Gar.InpEn}, \mathsf{Gar.De})$ be a decomposable garbling scheme with (σ, τ, δ)-decomposable indistinguishability where $\sigma = \mathrm{poly}(\lambda, n, d, \log(s))$ and $\tau = s2^{O(d)}$. Such schemes are implied by δ-secure one-way functions (Theorem 4.1).
- Let $\mathcal{PPRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Ev}, \mathsf{PRF.Punc})$ be a δ-secure puncturable PRF. These are implied by δ-secure one-way functions (Section 2.4).

Construction. The scheme consists of the following algorithms.

$\mathsf{PKFE.Setup}(1^\lambda)$**:**

- Run $(\mathsf{MSK}_{\mathsf{ibe}}, \mathsf{MPK}_{\mathsf{ibe}}) \leftarrow \mathsf{IBE.Setup}(1^\lambda)$.
- Set $\mathsf{MSK} = \mathsf{MSK}_{\mathsf{ibe}}$, $\mathsf{MPK} = \mathsf{MPK}_{\mathsf{ibe}}$.

$\mathsf{PKFE.Key}(\mathsf{MSK}, f)$**:**

- Compute $\mathsf{SK}_{i,f_i} \leftarrow \mathsf{IBE.KeyGen}(\mathsf{MSK}_{\mathsf{ibe}}, (i, f_i))$ for $i \in [s]$, where $f = (f_1, \ldots, f_s)$.
- Return $\mathsf{SK}_f = \{\mathsf{SK}_{i,f_i}\}_{i\in[s]}$.

$\mathsf{PKFE.Enc}(\mathsf{MPK}, x)$**:**

- Compute $U^{(x)}_{\mathsf{de}}$ and pick a garbling key $K \leftarrow \{0,1\}^\lambda$ and a punctured key $S \leftarrow \mathsf{PRF.Gen}(1^\lambda)$;
- Generate an obfuscation $\widetilde{\mathsf{IGC}} = \mathsf{sxi}\mathcal{O}(1^\lambda, \mathsf{IGC}[K, S, \mathsf{MPK}])$ of the input garbling circuit defined in Figure 5;

[5]The restriction regarding fan-out is not stated explicitly in [CH85], but can always be achieved by blowing up the size and depth by a factor of at most $\mathrm{polylog}(s)$.

- Generate an obfuscation $\widetilde{\mathsf{GGC}} = \mathsf{sxi}\mathcal{O}(1^\lambda, \mathsf{GGC}[K, U^{(x)}_{\mathsf{de}}])$ of the gate garbling circuit defined in Figure 6;
- Return $\mathsf{CT}_x = (\widetilde{\mathsf{IGC}}, \widetilde{\mathsf{GGC}})$.

$\mathsf{PKFE.Dec}(\mathsf{SK}_f, \mathsf{CT}_x)$**:**

- For $i \in [s]$, run $\widetilde{\mathsf{IGC}}(i, f_i)$ to obtain an IBE ciphertext, and decrypt the output using SK_{i,f_i} to obtain $\hat{f}_i$.
- For all $g \in [L]$, run $\widetilde{\mathsf{GGC}}(g)$, in order to obtain the garbled gate $\widehat{G}_g$.
- Return $y \leftarrow \mathsf{Gar.De}(\widehat{C}, \hat{f})$, with $\widehat{C} = \left\{\widehat{G}_g\right\}_{g\in[L]}$ and $\hat{f} = \left\{\hat{f}_i\right\}_{i\in[s]}$.

Input Garbling Circuit $\mathsf{IGC}[K, S, \mathsf{MPK}]$

Hardwired: a garbling key K, a puncturable PRF key S, and $\mathsf{MPK} = \mathsf{MPK}_{\mathsf{ibe}}$.
Input: identity (i, b), consisting of an input label $i \in [s]$ and a bit $b \in \{0, 1\}$.
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{IGC}} = \mathsf{pad}_{\mathsf{IGC}}(s, d, n, \lambda)$, determined in the analysis.

1. Compute a corresponding garbled input bit $\widehat{b}_i = \mathsf{Gar.InpEn}(i, b; K)$.
2. Output an IBE encryption $\mathsf{IBE.Enc}(\mathsf{MPK}_{\mathsf{ibe}}, (i, b), \widehat{b}_i; \mathsf{PRF.Ev}_S(i, b))$.

Figure 5: Circuit $\mathsf{IGC}[K, S, \mathsf{MPK}]$

Gate Garbling Circuit $\mathsf{GGC}[K, U^{(x)}_{\mathsf{de}}]$

Hardwired: a garbling key K and the decomposition circuit $U^{(x)}_{\mathsf{de}}$ of $U^{(x)}$.
Input: a gate label $g \in [L]$.
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{GGC}} = \mathsf{pad}_{\mathsf{GGC}}(s, d, n, \lambda)$, determined in the analysis.

Output $\widehat{G}_g = \mathsf{Gar.CirEn}(U^{(x)}_{\mathsf{de}}, g; K)$.

Figure 6: Circuit $\mathsf{GGC}[K, U^{(x)}_{\mathsf{de}}]$

Proof of Theorem 5.2. We now prove Theorem 5.2, starting by correctness, continuing to succinctness, and ending with the proof of security. In what follows, let s, d, and n be bounds on the size, the depth, and the input length of functions in $\mathcal{C}$.

Correctness

Correctness immediately follows from the correctness of the underlying identity-based encryption and decomposable garbling schemes.

Padding Parameter

The proof of security relies on the indistinguishability guarantee of SXIO with respect to two hybrid sequences of circuits, one corresponding to input garbling, and the other one corresponding to gate garbling. Thus, we pad every circuit of each sequence to the maximal size of any circuit in the sequence. That is, we consider

$$\mathcal{S}_{\mathsf{IGC}} = \{\mathsf{IGC}, \mathsf{HIGC}_t, \mathsf{HIGC}_t^{(j)} | 1 \leq t \leq \tau, 1 \leq j \leq 5\}$$
$$\mathcal{S}_{\mathsf{GGC}} = \{\mathsf{GGC}, \mathsf{HGGC}_t, \mathsf{HGGC}_t^{(j)} | 1 \leq t \leq \tau, 1 \leq j \leq 5\} \ ,$$

and let $\mathsf{pad}_{\mathsf{IGC}} = \max_{C \in \mathcal{S}_{\mathsf{IGC}}} |C|$ and $\mathsf{pad}_{\mathsf{GGC}} = \max_{C \in \mathcal{S}_{\mathsf{GGC}}} |C|$.

