Algorithmica (2017) 79:1353–1373
DOI 10.1007/s00453-017-0276-6

On the Implausibility of Differing-Inputs Obfuscation and Extractable Witness Encryption with Auxiliary Input

Sanjam Garg[1] · **Craig Gentry**[2] · **Shai Halevi**[2] · **Daniel Wichs**[3]

Received: 1 July 2015 / Accepted: 5 January 2017 / Published online: 12 January 2017

Abstract The notion of *differing-inputs obfuscation* (diO) was introduced by Barak et al. (CRYPTO, pp 1–18, 2001). It guarantees that, for any two circuits C_0, C_1 for which it is difficult to come up with an input x on which $C_0(x) \neq C_1(x)$, it should also be difficult to distinguish the obfuscation of C_0 from that of C_1. This is a strengthening of *indistinguishability obfuscation*, where the above is only guaranteed for circuits that agree on all inputs. Two recent works of Ananth et al. (Differing-inputs obfuscation and applications, http://eprint.iacr.org/, 2013) and Boyle et al. (Lindell, pp 52–73, 2014) study the notion of diO in the setting where the attacker is also given some auxiliary information related to the circuits, showing that this notion leads to many interesting applications. In this work, we show that the existence of *general-purpose* diO with *general* auxiliary input has a surprising consequence: it implies that a *specific* circuit C^* with *specific* auxiliary input aux^* cannot be obfuscated in a way that hides some *specific* information. In other words, under the conjecture that such *special-purpose obfuscation* exists, we show that general-purpose diO cannot exist. This conjecture is a falsifiable assumption which we do not know how to break for candidate obfuscation

✉ Daniel Wichs
wichs@ccs.neu.edu

Sanjam Garg
sanjamg@berkeley.edu

Craig Gentry
cbgentry@us.ibm.com

Shai Halevi
shaih@alum.mit.edu

1 UC Berkeley, Berkeley, CA, USA

2 IBM Research, T.J. Watson, Yorktown Heights, NY, USA

3 Department of Computer Science, Northeastern University, Boston, MA, USA

schemes. We also show similar implausibility results for *extractable witness encryption* with auxiliary input and for *"output-only dependent" hardcore bits* for general one-way functions.

Keywords Obfuscation · Witness encryption

1 Introduction

The formal study of program obfuscation was initiated by Hada [16] and Barak et al. [6,7]. Since then there have been many negative and, more recently, also positive results on obfuscation. We briefly survey both directions.

1.1 Negative Results

Hada observed that a super-strong notion of obfuscation, requiring that the obfuscated code does not leak anything beyond what can be learned given black-box oracle access to the underlying function, cannot be met unless the obfuscated function is learnable. Barak et al. define a slightly weaker (but still very strong) notion of "virtual-black-box" (VBB) obfuscation, roughly requiring that the obfuscated circuit does not leak any *predicate of the obfuscated function* beyond what can be learned given black-box oracle access to that function. The main result of Barak et al. shows the impossibility of VBB-obfuscation for general circuits. The impossibility result constructs specific albeit "contrived" functions that cannot be VBB-obfuscated, but also shows that such functions can be embedded into cryptosystems giving "contrived" constructions of cryptosystems (signature schemes, encryption, pseudo-random functions) that cannot be VBB-obfuscated. The main idea behind this result is to construct functions where obfuscated code can be "fed" into the function as an input, causing it to output extra information. Such counterexamples even exist in weak computational classes, (such as any class that simultaneously contains NC0 and a PRF [2]), and therefore we cannot even get general VBB obfuscation for such weak classes. However, this result still leaves open the possibility that many specific functions and most natural cryptosystems can be VBB-obfuscated.

The work of Goldwasser and Kalai [14] considers a notion of VBB obfuscation with auxiliary input and shows that *no* pseudo-random function (even natural ones) can be VBB-obfuscated in the presence of arbitrary auxiliary input. Recent work extends this result to weaker assumptions and more restricted forms of auxiliary input [3]. In all these works, the impossibility result constructs some contrived auxiliary input. In particular, in all these results, the auxiliary input is itself an obfuscated circuit. These negative results leave open two interesting possibilities:

– Perhaps most "natural" functions and "standard-construction" cryptosystems can be VBB-obfuscated in the presence of most "natural" auxiliary inputs, even though there are "contrived" examples of functions and auxiliary inputs that cannot be obfuscated.

– Perhaps some general form of obfuscation, weaker than VBB, is possible for general functions.

1.2 Positive Results

In the face of their impossibility result, Barak et al. proposed two weaker notions of obfuscation that may be achievable for general functions: *indistinguishability obfuscation* and *differing-inputs obfuscation*. Indistinguishability obfuscation says that for any pair of circuits C_0, C_1 that agree on all inputs $C_0(x) = C_1(x)$, it should be hard to distinguish the obfuscation of C_0 from that of C_1. The work of Garg et al. [12] gave the first candidate for general-purpose indistinguishability obfuscation based on multilinear maps, and several applications of this primitive. The work of Sahai and Waters [22] showed how to use iO in applications and many works relying on iO have appeared since then. Indistinguishability obfuscation is also called the "best possible" obfuscation since anything that any obfuscator can hide, an indistinguishability obfuscator (with sufficient padding) is guaranteed to hide as well. Therefore, one can conjecture that this obfuscator satisfies stronger properties.

The works of [8,10] also give constructions of obfuscators satisfying even the stronger VBB property in the "generic multilinear map" model. This result is difficult to interpret since we do have "non-generic" attacks given by the prior negative results.

Reconciling the positive and negative results suggests the following interpretation: when it comes to *general* functionalities and *general* auxiliary input, one can cook-up clever "contrived" counterexamples that allow for "non-generic" attacks, and therefore one must settle for weak notions of obfuscation, like indistinguishability obfuscation. On the other hand, when it comes to most *specific* functionalities with *specific* auxiliary input distributions, even strong notions of VBB obfuscation may be achievable. In particular, if we fix a specific function and auxiliary input, unless there is some "obvious" attack where the code of the function can be meaningfully used as an input (either to the function itself or to some other function given by the auxiliary input) it may be reasonable to assume that VBB obfuscation is possible in this specific case.

1.3 Differing-Inputs Obfuscation

Despite its usefulness in many recent applications, indistinguishability obfuscation is often difficult to use as a general assumption. The work of Barak et al. also proposed a stronger notion called differing-inputs obfuscation (diO). In particular, this notion says that for any distribution on circuits (C_0, C_1), if it is hard to find an input x such that $C_0(x) \neq C_1(x)$, then it should also be hard to distinguish the obfuscation of C_0 from that of C_1. The recent work of Ananth et al. [1] and Boyle et al. [4] extend this notion to the setting of auxiliary input, where the attacker is given (C_0, C_1, aux) and, if it is hard to use this information to find an input x on which $C_0(x) \neq C_1(x)$, then it should also be hard to use this information to distinguish the obfuscation of C_0 and C_1. These works give several interesting applications of this notion, including the ability to obfuscate Turing Machine without the cost of converting them into a circuit.

1.4 Our Result

As our main result, we show that the existence of general-purpose differing-inputs obfuscation (diO) with auxiliary-input leads to a surprising consequence: it would show the impossibility of obfuscating a *specific* circuit C^* with *specific* auxiliary input aux^* in a way that hides some *specific* information. In particular, we put forth a "counter-conjecture" that such "special purpose" circuit-obfuscators exist and, under this conjecture, general-purpose diO with auxiliary input does not exist. Moreover, under the same conjecture, we also show that extractable witness encryption (with auxiliary input) does not exist. We also consider a restricted scenario of "bounded-length auxiliary input" where the length of the auxiliary input is bounded a-priori, and the diO obfuscator is given the length bound. We show that a variant of our 'special purpose obfuscation" conjecture (using an obfuscator for Turing Machines rather than circuits) rules this out as well. Lastly, in Appendix 7, we also show that this variant of our conjecture rules out "output-only dependent" hardcore bits for general one-way functions, where the value of the hardcore bit is completely determined by the output of the function. Such hardcore bits were recently constructed using diO with bounded-length auxiliary input by Bellare and Tessaro [11].

