Poly-Many Hardcore Bits for Any One-Way Function and a Framework for Differing-Inputs Obfuscation

Mihir Bellare[1], Igors Stepanovs[1], and Stefano Tessaro[2]

[1] Department of Computer Science and Engineering, University of California San Diego, USA. `http://cseweb.ucsd.edu/~mihir/`

[2] Department of Computer Science, University of California Santa Barbara, USA. `http://www.cs.ucsb.edu/~tessaro/`

Abstract. We show how to extract an arbitrary polynomial number of simultaneously hardcore bits from any one-way function. In the case the one-way function is injective or has polynomially-bounded pre-image size, we assume the existence of indistinguishability obfuscation (iO). In the general case, we assume the existence of differing-input obfuscation (diO), but of a form weaker than full auxiliary-input diO. Our construction for injective one-way functions extends to extract hardcore bits on multiple, correlated inputs, yielding new D-PKE schemes. Of independent interest is a definitional framework for differing-inputs obfuscation in which security is parameterized by circuit-sampler classes.

1 Introduction

When RSA was invented [52], the understanding was that public-key encryption (PKE) would consist of applying the RSA one-way function f directly to the message. In advancing semantic security (unachieved by this plain RSA scheme) as the appropriate target for PKE, Goldwasser and Micali [37] created a new challenge. Even given RSA, how was one to achieve semantic security?

The answer was hardcore functions [17, 58, 37]. Let f be a one-way function, and h a function that has the same domain as f and returns strings of some length s. We say that h is hardcore for f if the distributions $(f, h, f(x), h(x))$ and $(f, h, f(x), r)$ are computationally indistinguishable when x is chosen at random from the domain of f and r is a random s-bit string.[3] We will refer to the output length s of h, which is the number of (simultaneously) hardcore bits produced by h, as the *span* of h, and say that f achieves a certain span if there exists a hardcore function h for f with the span in question. Hardcore predicates are hardcore functions with span one. Semantically-secure PKE can now be easily achieved: encrypt s-bit message m as $(f(x), h(x) \oplus m)$ for random x where trapdoor, injective one-way function f together with hardcore function

[3] In the formal definitions in Section 2, both one-way functions and hardcore functions are families. Think here of f, h as instances chosen at random from the respective families, their descriptions public.

h constitute the public key. The span s determines the number of message bits that can be encrypted, so the larger it is, the better.

The quest for semantic security turned into a quest for hardcore functions [17, 58, 37, 54, 15, 42, 3, 48, 39, 56, 46, 35, 40]. It became quickly evident that finding hardcore predicates, let alone hardcore functions of span larger than one, was very challenging, and many sophisticated techniques were invented. With time, encryption methods avoiding hardcore functions did emerge [26]. But hardcore functions have never lost their position as a fundamental cryptographic primitive, as new usages, applications and constructions have arisen [28, 25, 1, 2, 27]. Hardcore functions have been particularly important in developing new forms of encryption, including lossy-TDF based encryption [51] and deterministic encryption [6, 8, 30].

Generic constructions. Across all this work and applications, something that stands out is the value and appeal of generic constructions, the representative result being that of Goldreich and Levin (GL) [35]. Prior to this, hardcore functions were obtained by dedicated and involved analyses that exploited the algebra underlying the one-way function itself. GL [35] showed that the inner product of x with a random string (the latter is part of the description of h) results in a hardcore predicate for any OWF. This result has had an enormous impact, as evidenced by over 900 citations to the paper to date. A generic construction such as that of GL allows theoreticians to develop constructions and proofs independently of specific algebraic assumptions. Furthermore, it allows us to immediately obtain hardcore functions, and thus encryption, from new, candidate one-way functions.

Questions and answers. The GL construction however only provided a hardcore predicate, meaning span one, and by extension logarithmic span. The most desired goal was polynomial span.[4] Significant effort was been invested towards this goal, allowing it to be reached for specific algebraic OWFs [40, 25, 47, 50, 2] or ones satisfying extra properties [51]. But a generic construction providing polynomial span seemed out of reach.

This is the question resolved by our work. We present generic constructions of hardcore functions with polynomial span for both injective and non-injective one-way functions. The tools we use are indistinguishability obfuscation (iO) [5, 31] and differing-inputs obfuscation (diO) [5, 20, 4]. See Fig. 1 for a summary of our results.

Prior work. In more detail, early results gave hardcore predicates (ie. span one) for specific one-way functions including discrete exponentiation modulo a prime, RSA and Rabin [17, 58, 54, 15, 42, 3, 48, 39, 28]. Eventually, as noted above, the impactful and influential work of Goldreich and Levin [35] gave a generic hardcore predicate for any one-way function. Extensions of these results are able to achieve logarithmic span [56, 46, 35, 1, 27].