We bound $\mathsf{pad}_{\mathsf{IGC}}$ and $\mathsf{pad}_{\mathsf{GGC}}$ as follows:

- Any circuit in $\mathcal{S}_{\mathsf{IGC}}$ consists of the following:
  - A punctured PRF computation (deriving randomness for IBE encryption) over a domain of size $O(s)$, using keys that may be punctured at at most constant number of points. The contribution to the circuit size is bounded by $\mathrm{poly}(\lambda, \log(s))$.
  - Input garbling (possibly via one of the hybrid input garbling circuits). The contribution to the circuit size is bounded by $\sigma = \mathrm{poly}(\lambda, d, \log(s), n)$.
  - IBE encryption of garbled inputs, using a hardwired public key and up to a constant number of hardwired ciphertexts. The contribution to the circuit size is bounded by $s^{\alpha} \cdot \mathrm{poly}(\lambda, \sigma) = s^{\alpha} \cdot \mathrm{poly}(\lambda, d, \log(s), n)$.

  Overall,
  $$\mathsf{pad}_{\mathsf{IGC}} \leq s^{\alpha} \cdot \mathrm{poly}(\lambda, n, d, \log(s)) \ .$$
- Any circuit in $\mathcal{S}_{\mathsf{GGC}}$ performs a gate garbling operation (possibly via one of the hybrid gate garbling circuits). The overall contribution to the circuit size is bounded by
  $$\mathsf{pad}_{\mathsf{GGC}} \leq \sigma \leq \mathrm{poly}(\lambda, n, d, \log(s)) \ .$$

Succinctness

We show that for any $\beta < \gamma < 1$ and an appropriate choice of $\alpha < 1$, the size of the encryption circuit is bounded by $s^{\gamma} \cdot \mathrm{poly}(\lambda, n, d)$.

Let $\mathsf{sxi}\mathcal{O}$ be β-compressing. Let e be the size of the obfuscated circuits $\widetilde{\mathsf{IGC}}$ and $\widetilde{\mathsf{GGC}}$ created during encryption. Since the input spaces of IGC and GGC are respectively $[2s]$ and $[L]$, SXIO guarantees:

$$\begin{aligned} e &\leq (2s)^{\beta} \cdot \mathrm{poly}(\lambda, |\mathsf{IGC}|) + L^{\beta} \cdot \mathrm{poly}(\lambda, |\mathsf{GGC}|) \\ &\leq s^{\beta} \cdot \mathrm{poly}(\lambda, \mathsf{pad}_{\mathsf{IGC}}) + L^{\beta} \cdot \mathrm{poly}(\lambda, \mathsf{pad}_{\mathsf{GGC}}) \ , \end{aligned}$$

where poly is a fixed polynomial. Then, using the above bounds on $\mathsf{pad}_{\mathsf{IGC}}$ and $\mathsf{pad}_{\mathsf{GGC}}$, and denoting by $c = O(1)$ the polynomial blowup in the circuit-size incurred in $\mathsf{sxi}\mathcal{O}$, we obtain:

$$e \leq s^{\beta + c\alpha} \cdot \mathrm{poly}(\lambda, n, d, \log(s)) + L^{\beta} \cdot \mathrm{poly}(\lambda, n, d, \log(s)) \ .$$

Recalling that L is bounded by $s^3 \cdot \mathrm{polylog}(s)$, and that $3\beta < \gamma$, we deduce that

$$e \leq s^{\gamma} \cdot \mathrm{poly}(\lambda, n, d) \ ,$$

provided that we choose α such that $\beta + \alpha c < \gamma$ (which is possible since $\beta < \gamma$ and c is a constant).

Security Proof

Let $x_0, x_1 \in \{0,1\}^n$ denote the challenge messages and f denote the function query provided by the adversary, and assume $f(x_0) = f(x_1)$. Let $U^{(x_0)}$ and $U^{(x_1)}$ be the universal circuits defined similarly to $U^{(x)}$ above, and $U^{(x_0)}_{\mathsf{de}}$ and $U^{(x_1)}_{\mathsf{de}}$ their respective decomposition circuits. Note that $U^{(x_0)}$ and $U^{(x_1)}$ have the same topology (they are the same universal circuit with a different hardwired input of the same size n), and in particular, their decomposition circuits are of the same size. Furthermore, these two circuits also satisfy

$$U^{(x_0)}(f) = f(x_0) = f(x_1) = U^{(x_1)}(f) \ .$$

$U^{(x_0)}$ and $U^{(x_1)}$ satisfy the properties required by the (σ, τ, δ)-decomposable indistinguishability security of the decomposable garbling scheme, and thus there exist circuits $\left\{\mathsf{Gar.HInpEn}^{(t)}, \mathsf{Gar.HCirEn}^{(t)} \;\middle|\; t \in [\tau]\right\}$, whose size is at most σ, as well as (possibly inefficient) hybrid functions $\left\{\mathsf{Gar.HPunc}^{(t)} \;\middle|\; t \in [\tau]\right\}$ as given by Definition 4.1.

We accordingly consider a sequence of $\tau + 2$ hybrid games:

Hyb_0**:** The first game is $\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 0)$, where the adversary hands x_0, x_1, f and gets back from the challenger an encryption of x_0.