1.5 What to Believe?

Our "special-purpose obfuscation" conjecture is not known to be implied by differing-inputs obfuscation itself, and hence we do not get unconditional *impossibility* results. In particular, our main result leaves us with the following two opposing possibilities: (I) general-purpose diO with auxiliary input exists, (II) our special-purpose obfuscation assumption holds. We cannot objectively say which one of these is false. However, (II) is a falsifiable assumption in the formal sense of [19], where an efficient challenger can check if an attack is valid. Using the obfuscator of [12] (or [8,10]), we currently do not know of any attacks on (II). In other words, the validity of (I) would imply the existence of an efficient algorithm whose correctness would be easy to verify, but we do not have any candidate for this algorithm. On the other hand, (I) itself is not stated as a falsifiable assumption, and there is no direct way to verify an attack against it via an efficient challenger. Indeed, we present an efficient attack that contradicts the security of (I), but there is no direct way to check if our attack is "valid" since doing so requires proving (II). Therefore, we view our result as presenting a significant challenge to the plausibility of general-purpose diO with auxiliary input. See further discussion on our conjecture in Sect. 4.

1.6 Consequences of Our Result

Assuming that our "special-purpose obfuscation" conjecture holds, we have ruled out the existence of general-purpose diO with auxiliary input. However, it may still be reasonable to assume that diO security and even VBB security with auxiliary input can hold in concrete cases. Many of the applications of diO in the works [1,4] and follow-up works remain plausible and only rely on diO security with some concrete auxiliary

input, which is unlikely to contain our "counterexample".[1] Nevertheless, to avoid our implausibility result, one would have to carefully pose a new diO assumption for the specific auxiliary input required in each new application and convincingly argue that this assumption is plausible even if the general one is not. Although this approach may be a sound, it runs counter to our goal of constructing a wide variety of cryptosystems from a few general (and plausible) assumptions.

1.7 Related Work

Our technique follows the approach of similar results [3,5,9,14], all of which use one form of obfuscation to derive counterexamples for other forms of obfuscation and/or various extractability assumptions. In particular, the results of [3,14] show that existence of iO implies the impossibility of VBB obfuscation of natural functionalities with (unnatural) auxiliary input, whereas [5,9] show that existence of iO/diO implies impossibility of extractable functions and related extractability primitives.

1.8 Our Technique

The main idea of our technique is to create a contrived "auxiliary input" aux which is itself an obfuscated circuit. In particular, aux allows the attacker to distinguish any obfuscations of some carefully designed C_0, C_1, without gaining the ability to find an input on which they differ. The "special purpose" assumption is needed to guarantee that aux does not "leak" an input x on which $C_0(x) \neq C_1(x)$.

In more detail, the circuits C_b ($b \in \{0, 1\}$) have a verification key vk of a signature scheme hard-coded in them. If they get an input $x = (m, \sigma)$ consisting of a valid message/signature pair, they output the bit b, else they just both output 0. Finding an input x on which $C_0(x) \neq C_1(x)$ requires finding a valid message/signature pair (which is hard to do even given vk). We set the auxiliary input aux to be a "special-purpose" obfuscation of a circuit C^* that has the signing key sk hard-coded and is defined as follows: given as input any circuit C with 1-bit output, it outputs $C(m, \sigma)$ where $m = H(C)$ is a collision-resistant hash of C and σ is a signature of m under sk. It is easy to use (an obfuscation of) C^* to distinguish C_0 and C_1 just by feeding them to C^*. However, given black-box access to C^*, we show that it is impossible to recover any message/signature pair and therefore any input x on which $C_0(x) \neq C_1(x)$. Intuitively, each call to C^* leaks one bit of information on a fresh message/signature pair, which is not enough to recover any such pair in full. We therefore put forth the conjecture that there exists a "special-purpose" method of obfuscating C^*, that does not allow the attacker to learn any message/signature pair. Under this special-purpose obfuscation assumption, the auxiliary input aux allows us to distinguish any obfuscation of C_0, C_1 but does not allow us to find any input x on which $C_0(x) \neq C_1(x)$.

[1] The notable exceptions are "extractable/functional witness encryption" [4] and "output-only dependent hardcore bits for any one-way function" [11] where the auxiliary input is external and is not fixed by the construction. Our counterexamples show that these notions are "implausible" in their general form.

1.9 Follow-Up Work

Following our work, the work of Ishai, Pandey and Sahai [17] proposes a relaxed definition of diO called *public-coin* diO. This definition considers distributions on circuits (C_0, C_1) that are generated using some random coins r, and such that it is difficult to find an input x on which $C_0(x) \neq C_1(x)$ *even given* r. For such circuits, it insists that the obfuscation of C_0 and C_1 is indistinguishable even given the coins r. The notion of public-coin diO completely avoids our counter-example. In particular, it prevents us from embedding some hidden information in the auxiliary input or in the definition of the circuits C_0, C_1 since the adversary sees all the information needed to generate these. The work of [17] uses this notion of public-coin diO to construct succinct obfuscation and functional encryption schemes for Turing Machines.

2 Preliminaries and Definitions

2.1 Notation

We let λ denote the security parameter throughout the paper. We use the notation $C[\mathsf{prm}]$ to denote a circuit that depends on a parameter prm. The parameter can be an arbitrary string, and we think of prm as being "hard wired" in the description of the corresponding circuit. The input to a circuit is specified inside parenthesis, so $C[\mathsf{prm}](x)$ describes the computation of the circuit $C[\mathsf{prm}]$ (whose definition depends on prm) on the input x.

2.2 Differing-Inputs Obfuscation

Our definition of differing-inputs obfuscation (diO) with auxiliary input follows that of Ananth et al. [1], which is also equivalent to that of Boyle et al. [4]. First, we define the notion of "differing-inputs" circuits.

Definition 1 A circuit family $\mathcal{C}$ with a sampler $(C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)$ which samples $C_0, C_1 \in \mathcal{C}$ is said to be a *differing-inputs* family if for all PPT attackers $\mathcal{A}$ there is a negligible function ε such that:

$$\Pr[C_0(x) \neq C_1(x) \ : \ (C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda), x \leftarrow \mathcal{A}(1^\lambda, C_0, C_1, \mathsf{aux})] \leq \varepsilon(\lambda).$$

Definition 2 A PPT algorithm $\mathcal{O}$ is a *differing-inputs obfuscator* (diO) for a differing-inputs family $\mathcal{C}$, Sam if the following holds:

– Correctness: For all $\lambda \in \mathbb{N}$, $C \in \mathcal{C}$ and all inputs x, we have:

$$\Pr[C'(x) = C(x) \mid C' \leftarrow \mathcal{O}(1^\lambda, C)] = 1.$$

– Security: For all PPT distinguishers $\mathcal{D}$, there is a negligible function ε such that:

$$|\Pr[\mathcal{D}(1^\lambda, \mathcal{O}(1^\lambda, C_0), \mathsf{aux}) = 1] - \Pr[\mathcal{D}(1^\lambda, \mathcal{O}(1^\lambda, C_1), \mathsf{aux}) = 1]| \leq \varepsilon(\lambda)$$

where $(C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)$.

A PPT algorithm $\mathcal{O}$ is a *general-purpose differing-inputs obfuscator* if the above holds for all differing-inputs families $\mathcal{C}$, Sam.

The works of [1,4] put forth the conjecture that general-purpose diO exists, and that the obfuscator of [12] is a good candidate.

2.3 Extractable Witness Encryption

Next we define the "extractable" variant of witness encryption following Goldwasser et al. [15]. The notion of witness encryption was first defined and realized by Garg et al. [13]. Goldwasser et al. [15] conjecture that the same construction can be assumed to be extractable with auxiliary input. For simplicity, we assume that the message space is 1 bit. Next we present these definitions formally (following [13,15], but making the definitions even weaker by assuming the auxiliary input comes from an efficiently sampleable distribution and allowing the extractor to depend on this distribution).

Definition 3 A *witness encryption* scheme for an ***NP*** language L (with corresponding witness relation R) consists of the following two polynomial-time algorithms:

Encryption. The algorithm $\mathsf{Enc}(1^\lambda, x, b)$ takes as input a security parameter 1^λ, an unbounded-length string x, and a message $b \in \{0, 1\}$ and outputs a ciphertext c.
Decryption. The algorithm $\mathsf{Dec}(\mathsf{c}, w)$ takes as input a ciphertext c and an unbounded-length string w, and outputs a message b or the symbol $\perp$.