[4] Once one can obtain a particular polynomial number of output bits, one can always expand to an arbitrary polynomial via a PRG, so we do not distinguish these cases.

OWF	**Span**	**Assumption**	**Construction**	**See**
injective	poly	iO	**HC1**	Cor. 5 of Th. 4
poly pre-image size	poly	iO	**HC2**	Cor. 8 of Th. 6
any	poly	diO$^-$	**HC2**	Cor. 7 of Th. 6

Fig. 1. Our results: We indicate the assumptions we make in order to construct hardcore functions with polynomial span. By diO$^-$ we mean a weakening of diO that we define.

Hardcore functions with polynomial span have been provided for specific, algebraic functions, usually under a stronger assumption than one-wayness of the function itself. These include the works by Håstad, Schrift and Shamir [40] for discrete exponentiation modulo a composite, by Patel and Sundaram [50] for discrete exponentiation modulo a safe prime, by Catalano, Gennaro and Howgrave-Graham [25] for the Paillier function [49], and by Akavia, Goldwasser and Vaikuntanathan [2] for certain LWE-based functions. Moreover, polynomial-span hardcore functions for RSA [45, 55] and the Rabin trapdoor function [55] have been given under different assumptions. Peikert and Waters showed that lossy trapdoor functions [51] achieve polynomial span, yielding further examples of specific one-way functions with polynomial span [51, 29].

The basic question that remains open was whether polynomial span is achievable for an arbitrary, given OWF, meaning whether there is a generic hardcore function with polynomial span. One answer was provided by [9], who showed that UCE-security (specifically, relative to split, computationally-unpredictable sources) of a function h with polynomial span suffices for h to be hardcore for any one-way function, but the assumption made is arguably close to the desired conclusion.

Our results. Injective one-way functions are the most important case for aplications. (Most applications are related to some form of encryption.) Accordingly we begin by focusing on the case where f is injective. We give a construction of a hardcore function called **HC1** that provides polynomial span for *any* injective OWF f. Beyond one-wayness of f we assume iO [5, 53].

We then provide the **HC2** construction of a hardcore function with polynomial span for any, even non-injective OWF f. In the case f has polynomially-bounded pre-image size, the extra assumption remains iO. In the general case, it is diO, but of a form that is weaker than full auxiliary-input diO as defined in [20, 4].

As direct corollaries, we obtain hardcore functions of polynomial span for many specific OWFs f, in some cases providing the first hardcore function with polynomial span for this f, in others providing a construction under different and new assumptions compared to prior ones. Thus for the basic discrete exponentiation OWF over the integers modulo a prime or an elliptic curve group, we provide the first construction with polynomial span. (Results were known when exponents are short [33, 50] and for discrete exponentiation modulo composites [40, 36].)

Our results have the above-noted benefits of generic constructions. They yield conceptual simplifications for the development of theoretical cryptography, and also provide automatic ways to get hardcore functions for new, candidate one-way functions. We view our work as also been interesting from the point of view of showing the power and applicability of iO, in the wake of Sahai-Waters [53].

Technical approach. We now take a closer look at our constructions and proofs to highlight the technical approach and novelties. Recall that the guarantee of iO [5, 53] is that the obfuscations of two circuits are indistinguishable if the circuits themselves are equivalent, meaning return the same output on all inputs. Differing-inputs obfuscation (diO) [5, 20, 4] relaxes the equivalence condition, asking instead that it only be hard, given the (unobfuscated) circuits, to find an input where they differ. See Section 3 for formal definitions.