Hyb_t **($1 \le t \le \tau$):** The t-th game is defined similarly to $\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 0)$ except that the challenge ciphertext CT^* consists of obfuscations of corresponding hybrid circuits:

$$\mathsf{CT}^* = (\mathsf{sxi}\mathcal{O}(1^\lambda, \mathsf{HIGC}_t[K, S, \mathsf{MPK}]), \mathsf{sxi}\mathcal{O}(1^\lambda, \mathsf{HGGC}_t[K])) \ ,$$

where circuits $\mathsf{HIGC}_t[K, \mathsf{MPK}]$ and $\mathsf{HGGC}_t[K]$ are defined in Figure 7 and Figure 8 respectively, and K is the key used in the decomposable garbling scheme for computing the challenge ciphertext.

$\mathsf{Hyb}_{\tau+1}$**:** The last game is $\mathsf{Expt}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 1)$, where x_1 is encrypted (rather than x_0).

Hybrid Input Garbling Circuit $\mathsf{HIGC}_t[K, S, \mathsf{MPK}]$

Hardwired: a garbling key K, a puncturable PRF key S, and $\mathsf{MPK} = \mathsf{MPK}_{\mathsf{ibe}}$.
Input: an input label $i \in [s]$ and a bit $b \in \{0,1\}$, so an identity (i, b).
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{IGC}}$.

1. Compute the garbled input bit $\widehat{b}_i = \mathsf{Gar.HInpEn}^{(t)}(i, b; K)$.
2. Output an IBE encryption $\mathsf{IBE.Enc}(\mathsf{MPK}_{\mathsf{ibe}}, (i, b), \widehat{b}_i; \mathsf{PRF.Ev}_S(i, b))$.

Figure 7: Circuit $\mathsf{HIGC}_t[K, S, \mathsf{MPK}]$

We first note that $\mathsf{Hyb}_0 \approx_\delta \mathsf{Hyb}_1$ and $\mathsf{Hyb}_\tau \approx_\delta \mathsf{Hyb}_{\tau+1}$. Indeed, by the guarantee of the decomposable garbling scheme (Definition 4.2), the obfuscated circuits in the respective hybrids compute the exact same function (and are padded to the the same size). δ-indistinguishability follows by the SXIO guarantee.

Hybrid Gate Garbling Circuit $\mathsf{HGGC}_t[K]$

Hardwired: a garbling key K.
Input: a gate label $g \in [L]$.
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{GGC}}$.

Output the garbled gate $\widehat{G}_g = \mathsf{Gar.HCirEn}^{(t)}(g; K)$.

Figure 8: Circuit $\mathsf{HGGC}_t[K]$

We now argue that hybrids Hyb_t and Hyb_{t+1}, for $1 \leq t \leq \tau - 1$ are computationally indistinguishable.

Indistinguishability of Hyb_t and Hyb_{t+1} $(1 \leq t \leq \tau - 1)$: The indistinguishability of each such two hybrid games is proven by a sequence of five intermediate hybrid games $\mathsf{Hyb}_t^{(1)}, \ldots, \mathsf{Hyb}_t^{(5)}$ as follows. The five hybrid games only differ from Hyb_t in the way the challenge ciphertext CT^* is computed. For $1 \leq j \leq 5$, the challenge ciphertext is computed in $\mathsf{Hyb}_t^{(j)}$ as the obfuscations of the two circuits $\mathsf{HIGC}_t^{(j)}$ and $\mathsf{HGGC}_t^{(j)}$, described in Figures 9–10, where the role of the hardwired values $K^{(j)}, S^{(j)}, \mathsf{CT}^{(j)}_{i^*,f_{i^*}}, \mathsf{CT}^{(j)}_{i^*,1-f_{i^*}}$ and $\widehat{G}^{(j)}_{g^*}$ is described below. We also let $(K^*, g^*, i^*) \leftarrow \mathsf{Gar.HPunc}^{(t)}(K)$. Then, f_{i^*} denotes the i^*-th bit of the function queried by the adversary.

Hybrid Input Garbling Circuit $\mathsf{HIGC}_t^{(j)}[K^{(j)}, S^{(j)}, \mathsf{MPK}, \mathsf{CT}^{(j)}_{i^*,f_{i^*}}, \mathsf{CT}^{(j)}_{i^*,1-f_{i^*}}]$

Hardwired: MPK, a garbling key $K^{(j)}$, a punctured key $S^{(j)}$, and two outputs $\mathsf{CT}^{(j)}_{i^*,f_{i^*}}, \mathsf{CT}^{(j)}_{i^*,1-f_{i^*}}$.
Input: an input label $i \in [s]$ and a bit $b \in \{0,1\}$, so an identity (i, b).
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{IGC}}$.

1. If $(i, b) = (i^*, f_{i^*})$, then output $\mathsf{CT}^{(j)}_{i^*,f_{i^*}}$.
2. If $(i, b) = (i^*, 1 - f_{i^*})$, then output $\mathsf{CT}^{(j)}_{i^*,1-f_{i^*}}$.
3. Else compute and output $\mathsf{IBE.Enc}(\mathsf{MPK}, (i, b), \mathsf{Gar.HInpEn}^{(t)}(i, b; K^{(j)}); \mathsf{PRF.Ev}_{S^{(j)}}(i, b))$.

Figure 9: Circuit $\mathsf{HIGC}_t^{(j)}[K^{(j)}, S^{(j)}, \mathsf{MPK}, \mathsf{CT}^{(j)}_{i^*,f_{i^*}}, \mathsf{CT}^{(j)}_{i^*,1-f_{i^*}}]$, $1 \leq j \leq 5$

We now prove indistinguishability of Hyb_t and Hyb_{t+1} as follows:

- **Hyb_t to $\mathsf{Hyb}_t^{(1)}$:** in $\mathsf{Hyb}_t^{(1)}$, we fix $S^{(1)} \leftarrow \mathsf{PRF.Punc}(S, \{(i^*, 1 - f_{i^*})\})$ and $K^{(1)} \leftarrow K^*$. We hardwire the value $\mathsf{CT}^{(1)}_{i^*,f_{i^*}} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, f_{i^*}), \mathsf{Gar.HInpEn}^{(t)}(i^*, f_{i^*}; K); \mathsf{PRF.Ev}_{S^{(1)}}(i^*, f_{i^*}))$, as well as $\mathsf{CT}^{(1)}_{i^*,1-f_{i^*}} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, 1-f_{i^*}), \mathsf{Gar.HInpEn}^{(t)}(i^*, 1-f_{i^*}; K); \mathsf{PRF.Ev}_S(i^*, 1-f_{i^*}))$ in the first circuit, and $\widehat{G}^{(1)}_{g^*} \leftarrow \mathsf{Gar.HCirEn}^{(t)}(g^*; K)$ in the second circuit. Therefore, it is immediate that, by definitions, both circuits are functionally equivalent in both games since hardwired values are correctly computed, so the two hybrid games are computationally indistinguishable under