These algorithms satisfy the following two conditions:

– **Correctness.** For any security parameter λ, for any $b \in \{0, 1\}$, and for any $x \in L$ such that $R(x, w)$ holds, we have that

$$\Pr\left[\mathsf{Dec}\big(\mathsf{Enc}(1^\lambda, x, b), w\big) = b\right] = 1.$$

– **Extractable Security.** For any PPT adversary A, polynomial-time sampler $(x, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)$ and for any polynomial $q(\cdot)$, there exists a PPT extractor E and a polynomial $p(\cdot)$, such that:

$$\Pr\left[A(1^\lambda, x, \mathsf{c}, \mathsf{aux}) = b \;\middle|\; \begin{array}{c} b \leftarrow \{0,1\}, (x, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda), \\ \mathsf{c} \leftarrow \mathsf{Enc}(1^\lambda, x, b) \end{array}\right] \geq \frac{1}{2} + \frac{1}{q(\lambda)}$$

$$\Rightarrow \Pr[E(1^\lambda, x, \mathsf{aux}) = w \text{ s.t. } (x, w) \in R_L \;:\; (x, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)] \geq \frac{1}{p(\lambda)}.$$

3 The Counterexample to DiO and the Counter-Conjecture

We construct a family $(\mathcal{C}, \mathsf{Sam})$ which we show to be unobfuscatable with respect to differing-inputs obfuscation. However, to show that this family $(\mathcal{C}, \mathsf{Sam})$ is a differing-inputs family, we will in turn need to rely on a new "special purpose obfuscation" conjecture.

Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver})$ be a signature scheme with signature size $\ell_{sig}(\lambda)$ and a deterministic signing algorithm.[2] Let $\mathcal{H} = \{\mathcal{H}_\lambda\}$ be a collision-resistant hash function (CRHF) family with output size $\ell_{hash}(\lambda)$. Define the circuit family $\mathcal{C}$ consisting of circuits $C[b, \mathsf{vk}] \in \mathcal{C}$ defined as follows:

$\underline{C[b, \mathsf{vk}](m, \sigma)}$ // Hard-coded values: $b \in \{0, 1\}$, vk verification key
// Input: $m \in \{0, 1\}^{\ell_{hash}(\lambda)}$, $\sigma \in \{0, 1\}^{\ell_{sig}(\lambda)}$
– Check $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$. If not output 0 else output b.

Let $\ell_{circ}(\lambda)$ be the maximal size of the circuit $C[b, \mathsf{vk}]$ when $b \in \{0, 1\}$ and $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

Our counterexample to diO will consist of setting $C_0 = C[0, \mathsf{vk}]$ and $C_1 = C[1, \mathsf{vk}]$. Finding an input on which $C_0(x) \neq C_1(x)$ is equivalent to finding any valid message/signature pair $x = (m, \sigma)$ which is hard given only the description of C_0, C_1 (which includes vk). However, we will provide an additional auxiliary input aux which makes it easy to distinguish any (bounded size) obfuscation of C_0 from that of C_1. We will need to argue that aux does not leak any valid message/signature pair, which will require a new assumption.

Let $\ell^* = \ell^*(\lambda)$ be a length parameter (which will later be set to correspond to the size of a candidate obfuscation of the circuits $C[b, \mathsf{vk}]$). Define the circuit family $\mathcal{C}$ consisting of circuits $C^*[H, \mathsf{sk}] \in \mathcal{C}$ with input-length ℓ^* and 1-bit output as follows:

$\underline{C^*[H, \mathsf{sk}](C)}$ // Hard-coded values: $H \in \mathcal{H}$, sk signing key
// Input: C : a circuit of size $|C| = \ell^*$ with 1-bit output.
– Compute $m = H(C)$, $\sigma = \mathsf{Sig}_{\mathsf{sk}}(m)$.
– Output the bit $C(m, \sigma)$.

Let $\mathsf{sp}\mathcal{O}$ be a "special purpose" obfuscator that satisfies correctness and whose security properties we will define shortly. We define the circuit sampler $\mathsf{Sam}_{\ell^*}(1^\lambda)$, parameterized by some polynomial $\ell^*(\cdot)$, as follows:

– Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $H \leftarrow \mathcal{H}_\lambda$.
– Set $C_0 = C[0, \mathsf{vk}], C_1 = C[1, \mathsf{vk}] \in \mathcal{C}$.
– Set $C^* = C^*[H, \mathsf{sk}] \in \mathcal{C}$ to be a circuit with input-length $\ell^* = \ell^*(\lambda)$ and set $\mathsf{aux} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, C^*)$.
– Output C_0, C_1, aux.

It is easy to see that the circuit family $\mathcal{C}, \mathsf{Sam}_{\ell^*}$ is unobfuscatable since aux allows one to easily distinguish any obfuscations of C_0 and C_1 that have circuit-size at most

[2] Any signature scheme can be converted into one with a deterministic signing algorithm by replacing the random coins with a PRF of the message.

ℓ^*. For any candidate obfuscator $\mathcal{O}$, we can choose ℓ^* sufficiently large to ensure that $\mathcal{O}$ fails.

Lemma 1 *Fix any signature/hash schemes $\mathcal{S}$, $\mathcal{H}$ which define the class of circuits $\mathcal{C}$, and let $\mathsf{sp}\mathcal{O}$ be any "special-purpose obfuscator" satisfying correctness. Then for any candidate diO obfuscator $\mathcal{O}$ there is a polynomial $\ell^*(\lambda)$ such that the obfuscations of the family $(\mathcal{C}, \mathsf{Sam}_{\ell^*})$ under $\mathcal{O}$ are easily distinguishable: there is a polynomial-time distinguisher $\mathcal{D}$ such that*

$$|\Pr[\mathcal{D}(1^\lambda, \mathcal{O}(1^\lambda, C_0), \mathsf{aux}) = 1] - \Pr[\mathcal{D}(1^\lambda, \mathcal{O}(1^\lambda, C_1), \mathsf{aux}) = 1]| = 1$$

where $(C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}_{\ell^*}(1^\lambda)$.

Proof Let $\ell_{circ}(\lambda)$ be the maximal size of the circuit $C[b, \mathsf{vk}] \in \mathcal{C}$ when $b \in \{0, 1\}$ and $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$. Set $\ell^*(\lambda)$ be the maximal size of $\mathcal{O}(1^\lambda, C)$ for any $C \in \mathcal{C}$ of size $|C| = \ell_{circ}(\lambda)$. The distinguisher $\mathcal{D}(1^\lambda, \widetilde{C}, \mathsf{aux})$ simply interprets aux as a circuit and outputs $\mathsf{aux}(\widetilde{C})$. It is easy to see that, if $\widetilde{C} = \mathcal{O}(1^\lambda, C_b)$, then $\mathsf{aux}(\widetilde{C}) = b$ and therefore the distinguishing advantage is 1. Also the size of $\widetilde{C}$ is at most $\ell^*(\lambda)$ and hence it can be used as an input to aux.

□

To get a counterexample to the existence of general-purpose differing-inputs obfuscation, we need to show that, for *some* signature scheme $\mathcal{S}$, CRHF $\mathcal{H}$ and obfuscator $\mathsf{sp}\mathcal{O}$, the family $(\mathcal{C}, \mathsf{Sam}_{\ell^*})$ is a differing-inputs family for any ℓ^*. Notice that finding an input $x = (m, \sigma)$ on which $C_0(x) \neq C_1(x)$ is the same as finding a valid message/signature pair. Therefore, the above reduces to the following conjecture which says that, given the obfuscation of the "breaker" circuit C^* it is difficult to produce any valid message/signature pair.

Conjecture 1 (Special-Purpose Obfuscation) There exists a signature scheme $\mathcal{S}$, CRHF $\mathcal{H}$ and an obfuscator $\mathsf{sp}\mathcal{O}$ such that the following hods. For any PPT attacker $\mathcal{A}$ and any polynomial $\ell^*(\cdot)$ there is a negligible $\varepsilon(\lambda)$ such that:

$$\Pr\left[\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1 \;\middle|\; \begin{array}{c}(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda), H \leftarrow \mathcal{H}_\lambda \\ \tilde{C} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, C^*[H, \mathsf{sk}]) \\ (m, \sigma) \leftarrow \mathcal{A}(1^\lambda, \mathsf{vk}, \tilde{C})\end{array}\right] \leq \varepsilon(\lambda)$$

where we take the circuit $C^*[H, \mathsf{sk}] \in \mathcal{C}$ with input-size $\ell^*(\lambda)$ as defined above.

If we fix some specific choice of schemes $\mathcal{S}$, $\mathcal{H}$, $\mathsf{sp}\mathcal{O}$ (e.g., a standard construction of signatures and hash functions and the obfuscation scheme of [12]) then the above becomes a falsifiable assumption. We can efficiently test if an attacker $\mathcal{A}$ breaks the scheme. We now show that, under the above conjecture, the circuit family $(\mathcal{C}, \mathsf{Sam})$ defined above is a differing-inputs family.