Hybrid Gate Garbling Circuit $\mathsf{HGGC}_t^{(j)}[K^{(j)}, \widehat{G}_{g^*}^{(j)}]$

Hardwired: a garbling key $K^{(j)}$, an output $\widehat{G}_{g^*}^{(j)}$.
Input: a gate label $g \in [L]$.
Padding: the circuit is padded to size $\mathsf{pad}_{\mathsf{GGC}}$.

1. If $g = g^*$, then output $\widehat{G}_{g^*}^{(j)}$.
2. Else compute and output $\mathsf{Gar.HCirEn}^{(t)}(g; K^{(j)})$.

Figure 10: Circuit $\mathsf{HGGC}_t^{(j)}[K^{(j)}, \widehat{G}_{g^*}^{(j)}]$, $1 \leq j \leq 5$

the security of $\mathsf{sxi}\mathcal{O}$. Hence, we have $\mathsf{Hyb}_t \approx_\delta \mathsf{Hyb}_t^{(1)}$.

- $\mathsf{Hyb}_t^{(1)}$ **to** $\mathsf{Hyb}_t^{(2)}$**:** The difference between $\mathsf{Hyb}_t^{(1)}$ and $\mathsf{Hyb}_t^{(2)}$ is that we now define $\mathsf{CT}_{i^*,f_{i^*}}^{(2)} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, f_{i^*}), \mathsf{Gar.HInpEn}^{(t+1)}(i^*, f_{i^*}; K); \mathsf{PRF.Ev}_{S^{(2)}}(i^*, f_{i^*}))$ in the first circuit, and $\widehat{G}_{g^*}^{(2)} \leftarrow \mathsf{Gar.HCirEn}^{(t+1)}(g^*; K)$. Other values remain the same as before. Thus, these two hybrid games are computationally indistinguishable assuming decomposable indistinguishability of the decomposition garbling scheme, and we have $\mathsf{Hyb}_t^{(1)} \approx_\delta \mathsf{Hyb}_t^{(2)}$.
- $\mathsf{Hyb}_t^{(2)}$ **to** $\mathsf{Hyb}_t^{(3)}$**:** In the third hybrid game, we only modify the previous game by letting $\mathsf{CT}_{i^*,1-f_{i^*}}^{(3)} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, 1-f_{i^*}), \mathsf{Gar.HInpEn}^{(t)}(i^*, 1-f_{i^*}; K); r)$, with r being a fresh uniformly random value. Other values remain the same as before. Assuming $\mathsf{PRF.Ev}$ is a δ-secure puncturable pseudorandom function, these two hybrid games are computationally indistinguishable, and we have $\mathsf{Hyb}_t^{(2)} \approx_\delta \mathsf{Hyb}_t^{(3)}$.
- $\mathsf{Hyb}_t^{(3)}$ **to** $\mathsf{Hyb}_t^{(4)}$**:** In the fourth hybrid game, we define $\mathsf{CT}_{i^*,1-f_{i^*}}^{(4)} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, 1-f_{i^*}), \mathsf{Gar.HInpEn}^{(t+1)}(i^*, 1-f_{i^*}; K); r)$, with r being a fresh uniformly random value. Other values remain the same as before. Assuming the security of the identity-based encryption scheme, these two hybrid games are computationally indistinguishable, and we have $\mathsf{Hyb}_t^{(3)} \approx_\delta \mathsf{Hyb}_t^{(4)}$. The reduction is immediate but crucially relies on the fact that $\mathsf{CT}_{i^*,1-f_{i^*}}^{(4)}$ is computed as an encryption for the identity $(i^*, 1-f_{i^*})$ and not for the identity (i^*, f_{i^*}), since the secret key for identity (i^*, f_{i^*}) is revealed to the adversary when giving the functional secret key for f, but not the secret key for identity $(i^*, 1-f_{i^*})$.
- $\mathsf{Hyb}_t^{(4)}$ **to** $\mathsf{Hyb}_t^{(5)}$**:** In this fifth hybrid game, we once again use randomness computed with the PPRF by letting $\mathsf{CT}_{i^*,1-f_{i^*}}^{(5)} \leftarrow \mathsf{IBE.Enc}(\mathsf{MPK}, (i^*, 1-f_{i^*}), \mathsf{Gar.HInpEn}^{(t+1)}(i^*, 1-f_{i^*}; K); \mathsf{PRF.Ev}_S(i^*, 1-f_{i^*}))$. Other values remain the same as before. Once again, assuming $\mathsf{PRF.Ev}$ is a secure puncturable pseudorandom function, we have $\mathsf{Hyb}_t^{(4)} \approx_\delta \mathsf{Hyb}_t^{(5)}$.
- $\mathsf{Hyb}_t^{(5)}$ **to** Hyb_{t+1}**:** By definition, it is clear that the two circuits used in $\mathsf{Hyb}_t^{(5)}$ are functionally equivalent to the two circuits used in $\mathsf{Hyb}_t^{(5)}$, since $\mathsf{Gar.HInpEn}^{(t)}(i, b; K^{(5)}) = \mathsf{Gar.HInpEn}^{(t+1)}(i, b; K)$

for any (i, b) with that $i \neq i^*$. Therefore, under the security of $\mathsf{sxi}\mathcal{O}$, these two hybrid games are computationally indistinguishable, and we have $\mathsf{Hyb}_t^{(5)} \approx_\delta \mathsf{Hyb}_{t+1}$.

This concludes the proof of Theorem 5.2. □

5.3 Weakly Succinct PKFE for Non-Boolean Functions

In this section, we give a transformation from weakly succinct PKFE schemes for boolean functions into ones for *non-boolean* functions.

Theorem 5.3. *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of circuits (with multiple output bits) and let $n(\lambda), s(\lambda), d(\lambda)$ be bounds on their input length, size, and depth (respectively). For any constants $\beta < \gamma < 1$, assuming a β-compressing SXIO, there exists a constant α, such that given any α-compressing weakly succinct PKFE for boolean functions of size $s \cdot \mathrm{polylog}(s)$ and depth $d \cdot \mathrm{polylog}(s)$, and one-way functions, there exists a succinct PKFE for $\mathcal{C}$ with compression factor γ. If all primitives are δ-secure so is the resulting scheme.*

The transformation is essentially the same transformation presented in [LPST16a, Section 4], with the following differences:

- They use XIO rather than SXIO, which results in a PKFE scheme where only the size of ciphertexts is compressed, whereas the time to encrypt may be large. They then make an extra step, based on LWE, to make encryption efficient. Using SXIO directly as we do, allows avoiding this step.
- They start from weakly succinct PKFE for boolean functions where the size of ciphertexts is completely independent of the size s of the function class considered. Due to this, they can start from XIO with any compression factor $\beta < 1$. In our notion of weakly succinct, there is dependence on s^α, for some $\alpha < 1$, and we need to make sure that β and α are appropriately chosen to account for this.
- As stated, their notion of weak succinctness for PKFE does not explicitly scale with the depth of the function class considered. Eventually, they apply their transformation to function classes in $\mathbf{NC}^1$, assuming puncturable PRFs in $\mathbf{NC}^1$ (which exist under LWE). Our succinctness notion allows polynomial dependence on the depth, which should be roughly preserved through the transformation.

For the sake of completeness we describe the transformation in full and then analyze the efficiency aspects. The proof of security is identical to the one in [LPST16a] and is omitted.

Construction. Let $\mathsf{BFE} = (\mathsf{BFE.Setup}, \mathsf{BFE.Key}, \mathsf{BFE.Enc}, \mathsf{BFE.Dec})$ be a weakly succinct single-key public-key FE scheme for boolean functions. Our weakly succinct single-key public-key FE scheme for multi-bit functions is given by the following algorithms.

$\mathsf{Setup}(1^\lambda)$**:**

- Compute $(\mathsf{BFE.MPK}, \mathsf{BFE.MSK}) \leftarrow \mathsf{BFE.Setup}(1^\lambda)$.
- Output $(\mathsf{MPK}, \mathsf{MSK}) = (\mathsf{BFE.MPK}, \mathsf{BFE.MSK})$.

$\mathsf{KeyGen}(\mathsf{MSK}, f)$**:**

- Run $\mathsf{BFE.SK}_{C_f} \leftarrow \mathsf{BFE.Key}(\mathsf{BFE.MSK}, C_f)$ where $C_f : \{0,1\}^n \times \{0,1\}^{\log \ell}$ is a function that takes (x, i) as inputs and outputs the i-th bit (out of ℓ) of $f(x)$ (this is a boolean function).

$\mathsf{Enc}(\mathsf{MPK}, x \in \{0,1\}^n)$**:**

- Sample a puncturable PRF key $S \leftarrow \mathsf{PRF.Gen}(1^\lambda)$,
- Generate $\mathsf{sxi}\mathcal{O}(\mathsf{SE}[x, S, \mathsf{BFE.MPK}])$ where the circuit $\mathsf{SE}[x, S, \mathsf{BFE.MPK}]$ is described in Figure 11.
- Output $\mathsf{CT} = \mathsf{sxi}\mathcal{O}(\mathsf{SE}[x, S, \mathsf{BFE.MPK}])$.

$\mathsf{Dec}(\mathsf{SK}_f, \mathsf{CT}_x)$**:**

- Compute $\mathsf{BFE.CT}_i \leftarrow \mathsf{CT}(i)$, and $y_i \leftarrow \mathsf{BFE.Dec}(\mathsf{BFE.SK}_{C_f}, \mathsf{BFE.CT}_i)$ for all $i \in [\ell]$.
- Output $y = (y_1, \ldots, y_\ell)$.

Encryption circuit $\mathsf{SE}[x, S, \mathsf{BFE.MPK}]$

Hardwired: a message x, a puncturable PRF key S, and a key $\mathsf{BFE.MPK}$.
Input: $i \in [\ell]$.
Padding: the circuit is padded to size pad, determined in the analysis.

1. Compute $r_i \leftarrow \mathsf{PRF.Ev}_S(i)$.
2. Output $\mathsf{BFE.CT}_i \leftarrow \mathsf{BFE.Enc}(\mathsf{BFE.MPK}, (x, i); r_i)$.

Figure 11: Circuit $\mathsf{SE}[x, S, \mathsf{BFE.MPK}]$

Proof of Theorem 5.3. We now prove Theorem 5.3. We focus on succinctness. The proof of security is based on a standard probabilistic IO argument [CLTV15], is identical to the one given by Lin et al. [LPST16a, Theorem 6], and thus omitted. In what follows, let s, d, n, and ℓ be bounds on the size, the depth, input and output lengths of functions in $\mathcal{C}$.

Padding Parameter

In the proof of security [LPST16a], the indistinguishability guarantee of SXIO is invoked for the circuit SE and several hybrid versions of this circuit. All of these circuits compute a puncturable pseudorandom function on an input $i \in [\ell]$ and an encryption using $\mathsf{BFE.Enc}$. Therefore, their size is bounded by $\mathrm{poly}(\lambda, \log(\ell)) + e'$ where e' is the size of the encryption circuit for the class of boolean circuits $\mathcal{C}' = \{C_f \mid f \in \mathcal{C}\}$.