Lemma 2 *For any signature scheme $\mathcal{S}$, CRHF $\mathcal{H}$ and an obfuscator $\mathsf{sp}\mathcal{O}$ satisfying Conjecture 1, for any polynomial ℓ^*, the circuit family $(\mathcal{C}, \mathsf{Sam}_{\ell^*})$ defined above is a differing-inputs family.*

Proof Assume there is a PPT attacker $\mathcal{B}$ such that:

$$\Pr[C_0(x) \neq C_1(x) \ : \ (C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}_{\ell^*}(1^\lambda), x \leftarrow \mathcal{B}(1^\lambda, C_0, C_1, \mathsf{aux})] = \varepsilon(\lambda).$$

Since $C_0(x) \neq C_1(x)$ means that $x = (m, \sigma)$ such that $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$, we get

$$\Pr[\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1 : (C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}_{\ell^*}(1^\lambda), x \leftarrow \mathcal{B}(1^\lambda, C_0, C_1, \mathsf{aux})] = \varepsilon(\lambda).$$

Define the attacker $\mathcal{A}(1^\lambda, \mathsf{vk}, \widetilde{C})$ that constructs $C_0 = C[0, \mathsf{vk}], C_1 = C[1, \mathsf{vk}]$, $\mathsf{aux} = \widetilde{C}$ and calls $\mathcal{B}(1^\lambda, C_0, C_1, \mathsf{aux})$. Then

$$\Pr\left[\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1 \ \middle| \ \begin{array}{c}(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda), H \leftarrow \mathcal{H}_\lambda \\ (m, \sigma) \leftarrow \mathcal{A}(1^\lambda, \mathsf{vk}, \mathsf{sp}\mathcal{O}(1^\lambda, C^*[H, \mathsf{sk}]))\end{array}\right] = \varepsilon(\lambda)$$

where the input size of $C^*[H, \mathsf{sk}]$ is $\ell^*(\lambda)$. Therefore, by the conjecture, we must have $\varepsilon(\lambda)$ is negligible, which means that the $(\mathcal{C}, \mathsf{Sam}_{\ell^*})$ is differing-inputs family.

□

Combining Lemma 2 and Lemma 1 we get the main theorem.

Theorem 1 *Under the special-purpose obfuscation conjecture (Conjecture 1), general-purpose differing-inputs obfuscators do not exist.*

4 Substantiating the Special-Purpose Obfuscation Conjecture

We now attempt to substantiate the special-purpose obfuscation conjecture (Conjecture 1). As a first step, we show that black-box access to the circuit $C^*[H, \mathsf{sk}]$ cannot be used to leak a message/signature pair. Intuitively, each query C allows the attacker to learn 1 bit of leakage $C(m, \sigma)$ on a signature of the message $m = H(C)$. Assuming the attacker cannot break collision-resistance, he cannot get more than 1 bit of leakage on any single signature. Generically, seeing 1 bit of leakage on signatures of many different messages does not allow an attacker to come up with any valid message, signature pair. We formalize this via the following Lemma.

Lemma 3 *For any signature scheme $\mathcal{S}$ and CRHF $\mathcal{H}$, and parameter $\ell^*(\lambda)$, for any PPT attacker $\mathcal{A}$ there is a negligible $\varepsilon(\cdot)$ such that:*

$$\Pr\left[\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1 \ \middle| \ \begin{array}{c}(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda), H \leftarrow \mathcal{H}_\lambda \\ (m, \sigma) \leftarrow \mathcal{A}^{C^*[H, \mathsf{sk}](\cdot)}(1^\lambda, \mathsf{vk}, H)\end{array}\right] \leq \varepsilon(\lambda)$$

where $C^*[H, \mathsf{sk}] \in \mathcal{C}$ is defined above and has input size $\ell^*(\lambda)$.

Proof Fix some signature scheme $\mathcal{S}$ and CRHF $\mathcal{H}$ and PPT attacker $\mathcal{A}$. Let $q = q(\lambda)$ be an upper bound on the number of queries that $\mathcal{A}$ makes to C^* and let $\varepsilon(\lambda)$ denote the success probability of $\mathcal{A}$. We define an attacker $\mathcal{B}$ on the EU-CMA (existential unforgeability against chosen message attack) signature security of $\mathcal{S}$ as follows:

- $\mathcal{B}$ guesses an index $i \leftarrow [q]$ and a bit $b \leftarrow \{0, 1\}$ uniformly at random.
- $\mathcal{B}$ gets vk from its challenger and samples $H \leftarrow \mathcal{H}_\lambda$. It runs $\mathcal{A}(1^\lambda, \mathsf{vk}, H)$.
  - Whenever $\mathcal{A}$ makes any query other than the ith query to C^* with some input C, the attacker $\mathcal{B}$ computes $m = H(C)$ uses its signing oracle to compute $\sigma = \mathsf{Sig}_{\mathsf{sk}}(m)$. It then output $C(m, \sigma)$.
  - When $\mathcal{A}$ makes the ith query C_i to C^*, the attacker $\mathcal{B}$ simply responds with the bit b it chose randomly.
- At the end $\mathcal{B}$ outputs the value (m, σ) that $\mathcal{A}$ outputs.

Define the events:

- $\mathsf{Win}_{\mathcal{B}}$ is the event that $\mathcal{B}$ wins the EU-CMA signature game.
- Ver is the event that $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$.
- Col is the event that, during the course of the game, the attacker $\mathcal{A}$ submits two different circuits C, C' to its oracle such that $H(C) = H(C')$.
- Good_1 is the event that, if $\mathcal{A}$ outputs (m, σ), then no query C_j to C^* resulted in $H(C_j) = m$ other than possibly the ith query.
- Good_2 is the event that, if the ith query is C_i, and we set $m = H(C_i)$, $\sigma = \mathsf{Sig}_{\mathsf{sk}}(m)$, then $C_i(m, \sigma) = b$.

Then we have

$$\begin{aligned}
\Pr[\mathsf{Win}_{\mathcal{B}}] &\geq \Pr[\mathsf{Ver} \wedge \mathsf{Good}_1] \geq \Pr[\mathsf{Ver} \wedge \mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \neg\mathsf{Col}] \\
&\geq \Pr[\mathsf{Good}_1 \mid \mathsf{Ver} \wedge \mathsf{Good}_2 \wedge \neg\mathsf{Col}] \Pr[\mathsf{Ver} \wedge \mathsf{Good}_2 \wedge \neg\mathsf{Col}] \\
&\geq \frac{1}{q} \Pr[\mathsf{Ver} \wedge \mathsf{Good}_2 \wedge \neg\mathsf{Col}] \qquad (1) \\
&\geq \frac{1}{q} \Pr[\mathsf{Good}_2] \Pr[\mathsf{Ver} \mid \mathsf{Good}_2] - \Pr[\mathsf{Col}] \\
&\geq \frac{1}{2q} \varepsilon(\lambda) - \delta_{col}(\lambda) \qquad (2)
\end{aligned}$$

where $\delta_{col}(\lambda) := \Pr[\mathsf{Col}]$ is negligible by the security of the CRHF. Equation (1) follows since, even if we condition on $\neg\mathsf{Col}$ and all other randomness in the game other than the choice of i, the attacker $\mathcal{A}$ made at most 1 query C_j such that $H(C_j) = m$ and therefore with probability $1/q$ over only the choice of i we have $i = j$. Equation (2) follows since the probability of Good_2 is $\frac{1}{2}$ only over the choice of b, and conditioned on Good_2, the attacker $\mathcal{B}$ perfectly simulates the obfuscation game for $\mathcal{A}$.

Since, by the security of the signature scheme, we must have $\Pr[\mathsf{Win}_{\mathcal{B}}]$ is negligible, this must also mean that $\varepsilon(\lambda)$ is negligible, which concludes the proof. □

4.1 Further Informal Discussion

We stress that to rule out general-purpose diO we do *not* need the conjecture above to hold for *all* hash functions and signatures. [3] Rather, it is enough that it holds for *some* hash function and signature scheme (such as e.g., RSA PKCS #1 v1.5).

[3] Indeed, we suspect that one should be able to come up with some "unnatural" signature and hash function for which it does not hold (following similar counter-examples from [5,7,14]).

Let's consider attempts at attacking the conjecture, and give highly informal arguments for why they seem to fail. To do so, let's fix some "standard-construction" hash function and signature scheme such as RSA PKCS #1 v1.5, in which case we are also fixing the auxiliary information $\mathsf{aux} = \mathsf{vk}$. As mentioned, all of the prior obfuscation impossibility results have the same general structure which, applied to our problem, would require us to either: (i) use the obfuscated-code $\mathsf{sp}\mathcal{O}(C^*)$ to design a special input on which C^* outputs additional information [7], or (ii) interpret the auxiliary information $\mathsf{aux} = \mathsf{vk}$ as code which outputs some information when given $\mathsf{sp}\mathcal{O}(C^*)$ as an input [3,14]. Since in our case vk is just the verification of a standard scheme (e.g. RSA PKCS #1 v1.5), there does not seem to be much hope in approach (ii). On the other hand, there do not seem to be any special inputs on which C^* acts in any "special way" so as to exploit approach (i). The fact that the input to C^* is itself interpreted as a circuit C and executed by C^* should give us some pause. After all, we can make C depend on $\mathsf{sp}\mathcal{O}(C^*)$. But such inputs would not be treated in any kind of special way by C^*: they would still only allow the attacker to leak one bit of information $C(m, \sigma)$ on an honestly generated message/signature pair.

Finally, we note that a recent result that relates iO to a limited form of diO has no bearing on our counterexample: Boyle et al. [4] showed that differing-inputs obfuscation is already implied by indistinguishability obfuscation, in the special case where the two circuits C_0, C_1 only differ on polynomially many inputs. In our counterexample, the circuits C_0, C_1 differ on all valid message/signature pairs where the message-domain is super-polynomial. Therefore, we do not get any negative results for indistinguishability obfuscation.

5 Bounded-Length Auxiliary Input

Our counterexample shows that, under our special-purpose obfuscation conjecture, there is no general-purpose diO scheme that works with any auxiliary input. In particular, we constructed family (C_0, C_1, aux) where the definition of aux relies on some parameter ℓ^* such that any obfuscations of C_0 and C_1 having size at most ℓ^* are always distinguishable given aux. We can make the parameter ℓ^* arbitrary large at the expense of making the auxiliary input aux correspondingly large. This leaves open the possibility of a diO scheme that is secure for all auxiliary input of some arbitrary but a-priori bounded size. We define this as follows:

Definition 4 We define a general-purpose *diO obfuscator with bounded-length auxiliary input* analogously to Definition1 but with the following changes:

- The syntax of the obfuscator $\mathcal{O}(1^\lambda, 1^{\ell_{aux}(\lambda)}, C)$ now takes an additional parameter $\ell_{aux}(\lambda)$.
- We require that for all polynomial $\ell_{aux}(\lambda)$ security holds for differing-inputs families $(\mathcal{C}, \mathsf{Sam})$ where the size of aux in $(C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)$ is bounded by $\ell_{aux}(\lambda)$.

Our previously described counterexample does not rule out this definition. In particular, the auxiliary input aux in our counterexample is an obfuscated circuit that takes as input an obfuscation of C_b. If the obfuscation of C_b can depend on (and exceed) the

size of aux, then this would not work. However, we can rule out this weaker notion of diO for bounded-length auxiliary input if we additionally assume that we have a special-purpose obfuscator $\mathsf{sp}\mathcal{O}$ which works directly on Turing Machines rather than circuits. In particular, a Turing Machine special-purpose obfuscator $\mathsf{sp}\mathcal{O}(1^\lambda, M)$ takes as input a Turing Machine M and outputs an obfuscated Turing Machine $\tilde{M}$ where $\tilde{M}$ can be evaluated on arbitrary-length inputs and produces the same output as M.

The Counterexample. Fix a signature scheme $\mathcal{S}$ and hash function family $\mathcal{H}$ as before, and define the circuit family $\mathcal{C}$ consisting of circuits $C[b, \mathsf{vk}]$ as before. We define the "breaker" Turing Machine $M^*[H, \mathsf{sk}]$ which has H and sk hard-coded in its description analogously to the way we defined the "breaker" circuit $C^*[H, \mathsf{sk}]$, as follows:

$M^*[H, \mathsf{sk}](C)$ // Hard-coded values: $H \in \mathcal{H}_\lambda$, sk signing key
// Input: C circuit with 1-bit output and arbitrary size.
- Compute $m = H(C)$, $\sigma = \mathsf{Sig}_{\mathsf{sk}}(m)$.
- Output $C(m, \sigma)$.

Notice that, unlike before, we no longer have any parameter ℓ^* that would fix the maximal input length of the input circuit C given to $M^*[H, \mathsf{sk}]$.

Let $\mathsf{sp}\mathcal{O}$ be a Turing-Machine obfuscator that satisfies correctness. We define the circuit sampler $\mathsf{Sam}_{TM}(1^\lambda)$ as follows:

- Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $H \leftarrow \mathcal{H}_\lambda$.
- Set $C_0 = C[0, \mathsf{vk}]$, $C_1 = C[1, \mathsf{vk}] \in \mathcal{C}$.
- Set $M^* = M^*[H, \mathsf{sk}]$ and $\mathsf{aux} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, M^*)$.
- Output C_0, C_1, aux.

Conjecture 2 (Special-Purpose TM Obfuscation) There exists a signature scheme $\mathcal{S}$, CRHF $\mathcal{H}$ and a Turing Machine obfuscator $\mathsf{sp}\mathcal{O}$ such that the following hods: for any PPT attacker $\mathcal{A}$ there is a negligible $\varepsilon(\lambda)$ such that:

$$\Pr\left[\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1 \;\middle|\; \begin{array}{c}(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda), H \leftarrow \mathcal{H}_\lambda, \\ \tilde{M} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, M^*[H, \mathsf{sk}]), \\ (m, \sigma) \leftarrow \mathcal{A}(1^\lambda, \mathsf{vk}, \tilde{M})\end{array}\right] \leq \varepsilon(\lambda)$$

where the Turing Machine $M^*[H, \mathsf{sk}]$is defined above.

Theorem 2 *Under the special-purpose TM obfuscation conjecture (Conjecture 2), there is no general-purpose diO obfuscators (for circuits) that has security for bounded-length auxiliary input.*

In particular, under the conjecture, the circuit family $(\mathcal{C}, \mathsf{Sam}_{TM})$ defined above is a fixed differing-inputs family with some fixed polynomial bound on the length of the auxiliary input, yet there is no diO obfuscator for this particular family.

The proof of the above theorem is the same as that of Theorem 1.

Discussion. We note that candidate general-purpose iO and diO obfuscators for Turing Machines were constructed by [1,4]. Although the security claims rely on general-purpose (circuit) diO with auxiliary input, it seems reasonable to assume that these

constructions are secure in special cases, and also that they satisfies stronger security properties than merely iO and diO. In particular, using these candidate obfuscators, we do not know of any attacks on Conjecture 2. Moreover, it is still a falsifiable assumption once we fix some candidates $\mathcal{S}, \mathcal{H}, \mathsf{sp}\mathcal{O}$. On the other hand, the Turing Machine conjecture certainly seems stronger and more complex than the corresponding circuit conjecture (Conjecture 1).

6 Extending Implausibility to Extractable Witness Encryption

In previous section we showed that a "special-purpose obfuscation" conjecture (Conjecture 1) can be used to rule out existence of a general-purpose differing-inputs obfuscator. In this section we show that the same "special-purpose obfuscation" conjecture can also be used to rule out existence of extractable witness encryption. Note that this is a stronger result as general-purpose differing-inputs obfuscation is known to imply extractable witness encryption.

Theorem 3 *Under the special-purpose obfuscation conjecture (Conjecture 1), extractable witness encryption does not exist.*

Proof We prove our theorem by giving an ***NP***-relation R for which there does not exist an extractable witness encryption scheme. In order to prove this we will need to rely on our "special-purpose obfuscation" conjecture (Conjecture 1).

Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver})$ be a signature scheme with a deterministic signing algorithm. We define the ***NP***-relation R_{ver} so that $(\mathsf{vk}, (m, \sigma)) \in R_{ver}$ if and only if $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$. Let $(\mathsf{Enc}, \mathsf{Dec})$ be a candidate extractable witness encryption for this relation R. Given an string vk and a ciphertext c, let $C[\mathsf{vk}, \mathsf{c}](w)$ be the circuit that takes as input a witness w and computes $\mathsf{Dec}(\mathsf{c}, w)$. Let $\ell^*(\lambda)$ be the size of $C[\mathsf{vk}, \mathsf{c}]$.