To bound e', we note that the size and depth of circuits in $\mathcal{C}'$ is preserved up to polylogarithmic factors (induced by selecting one of $\ell \leq s$ output bits):

$$s' := |C_f| \leq s \cdot \mathrm{polylog}(s) \qquad \text{and} \qquad d' := \mathsf{depth}(C_f) \leq d \cdot \mathrm{polylog}(s) \ .$$

Thus, by the succinctness guarantee of the PKFE for boolean functions, we know that

$$e' \leq s'^{\alpha} \cdot \mathrm{poly}(d', \lambda, n + \log \ell) \leq s^{\alpha} \cdot \mathrm{poly}(d, \log(s), \lambda, n) \ ,$$

where α is the compression factor of the PKFE. Hence, we obtain:

$$\mathsf{pad} \leq s^{\alpha} \cdot \mathrm{poly}(d, \log(s), \lambda, n) \ .$$

Succinctness

We show that for any $\beta < \gamma < 1$ and an appropriate choice of $\alpha < 1$, the size of the encryption circuit is bounded by $s^{\gamma} \cdot \mathrm{poly}(\lambda, n, d)$.

Let $\mathsf{sxi}\mathcal{O}$ be β-compressing. Then, by the SXIO guarantee, the size e of the obfuscated encryption circuit is bounded by

$$e \leq \ell^{\beta} \cdot \mathrm{poly}(\lambda, |\mathsf{SE}|) \leq s^{\beta} \cdot \mathrm{poly}(\lambda, \mathsf{pad}) \ ,$$

where SE is the obfuscated encryption circuit, and poly is a fixed polynomial. Then, letting $c = O(1)$ represent the polynomial blowup in circuit-size incurred by $\mathsf{sxi}\mathcal{O}$, we can bound

$$e \leq s^{\beta} \cdot s^{c \cdot \alpha} \cdot \mathrm{poly}(d, n, \lambda, \log(s)) \leq s^{\gamma} \cdot \mathrm{poly}(d, n, \lambda) \ ,$$

where the last inequality holds provided that we choose the constant α such that $c\alpha + \beta < \gamma$. (Such a constant indeed exist since $\beta < \gamma$ and c is a constant.) □

5.4 Putting It All Together: From SKFE and PKE to IO

We now show how the results proved in this section come together to give our main result. The main implications are also illustrated in Figure 12.

Theorem 5.4. *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of circuits (with multiple output bits) and let $n(\lambda), s(\lambda), d(\lambda)$ be bounds on their input length, size, and depth (respectively). Then, for any constant $\gamma < 1$, there exists a constant β, such that given any δ-secure, β-compressing SXIO, and δ-secure PKE, there exists $2^d s\delta$-secure, γ-compressing, weakly succinct PKFE for $\mathcal{C}$.*

Proof. Fix some constant γ. The theorem follows from the previous theorems in this section as follows:

- By Theorem 5.3, there exist constants β', α', such that $2^d s\delta$-secure, γ-compressing, weakly succinct PKFE for $\mathcal{C}$, follows from $2^d s\delta$-secure, β'-compressing SXIO and $2^d s\delta$-secure, α'-compressing, weakly succinct PKFE for boolean functions of size $s \cdot \mathrm{polylog}(s)$ and depth $d \cdot \mathrm{polylog}(s)$, and $2^d s\delta$-secure one-way functions.
- By Theorem 5.2, there exist constants β'', α'' such that PKFE as required in the previous item follows from δ-secure, β''-compressing SXIO and δ-secure, α''-compressing IBE, and δ-secure one-way functions.
- By Theorem 5.1, there exists a constant β''' (in fact any $\beta''' < \alpha''$), such that IBE as required in the previous item follows from δ-secure, β'''-compressing SXIO and δ-secure PKE.
- In all of the above δ-secure one-way functions follow from δ-secure PKE.

Setting $\beta := \min(\beta', \beta'', \beta''')$, we derive the theorem. □

Combining the above theorem with the result from Section 3, we obtain the following corollary.

Corollary 5.1. *If there exist (1-input) SKFE for P/poly and PKE, both subexponentially-secure, then there exists IO for P/poly.*

Proof. Fix any constants $\varepsilon, \gamma < 1$. For $\delta(\lambda) = 2^{-\lambda^{\varepsilon}}$.

- By Theorem 2.2 [BV15], for any constant $\gamma < 1$, IO is implied by δ-secure, γ-compressing, weakly succinct PKFE.
- By the above Theorem 5.4, there exists β, such that, PKFE as required in the previous item follows from β-compressing, δ-secure SXIO and δ-secure PKE, when setting the security parameter for the SXIO and PKE to $\tilde{\lambda} = \lambda \cdot (d \log(s))^{2/\varepsilon}$ — this accounts for the $\mathrm{poly}(2^d s)$ security loss, and only incurs $\mathrm{poly}(\lambda, d, \log(s))$ overhead, which satisfies the succinctness requirements of the PKFE.
- By Theorem 3.1, letting $t = \frac{1}{\beta} - 1$, SXIO as required in the previous item, follows from δ-secure t-input SKFE.
- By Theorem 2.1 [BKS16], the required t-input SKFE follows from any 1-input, δ-secure SKFE.