We now define the same auxiliary input as in the previous section. Let $\mathcal{H} = \{\mathcal{H}_\lambda\}$ be a collision-resistant hash function (CRHF) family with output size $\ell_{in}(\lambda)$. Define the circuit family $\mathcal{C}$ consisting of circuits $C^*[H, \mathsf{sk}] \in \mathcal{C}$ defined as follows:

$C^*[H, \mathsf{sk}](C)$ // Hard-coded values: $H \in \mathcal{H}_\lambda$, sk signing key
// Input: C circuit of size $\ell^*(\lambda)$ with 1-bit output.
– Compute $m = H(C)$, $\sigma = \mathsf{Sig}_{\mathsf{sk}}(m)$.
– Output $C(m, \sigma)$.

Let $\mathsf{sp}\mathcal{O}$ be a "special purpose" obfuscator whose properties defined in Conjecture 1. We define the distribution samples $\mathsf{Sam}(1^\lambda)$ as follows:

– Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $H \leftarrow \mathcal{H}_\lambda$.
– Set $C^* = C^*[H, \mathsf{sk}] \in \mathcal{C}$ and $\mathsf{aux} \leftarrow \mathsf{sp}\mathcal{O}(C^*)$.
– Output vk, aux, where vk is the ***NP*** statement.

Now consider an experiment where we sample $(\mathsf{vk}, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)$ and encrypt $\mathsf{c} \leftarrow \mathsf{Enc}(1^\lambda, \mathsf{vk}, b)$ where $b \leftarrow \{0, 1\}$ and vk acts as an ***NP*** statement. We construct an adversary A that can output b with probability 1. Our adversary $A(1^\lambda, \mathsf{vk}, \mathsf{c}, \mathsf{aux})$ simply interprets aux as a circuit and outputs $\mathsf{aux}(C[\mathsf{vk}, \mathsf{c}])$. It is easy to see that, if

$\mathsf{c} = \mathsf{Enc}(1^\lambda, \mathsf{vk}, b)$, then $\mathsf{aux}(C[\mathsf{vk}, \mathsf{c}]) = b$ and therefore the adversary outputs b with probability 1.

On the other hand, we claim that no extractor E that can output valid witnesses given $(\mathsf{vk}, \mathsf{aux})$, contradicting the extractability property of the witness encryption scheme. Notice that finding a witness $w = (m, \sigma)$ for the statement consisting of a verification key vk under the relation R_{ver} is same as finding a valid message/signature pair given just the "special purpose" obfuscation aux and vk (the proof of this is similar to the proof of Lemma2). In other words, Conjecture 1 directly implies that for any PPT candidate extractor E there is a negligible ε such that:

$$\Pr[E(1^\lambda, \mathsf{vk}, \mathsf{aux}) = (m, \sigma) \text{ s.t. } (\mathsf{vk}, (m, \sigma)) \in R_{ver} \ : \ (x, \mathsf{aux}) \leftarrow \mathsf{Sam}(1^\lambda)] \leq \varepsilon(\lambda)$$

contradicting the extractability requirement of extractable witness encryption. This completes our proof. □

Bounded-Length Auxiliary Input. We could also define extractable witness encryption with bounded-length auxiliary input, where the encryption/decryption procedures can all depend on the size of the auxiliary input. This would be analogous to the definition of diO with bounded-length auxiliary input. We can rule out this notion of witness encryption with bounded-length auxiliary input under our special-purpose *Turing Machine* obfuscation assumption (Conjecture 2) analogously to our results for diO in Sect. 5.

7 Output-Only Dependent Hardcore Bits

In a recent work, Bellare, Stepanovs and Tessaro [11] show the existence of polynomially many hardcore bits for any one-way function. In the case of *injective* one-way functions, their construction relies on indistinguishability obfuscation. However, in the case of *arbitrary* one-way functions, it relies on diO with auxiliary input. The construction has a very interesting property which we call "output-only dependence". In particular, even if the one-way function $f(x)$ is many-to-one, the hardcore bits $h(x)$ are completely determined by $f(x)$; for any inputs x, x' such that $f(x) = f(x')$ we also get $h(x) = h(x')$. This property is interesting even in the case of a single hardcore bit, and does *not* hold for any of the known general constructions (such as for the Goldreich-Levin bit).

Unfortunately, we show that our special-purpose obfuscation assumption (for Turing Machines) also gives a counterexample to the security of the hardcore bit construction of [11]. More generally, we show that there is a contrived one-way function that does not have *any* output-only dependent hardcore bit. In more detail:

– Under the special-purpose obfuscation conjecture for *circuits* (Conjecture 1), we construct a one-way function that does not have any output-only dependent hardcore bits given *auxiliary input*.[4]

[4] The result of Bellare, Stepanovs and Tessaro [11] does not consider auxiliary input.

– Under the special-purpose obfuscation conjecture for *Turing Machines* (Conjecture 2) we get the above result even *without* auxiliary input. In particular, we construct a one-way function which does not have any output-only dependent hardcore bits.

The formal definitions and statements follow.

7.1 Definitions

7.1.1 One-Way Function Families (with Auxiliary Input).

A *one-way function family* consists of two polynomial-time procedures $(\mathsf{FKeyGen}, f)$ and an input-size $n(\cdot)$, with the following syntax: $\mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda)$ is a randomized algorithm that generates the function-key fk and $f_{\mathsf{fk}}(x)$ is a deterministic algorithm that evaluates the function with function-key fk and input $x \in \{0,1\}^{n(\lambda)}$. We say that $(\mathsf{FKeyGen}, f)$ is a one-way function family if for all PPT attackers $\mathcal{A}$ there is a negligible $\varepsilon(\cdot)$ such that

$$\Pr\left[f_{\mathsf{fk}}(x') = y \;\middle|\; \begin{array}{c} \mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda), x \leftarrow \{0,1\}^{n(\lambda)} \\ y := f_{\mathsf{fk}}(x), x' \leftarrow \mathcal{A}(1^\lambda, \mathsf{fk}, y) \end{array} \right] \leq \varepsilon(\lambda).$$

We also define a *one-way function family with auxiliary input*. If $(\mathsf{FKeyGen}, f)$ is a one-way function family, then a *compatible auxiliary input* consists of a sampling algorithm $(\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda)$ such that the distribution of fk is exactly the same when generated by $\mathsf{FKeyGen}$ and $\mathsf{FAuxGen}$. Furthermore, we require one-way security to hold even given the auxiliary info aux. In particular, for any PPT attacker $\mathcal{A}$ there is a negligible $\varepsilon(\cdot)$ such that

$$\Pr\left[f_{\mathsf{fk}}(x') = y \;\middle|\; \begin{array}{c} (\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda), x \leftarrow \{0,1\}^{n(\lambda)}, \\ y := f_{\mathsf{fk}}(x), x' \leftarrow \mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{aux}, y) \end{array} \right] \leq \varepsilon(\lambda).$$

7.1.2 Hardcore-Bit

A *hardcore bit* for a one-way function family $(\mathsf{FKeyGen}, f)$ consists of two polynomial-time procedures $(\mathsf{HKeyGen}, h)$ with the following syntax: $\mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk})$ is a randomized algorithm that generates the hardcore-function-key hk and $h_{\mathsf{hk}}(x)$ takes as input $x \in \{0,1\}^{n(\lambda)}$ and outputs a bit $b \in \{0,1\}$. For security, we require the following: for all PPT attackers $\mathcal{A}$ there is a negligible $\varepsilon(\cdot)$ such that:

$$\left| \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, f_{\mathsf{fk}}(x), h_{\mathsf{hk}}(x)) = 1\ ] - \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, f_{\mathsf{fk}}(x), b) = 1] \right| \leq \varepsilon(\lambda).$$

where $\mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda)$, $\mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk})$, $x \leftarrow \{0,1\}^{n(\lambda)}$ and $b \leftarrow \{0,1\}$.

In the setting of *auxiliary input*, the syntax of hard-core bits is the same. For a one-way function family $(\mathsf{FKeyGen}, f)$, we say that $(\mathsf{HKeyGen}, h)$ is a *hardcore bit*

with auxiliary input if for every compatible auxiliary input $\mathsf{FAuxGen}$ and every PPT attacker $\mathcal{A}$ we have:

$$\left| \begin{array}{l} \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, f_{\mathsf{fk}}(x), \mathsf{aux}, h_{\mathsf{hk}}(x)) = 1\ ] \\ -\ \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, f_{\mathsf{fk}}(x), \mathsf{aux}, b) = 1] \end{array} \right| \leq \varepsilon(\lambda).$$

where $(\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda)$, $\mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk})$, $x \leftarrow \{0,1\}^{n(\lambda)}$ and $b \leftarrow \{0,1\}$. Known constructions of hardcore bits (e.g., the Goldreich-Levin bit) are secure in the presence of auxiliary input.