The corollary follows. □

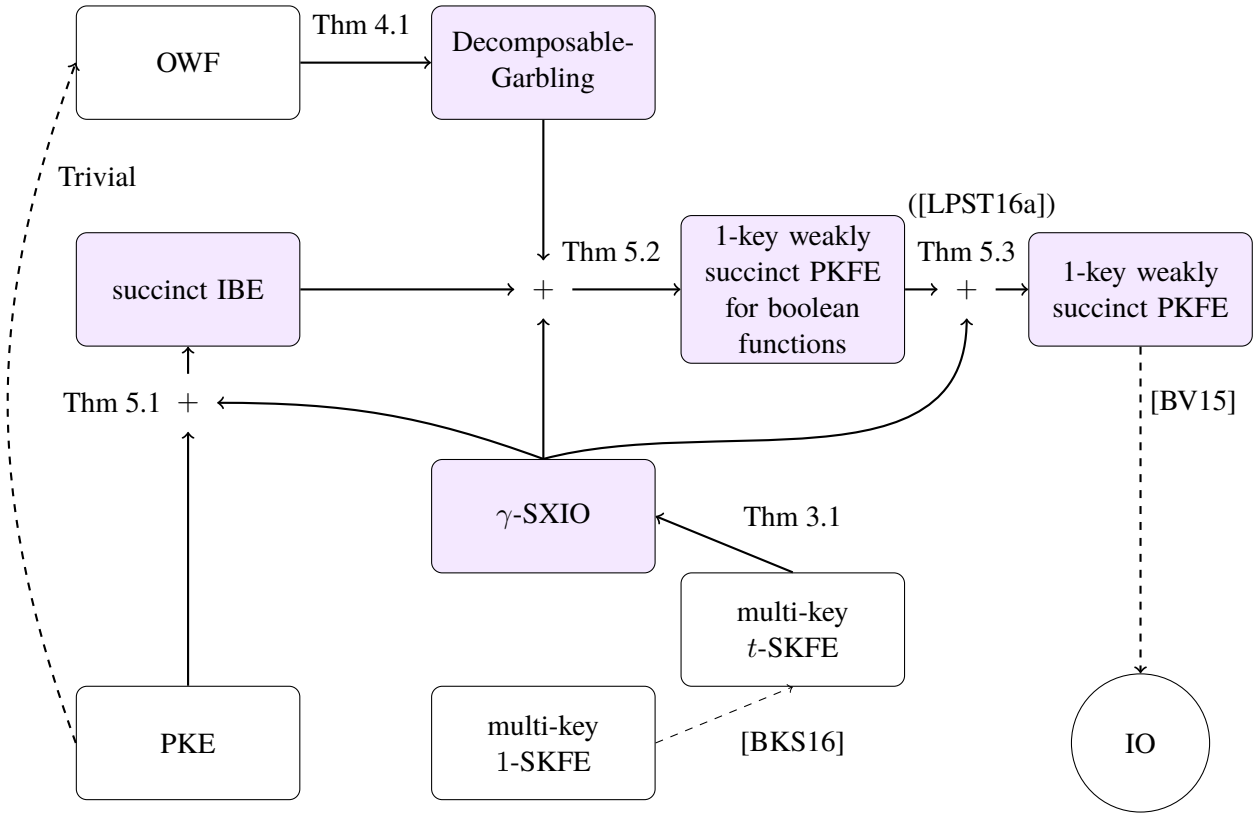

Figure 12: An illustration of the path from PKE and 1-SKFE to IO. Dashed lines denote known results. White backgrounds denote our ingredients or goal. Primitives in rounded rectangles are subexponentially-secure. γ-SXIO denotes SXIO with compression factor γ, which is an arbitrary constant less than 1. We ignore puncturable PRF in this figure since it is implied by OWF.

6 Polynomially-Secure PKE from Secret-Key FE

In this section, we construct public-key encryption (PKE) from secret-key functional encryption (SKFE). Our starting point is Corollary 3.1 that directly follows from Theorems 2.1,3.1. We now show how to construct a PKE scheme from such SXIO.

The construction. Let $\left\{\mathsf{PRG} : \{0,1\}^n \to \{0,1\}^{2n}\right\}_{n\in\mathbb{N}}$ be a length-doubling pseudorandom generator that is $2^{-n/\log\log n}$-secure. Let $\mathsf{sxi}\mathcal{O}$ be a SXIO with compression factor $\gamma(\lambda) = O(1/\log\log\lambda)$ (and $\mathrm{poly}(\lambda)$ security).

The scheme $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is defined as follows:

$\mathsf{KeyGen}(1^\lambda)$**:**

- Sample a PRG seed $s \leftarrow \{0,1\}^{\log\lambda/\gamma(\lambda)}$.
- Output $\mathsf{PK} = \mathsf{PRG}(s)$ and $\mathsf{SK} = s$.

$\mathsf{Enc}(\mathsf{PK}, b)$**:**

- Construct the (witness encryption) circuit $\mathsf{WE}[b, \mathsf{PK}]$ described in Figure 13 that takes $s' \in \{0,1\}^{\log\lambda/\gamma(\lambda)}$ as input and outputs x if and only if $\mathsf{PK} = \mathsf{PRG}(s')$ holds.
- Output $\mathsf{CT} = \mathsf{sxi}\mathcal{O}(\mathsf{WE}[x, \mathsf{PK}])$

$\mathsf{Dec}(\mathsf{SK}, \mathsf{CT})$**:**

- Compute $b' = \mathsf{CT}(\mathsf{SK})$.

Encryption circuit $\mathsf{WE}[b, \mathsf{PK}]$

Hardwired: a message bit b and a public key PK.

Input: a seed $s' \in \{0,1\}^{\log\lambda/\gamma(\lambda)}$.

1. If $\mathsf{PK} = \mathsf{PRG}(s')$, then output b.
2. Otherwise, output $\perp$.

Figure 13: Circuit $\mathsf{WE}[b, \mathsf{PK}]$

Proposition 6.1. *PKE is a (polynomially-secure) public-key encryption scheme.*

Proof sketch. The scheme is clearly correct. We now note that the scheme is efficient; namely, all algorithms run in time $\mathrm{poly}(\lambda)$ in the security parameter λ. This is clearly the case for KeyGen and Dec. As for Enc, by the SXIO compression guarantee, the time required to compute the obfuscation is $2^{\gamma(\lambda)\cdot\log\lambda/\gamma(\lambda)} \cdot \mathrm{poly}(\lambda, |\mathsf{WE}[b, \mathsf{PK}]|) = \mathrm{poly}(\lambda)$. Note that we can apply $\mathsf{sxi}\mathcal{O}$ to $\mathsf{WE}[b, \mathsf{PK}]$ since $|\mathsf{WE}[b, \mathsf{PK}]|$ is bounded by $\mathrm{poly}(\log\lambda)$ and it is less than $2^{O((\log\lambda)^\varepsilon)}$ (This bound comes from Corollary 3.1).

We turn to prove that the scheme is semantically secure. For this purpose, we first consider an alternative encryption scheme where the public-key is generated as a truly random string $\mathsf{PK} \leftarrow \{0,1\}^{2\log\lambda/\gamma(\lambda)}$. By

the security of the PRG, any polysize distinguisher cannot tell apart a real public key from such a fake public key except with advantage

$$2^{-\Omega(n(\lambda)/\log\log n(\lambda))} = 2^{-\Omega\left(\frac{\log\lambda/\gamma(\lambda)}{\log\log(\log\lambda/\gamma(\lambda))}\right)} \leq 2^{-\Omega\left(\frac{-\log\lambda\cdot\log\log\lambda}{\log\log\log\lambda}\right)} = 2^{-\omega(\log\lambda)} \ .$$

We next note that, by a union bound over all possible seeds, a random PK does not have any preimage under PRG, except with probability

$$2^{\log\lambda/\gamma(\lambda)} \cdot 2^{-2\log\lambda/\gamma(\lambda)} = 2^{-\log\lambda/\gamma(\lambda)} \leq 2^{-\omega(\log\lambda)} \ .$$

In this case $\mathsf{WE}[b, \mathsf{PK}]$ is functionally equivalent to a circuit $\mathsf{WE}[\bot, \mathsf{PK}]$ that is independent of b and always outputs $\bot$. Security thus follows from the usual IO guarantee. □

Acknowledgements

We thank Hoeteck Wee and Vinod Vaikuntanathan for valuable discussions.

References

[ABSV15] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. From selective to adaptive security in functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 657–677, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[AGVW13] Shweta Agrawal, Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption: New perspectives and lower bounds. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 500–518, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 308–326, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[AJS15] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Indistinguishability obfuscation from functional encryption for simple functions. Cryptology ePrint Archive, Report 2015/730, 2015. `http://eprint.iacr.org/2015/730`.

[AS15] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 191–209, Berkeley, CA, USA, October 17–20, 2015. IEEE Computer Society Press.

[AS16] Prabhanjan Vijendra Ananth and Amit Sahai. Functional encryption for turing machines. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 125–153, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

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

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany.

[BGJS15] Saikrishna Badrinarayanan, Divya Gupta, Abhishek Jain, and Amit Sahai. Multi-input functional encryption for unbounded arity functions. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015, Part I*, volume 9452 of *Lecture Notes in Computer Science*, pages 27–51, Auckland, New Zealand, November 30 – December 3, 2015. Springer, Heidelberg, Germany.

[BGL+15] Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Sidharth Telang. Succinct randomized encodings and their applications. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th Annual ACM Symposium on Theory of Computing*, pages 439–448, Portland, OR, USA, June 14–17, 2015. ACM Press.

[BHR12] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 12: 19th Conference on Computer and Communications Security*, pages 784–796, Raleigh, NC, USA, October 16–18, 2012. ACM Press.

[BKS16] Zvika Brakerski, Ilan Komargodski, and Gil Segev. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 852–880, 2016. Full version available from https://eprint.iacr.org/2015/158.

[BLR+15] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 563–594, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.

[BO13] Mihir Bellare and Adam O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *CANS 13: 12th International Conference on Cryptology and Network Security*, volume 8257 of *Lecture Notes in Computer Science*, pages 218–234, Paraty, Brazil, November 20–22, 2013. Springer, Heidelberg, Germany.

[BS15] Zvika Brakerski and Gil Segev. Function-private functional encryption in the private-key setting. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 306–324, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany.

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

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 171–190, Berkeley, CA, USA, October 17–20, 2015. IEEE Computer Society Press.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300, Bengalore, India, December 1–5, 2013. Springer, Heidelberg, Germany.

[CH85] Stephen A. Cook and H. James Hoover. A depth-universal circuit. *SIAM J. Comput.*, 14(4):833–839, 1985.

[CLTV15] Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 468–497, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany.

[DIJ+13] Angelo De Caro, Vincenzo Iovino, Abhishek Jain, Adam O'Neill, Omer Paneth, and Giuseppe Persiano. On the achievability of simulation-based security for functional encryption. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 519–535, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[GGG+14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 578–602, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th*

Annual Symposium on Foundations of Computer Science, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press.

[GGHZ16] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Functional encryption without obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 480–511, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th Annual Symposium on Foundations of Computer Science*, pages 464–479, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GJKS15] Vipul Goyal, Abhishek Jain, Venkata Koppula, and Amit Sahai. Functional encryption for randomized functionalities. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 325–351, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 555–564, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GPS15] Sanjam Garg, Omkant Pandey, and Akshayaram Srinivasan. On the exact cryptographic hardness of finding a nash equilibrium. Cryptology ePrint Archive, Report 2015/1078, 2015. `http://eprint.iacr.org/2015/1078`.

[GPSZ16] Sanjam Garg, Omkant Pandey, Akshayaram Srinivasan, and Mark Zhandry. Breaking the sub-exponential barrier in obfustopia. Cryptology ePrint Archive, Report 2016/102, 2016. `http://eprint.iacr.org/2016/102`.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 162–179, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 503–523, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[HJO+15] Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. Cryptology ePrint Archive, Report 2015/1250, 2015. `http://eprint.iacr.org/2015/1250`.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*, pages 134–147. IEEE Computer Society, 1995.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13: 20th Conference on Computer and Communications Security*, pages 669–684, Berlin, Germany, November 4–8, 2013. ACM Press.

[KSY15] Ilan Komargodski, Gil Segev, and Eylon Yogev. Functional encryption for randomized functionalities in the private-key setting from minimal assumptions. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 352–377, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany.

[Lin16] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 28–57. Springer, 2016.

[LPST16a] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 9615 of *Lecture Notes in Computer Science*, pages 447–462, Taipei, Taiwan, March 6–9, 2016. Springer, Heidelberg, Germany.

[LPST16b] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 96–124, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany.

[O'N10] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. `http://eprint.iacr.org/2010/556`.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1984. Springer, Heidelberg, Germany.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10: 17th Conference on Computer and Communications Security*, pages 463–472, Chicago, Illinois, USA, October 4–8, 2010. ACM Press.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 475–484, New York, NY, USA, May 31 – June 3, 2014. ACM Press.

[Wat15] Brent Waters. A punctured programming approach to adaptively secure functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 678–697, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

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