We say that a hardcore-bit $(\mathsf{HKeyGen}, h)$ for a one-way function family $(\mathsf{FKeyGen}, f)$ is *output-only dependent* if there is some negligible $\varepsilon(\lambda)$ such that:

$$\Pr[\exists\ x, x' \in \{0,1\}^{n(\lambda)} \text{ such that } f_{\mathsf{fk}}(x) = f_{\mathsf{fk}}(x') \text{ and } h_{\mathsf{hk}}(x) \neq h_{\mathsf{hk}}(x')] \leq \varepsilon(\lambda)$$

where the probability is over $\mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda)$, $\mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk})$. Informally, this means that $h_{\mathsf{hk}}(x)$ is completely determined by $f_{\mathsf{fk}}(x)$ with overwhelming probability. Known constructions of general hardcore bits (e.g., the Goldreich-Levin bit) are *not* output-only dependent.

7.2 Counterexample with Auxiliary Input

Let $g : \{\{0,1\}^\lambda \to \{0,1\}^{m(\lambda)}\}_{\lambda \in \mathbb{N}}$ be any (fixed) one-way function so that for all PPT attackers $\mathcal{A}$ there is a negligible ε such that:

$$\Pr[g(r') = g(r)\ :\ r \leftarrow \{0,1\}^\lambda, r' \leftarrow \mathcal{A}(1^\lambda, g(r))] \leq \varepsilon(\lambda).$$

Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver})$ be a signature scheme with signature-length $\ell_{sig}(\lambda)$ and let $\mathcal{H}$ be a CRHF with output size $\ell_{hash}(\lambda)$. Define the one-way function family $(\mathsf{FKeyGen}, f)$ as follows.

- $\mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda)$: Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ to be the signature signing/verification keys. Set $\mathsf{fk} := \mathsf{vk}$.
- $f_{\mathsf{fk}}(x)$: Interpret $x = (r, m, \sigma, y)$, $\mathsf{vk} = \mathsf{fk}$. If $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$ output y else output $g(r)$.

The input size to $f_{\mathsf{fk}}(\cdot)$ is $n(\lambda) = \lambda + \ell_{hash}(\lambda) + \ell_{sig}(\lambda) + m(\lambda)$ and the output size is $m(\lambda)$. It is easy to show that the one-way security of this construction follows from the one-way security of g and the signature security of $\mathcal{S}$. Intuitively, on a uniformly random x, $f_{\mathsf{fk}}(x) = g(x)$ with overwhelming probability. Inverting $f_{\mathsf{fk}}(x)$ therefore requires either (I) inverting $g(x)$ or (II) coming up with a valid message/signature pair (m, σ) given vk.

We now define a class of compatible auxiliary inputs for $(\mathsf{FKeyGen}, f)$, parameterized by some polynomial length-bound $\ell^*(\lambda)$. Let the circuit family $\mathcal{C}$, consisting of circuits $C^*[H, \mathsf{sk}] \in \mathcal{C}$, be defined the same way as previously. Let $\mathsf{sp}\mathcal{O}$ be any special-purpose obfuscator for circuits. The auxiliary-input sampling algorithm is defined via $\mathsf{FAuxGen}_{\ell^*}$ as follows:

– $(\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}_{\ell^*}(1^\lambda)$: Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ to be the signature signing/verification keys. Set $\mathsf{fk} = \mathsf{vk}$. Sample a hash function $H \leftarrow \mathcal{H}_\lambda$ and let $C^*[H, \mathsf{sk}] \in \mathcal{C}$ be a circuit with input length ℓ^*. Set $\mathsf{aux} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, C^*[H, \mathsf{sk}])$.

Lemma 4 *Under the special-purpose obfuscation conjecture for circuits (Conjecture 1) and the one-way security of the function g, the function family $(\mathsf{FKeyGen}, f)$ is one-way and, for any polynomial ℓ^*, $\mathsf{FAuxGen}_{\ell^*}$ is compatible auxiliary input.*

Proof Assume there is a polynomial $\ell^*(\lambda)$ and an attacker $\mathcal{A}$ whose probability of winning the one-wayness game with auxiliary input is:

$$\Pr\left[f_{\mathsf{fk}}(x') = y \;\middle|\; \begin{array}{c} (\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda), x \leftarrow \{0,1\}^{n(\lambda)}, \\ y := f_{\mathsf{fk}}(x), x' \leftarrow \mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{aux}, y) \end{array} \right] = \varepsilon(\lambda).$$

By the security of the signature scheme, we know that for a random (m, σ) the probability of $\mathsf{Ver}_{\mathsf{vk}}(m, \sigma) = 1$ is negligible. Therefore, with overwhelming probability, $f_{\mathsf{fk}}(x) = g(r)$ when we choose $x = (r, m, \sigma, y)$ at random. We can write:

$$\Pr\left[f_{\mathsf{fk}}(x') = y \;\middle|\; \begin{array}{c} (\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda), r \leftarrow \{0,1\}^\lambda, \\ y := g(r), x' \leftarrow \mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{aux}, y) \end{array} \right] = \varepsilon(\lambda) - \mathsf{negl}(\lambda).$$

Let us define the event E_1 to be the event that $\mathcal{A}$ wins with $x' = (r', m', \sigma', y')$ where $g(r') = g(r)$. Let E_2 be the event that $\mathcal{A}$ wins and $\mathsf{Ver}_{\mathsf{vk}}(m', \sigma') = 1$. Since one of E_1, E_2 must happen whenever $\mathcal{A}$ wins, we have $\Pr[E_1] + \Pr[E_2] + \mathsf{negl}(\lambda) \geq \varepsilon(\lambda)$. By the one-wayness of g, we have $\Pr[E_1] = \mathsf{negl}(\lambda)$. By the special-purpose obfuscation assumption, we have $\Pr[E_2] = \mathsf{negl}(\lambda)$. Therefore $\varepsilon(\lambda) = \mathsf{negl}(\lambda)$ which proves the lemma. □

Theorem 4 *Under the special-purpose obfuscation conjecture for circuits (Conjecture 1), the function family $(\mathsf{FKeyGen}, f)$ is one-way but does not have any output-only dependent hardcore bit with auxiliary input. In particular, for any candidate hardcore-bit construction $(\mathsf{HKeyGen}, h)$ there is a polynomial ℓ^* such that the $\mathsf{FAuxGen}_{\ell^*}$ is some compatible auxiliary input which breaks the security of the hardcore bit.*

Proof Let $(\mathsf{HKeyGen}, h)$ be any candidate hardcore bit construction for $(\mathsf{FKeyGen}, f)$, and assume that it is output-only dependent. Consider the circuit $C[\mathsf{hk}, y](m, \sigma)$ which gets $(m, \sigma) \in \{0,1\}^{\ell_{hash}(\lambda) + \ell_{sig}(\lambda)}$ as input, constructs $x' = (0^\lambda, m, \sigma, y)$ and outputs $h_{\mathsf{hk}}(x)$. Let $\ell^*(\lambda)$ be the size of this circuit when $\mathsf{fk} \leftarrow \mathsf{FKeyGen}(1^\lambda)$ and $\mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk})$.

By Lemma 4, we know that under Conjecture 1, the sampler $\mathsf{FAuxGen}_{\ell^*}$ is compatible auxiliary input for the one-way function family $(\mathsf{FKeyGen}, f)$. We now show that the hardcore bit $(\mathsf{HKeyGen}, h)$ can be easily distinguished given the auxiliary input aux. In particular, consider the distinguisher $\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, y, \mathsf{aux}, b)$ that constructs the circuit $C[\mathsf{hk}, y]$ described above and runs $b' = \mathsf{aux}(C[\mathsf{hk}, y])$. If $b' = b$ it outputs 1 else 0. We claim that:

$$\left| \begin{array}{l} \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, y, \mathsf{aux}, h_{\mathsf{hk}}(x)) = 1\ ] \\ -\ \Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, y, \mathsf{aux}, b) = 1] \end{array} \right| = \frac{1}{2} - \mathsf{negl}(\lambda)$$

where $(\mathsf{fk}, \mathsf{aux}) \leftarrow \mathsf{FAuxGen}(1^\lambda), \mathsf{hk} \leftarrow \mathsf{HKeyGen}(1^\lambda, \mathsf{fk}), x \leftarrow \{0,1\}^{n(\lambda)} y = f_{\mathsf{fk}}(x)$ and $b \leftarrow \{0,1\}$.

By definition, we have $b' = \mathsf{aux}(C[\mathsf{hk}, y]) = C[\mathsf{hk}, y](m', \sigma') = h_{\mathsf{hk}}(x')$ for some pre-image $x' = (0^\lambda, m', \sigma', y)$ such that $f_{\mathsf{fk}}(x') = f_{\mathsf{fk}}(x) = y$. By output-only dependence, we also have $h_{\mathsf{hk}}(x') = h_{\mathsf{hk}}(x)$ with probability $1 - \mathsf{negl}(\lambda)$. Therefore $\Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, y, \mathsf{aux}, h_{\mathsf{hk}}(x)) = 1\ ] = 1 - \mathsf{negl}(\lambda)$ and $\Pr[\mathcal{A}(1^\lambda, \mathsf{fk}, \mathsf{hk}, y, \mathsf{aux}, b) = 1\ ] = \frac{1}{2}$, which proves the theorem. □

7.3 Counterexample Without Auxiliary Input

We can modify the above counterexample to work in the setting without auxiliary input by simply thinking of the auxiliary input aux as part of the function key. In particular, we define $\mathsf{FKeyGen}(1^\lambda)$ to set $\mathsf{fk} = (\mathsf{vk}, \mathsf{aux})$ where aux is defined as above. The only problem is that we now get a circular dependence on sizes: the size of $\mathsf{fk} = (\mathsf{vk}, \mathsf{aux})$ exceeds the size of aux which needs to exceed the circuit-size of $h_{\mathsf{hk}}(\cdot)$ which in turn can depend on (and exceed) the size of fk. Indeed, the construction of [10] does have this property where the size of hk (and therefore also the circuit-size of h_{hk}) exceeds the size of fk. To get around this, we can use the same trick as in Sect. 5 by relying on a Turing Machine obfuscator rather than a circuit obfuscator. In particular, we can set aux to be a Turing Machine obfuscation of the breaker TM $M^*[H, \mathsf{sk}]$ defined in Sect. 5. The size of the obfuscated circuit aux is now some fixed polynomial no matter what the size ℓ^* is of the input that we want to feed to aux.

More formally, let $\mathsf{sp}\mathcal{O}$ be a special-purpose Turing Machine obfuscator and define the function family $(\mathsf{FKeyGen}_{TM}, f)$ where:

- $\mathsf{fk} \leftarrow \mathsf{FKeyGen}_{TM}(1^\lambda)$: Sample $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ to be the signature signing/verification keys. Sample a hash function $H \leftarrow \mathcal{H}_\lambda$ and let $M^*[H, \mathsf{sk}]$ be a TM defined the same way as in Sect. 5. Let $\mathsf{aux} \leftarrow \mathsf{sp}\mathcal{O}(1^\lambda, M^*[H, \mathsf{sk}])$. Set $\mathsf{fk} = (\mathsf{vk}, \mathsf{aux})$.
- f_{fk} is defined the same way as previously and ignores aux.

Theorem 5 *Under the special-purpose obfuscation conjecture for Turing Machines (Conjecture 2), the function family* $(\mathsf{FKeyGen}_{TM}, f)$ *is one-way but does not have any output-only dependent hardcore bit.*

The proof of the above theorem is the same as that of Theorem 4.

8 Conclusions

We propose a seemingly reasonable "special-purpose" obfuscation conjecture under which general-purpose diO and extractable witness encryption with auxiliary input cannot exist. Furthermore a variant of this conjecture also shows the impossibility of output-only dependent hardcore bits for every one-way function. Many interesting

open problems remain. Firstly, is there some inherent reason why our conjecture cannot hold? This is certainly possible, and we cannot objectively say which of the two conflicting possibilities (diO with auxiliary input vs. our conjecture) is false. However, the conjecture is a simple-to-state falsifiable assumption. Showing the possibility of general-purpose diO and extractable witness encryption would require coming up with an attack on this conjecture. On the other hand, general-purpose diO and witness encryption are not stated as falsifiable assumptions; indeed we give a candidate attack on these notions, but we cannot efficiently check if the attack is valid. In the absence of further evidence, we choose to interpret this result as giving strong evidence that general-purpose diO and extractable witness encryption are "implausible". Is there a way to convert this "implausibility" result into an "impossibility" result? On a different note, is it still reasonable to assume the existence of general-purpose diO *without* auxiliary input? We do not see any way to extend our "implausibility" result to the case without auxiliary input. Lastly, it remains as an interesting open problem to characterize the known techniques for getting obfuscation impossibility results, and come up with a strong and general obfuscation assumption that capture everything which is not directly ruled out by these techniques.

Acknowledgements We thank Mariana Raykvoa and Amit Sahai for initial discussions relating to this work, Nir Bitansky for suggesting we look at extractable witness encryption, and Mihir Bellare for pointing us to his paper on poly-many hardcore bits and for suggesting we consider diO with bounded-length auxiliary input.

References

1. Ananth, P., Boneh, D., Garg, S., Sahai, A., Zhandry, M.: Differing-inputs obfuscation and applications. Cryptology ePrint Archive. Report 2013/689 (2013) http://eprint.iacr.org/
2. Applebaum, B.: Bootstrapping obfuscators via fast pseudorandom functions. In: Sarkar and Iwata [21], pp. 162–172
3. Bitansky, N., Canetti, R., Cohn, H., Goldwasser, S., Kalai, Y.T., Paneth, O., Rosen, A.: The impossibility of obfuscation with auxiliary input or a universal simulator. In: Advances in Cryptology—CRYPTO 2014—34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2014, Proceedings, Part II, pp. 71–89 (2014)
4. Boyle, E., Chung, K.-M., Pass, R.: On extractability obfuscation. In: Lindell [18], pp. 52–73
5. Bitansky, N., Canetti, R., Paneth, O., Rosen, A.: On the existence of extractable one-way functions. In: Shmoys [20], pp. 505–514
6. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: CRYPTO, pp. 1–18 (2001)
7. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM **59**(2), 6 (2012)
8. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: Phong, Q. Nguyen, Elisabeth O., (eds.) Advances in Cryptology—EUROCRYPT 2014—33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11–15, 2014. Proceedings, volume 8441 of Lecture Notes in Computer Science, pp. 221–238. Springer (2014)
9. Boyle, E., Pass, R.: Limits of extractability assumptions with distributional auxiliary input. In: Tetsu I., Jung H.C. (eds.) Advances in cryptology—ASIACRYPT 2015—21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29—December 3, 2015, Proceedings, Part II, volume 9453 of Lecture Notes in Computer Science, pp. 236–261. Springer, (2015)

10. Brakerski, Z., Rothblum, G.N.: Virtual black-box obfuscation for all circuits via generic graded encoding. In: Lindell [18], pp. 1–25
11. Bellare, M., Stepanovs, I., Tessaro, S.: Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In: Sarkar, Iwata [21], pp. 102–121
12. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26–29 October, 2013, Berkeley, CA, USA, pp. 40–49. IEEE Computer Society (2013)
13. Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Dan, B., Tim, R., Joan, F., (eds), Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1–4, 2013, pp. 467–476. ACM (2013)
14. Goldwasser, S., Kalai, Y.T.: On the impossibility of obfuscation with auxiliary input. In: FOCS, pp. 553–562 (2005)
15. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to run turing machines on encrypted data. In: Ran C., Juan A.G. (eds), CRYPTO (2), volume 8043 of Lecture Notes in Computer Science, pp. 536–553. Springer (2013)
16. Hada, S.: Zero-knowledge and code obfuscation. In: Tatsuaki O., (ed), ASIACRYPT, volume 1976 of Lecture Notes in Computer Science, pp. 443–457. Springer (2000)
17. Ishai, Y., Pandey, O., Sahai, A.: Public-coin differing-inputs obfuscation and its applications. In: Yevgeniy D., Jesper Buus N., (eds), Theory of Cryptography—12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23–25, 2015, Proceedings, Part II, volume 9015 of Lecture Notes in Computer Science, pp. 668–697. Springer (2015)
18. Lindell, Y. (ed): Theory of cryptography—11th Theory of cryptography conference, TCC 2014, San Diego, CA, USA, February 24–26, 2014. Proceedings, volume 8349 of Lecture Notes in Computer Science. Springer (2014)
19. Naor, M.: On cryptographic assumptions and challenges. In: Dan B., (ed), CRYPTO, volume 2729 of Lecture Notes in Computer Science, pp. 96–109. Springer (2003)
20. Shmoys, D.B. (ed): Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31—June 03, 2014. ACM (2014)
21. Sarkar, P., Iwata, T. (eds): Advances in cryptology—ASIACRYPT 2014—20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014, Proceedings, Part II, volume 8874 of Lecture Notes in Computer Science. Springer (2014)
22. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys [20], pp. 475–484