Our **HC1** construction is a natural one, namely to let h be an obfuscation of the circuit $\mathsf{G}(gk, \cdot)$ where G is a PRF [34] and gk is a random key for G. Our proof assumes the PRF is punctured [19, 43, 21]. (This is not an extra assumption, as we assume the existence of one-way functions.) Moreover, the proof uses a weak form of diO shown by Boyle, Chung and Pass (BCP) [20] to be implied by iO. The proof considers an adversary $\mathcal{H}$ provided with $f, h, f(x^*), r^*$ (where f is injective) and we want to move from the real game, in which $r^* = \mathsf{G}(gk, x^*)$, to the random game, in which r^* is random. We begin by using the SW technique [53] to move to a game in which r^* is random and h is an obfuscation of the circuit C^1 that embeds the target input x^*, a punctured PRF key, and a random point r^*, returning the latter when called on $x = x^*$ (the trigger) and otherwise returning $\mathsf{G}(gk, x)$, computed via the punctured key. (This move relies on iO and punctured PRF security.) While this has made r^* random as desired, h is not what it should be in the random game, where it is in fact an obfuscation of the real circuit $\mathsf{G}(gk, \cdot)$. The difficulty is to move h back to an obfuscation of this real circuit while leaving r^* random. We realize that such a move must exploit the one-wayness of f, which has not so far been used. A one-wayness adversary, given $f(x^*)$ and aiming to find x^*, needs to run $\mathcal{H}$. The problem is that $\mathcal{H}$ needs an obfuscation of the above-described circuit C^1 as input, and construction of C^1 requires knowing the very point x^* that the one-wayness adversary is trying to find. The difficulty is inherent rather than merely one of proof, for the forms of obfuscation being used give no guarantee that an obfuscation of C^1 does not reveal x^*. We get around this by changing the trigger check from $x = x^*$ to $f(x) = f(x^*)$, so that now the circuit can embed $f(x^*)$ rather than x^*, a quantity there is no harm in revealing. This is reminescent of the technique in the security proof of the short signature scheme of Sahai and Waters [53]. The new check is equivalent to the old if f is injective, which is where we use this assumption. But we have still not arrived at the random game. We note that our modified circuit C^2 and the target circuit $\mathsf{G}(gk, \cdot)$ of the random game are inherently non-equivalent, and iO would not apply. However, these circuits differ only at input x^*. We exploit the one-wayness of f to prove that it is hard to find this input even given the two circuits. The assumed diO-security of the obfuscator now implies that the obfuscations of these circuits are indistinguishable, allowing us to conclude. Finally, since we exploit diO only for circuits that differ at one (hard to find) point, BCP [20] says that iO in fact suffices, making iO the only assumption needed for the result beyond the necessary one-wayness of f.

The above argument makes crucial use of the assumption that f is injective. To handle an arbitrary one-way function, our **HC2** construction modifies the above so that h is an obfuscation of the circuit $\mathsf{G}(gk, f(\cdot))$ where gk is a random key for punctured PRF G. Thus, $h(x) = \mathsf{G}(gk, f(x))$. We refer to Section 5 for the proof but note that BCP [20] implies that iO suffices for our proof when the number of preimages of any output of f is polynomial, but in general it could be exponential. In this case, diO suffices, but in fact a weaker version of it, that we define, does as well.

In summary, in the important case of injective one-way functions, and even for one-way functions with polynomial pre-image size, we provide a hardcore function with polynomial span assuming only iO. For one-way functions with super-polynomial pre-image size, we provide a hardcore function with polynomial span assuming a weak form of diO that we denote by diO$^-$ in Fig. 1.

Extensions and applications. We show that our **HC1** hardcore function construction is able to extract random, independent bits even on inputs that are arbitrarily correlated, which is not true of most prior constructions and, combined with the fact that we get polynomially-many output bits for each input, yields new applications. In more detail, an injective function f is said to be one-way on a distribution $\mathcal{I}$ over vectors $\mathbf{x}$ if, given the result $f(\mathbf{x})$ of applying f to $\mathbf{x}$ component-wise, it is hard to recover any component of $\mathbf{x}$. We want a hardcore function such that the components of $h(\mathbf{x})$ look not only random but *independent* even given $f(\mathbf{x})$. If the components of $\mathbf{x}$ are independent, this is true for any hardcore function meeting the standard definition, and thus for ours, but if the components are correlated, standard hardcore functions give no such guarantee and may fail. As an example, many existing hardcore functions [17, 58, 54, 15, 42, 3, 48, 39, 28, 56, 46, 35, 1, 27, 40, 25, 49, 2] return certain specific bits of their input and will thus fail to be hardcore relative to the distribution in which $\mathbf{x}[2]$ is the bitwise complement of $\mathbf{x}[1]$, even if f is one-way on this distribution. We show however that **HC1** remains hardcore for f on *any* efficiently sampleable distribution $\mathcal{I}$ over which f is one-way, even when the entries of the vectors produced by $\mathcal{I}$ are arbitrarily correlated. This answers open questions from [38, 30].

Deterministic PKE (D-PKE) is useful for many applications, including efficient search on encrypted data [6] and providing resilience in the face of the low-quality randomness that pervades systems [7]. However, it cannot provide IND-CPA security. BBO [6] define what it means for a D-PKE scheme to provide PRIV-security over an input sampler $\mathcal{I}$, the latter returning vectors of *arbitrarily correlated* messages to be encrypted. We restrict attention to distributions that are admissible, meaning that there exists a family of injective, trapdoor functions that is one-way relative to $\mathcal{I}$. The basic question that emerges is, for which admissible distributions $\mathcal{I}$ does there exist a D-PKE scheme that is PRIV-secure over $\mathcal{I}$? We show that this is true for all admissible distributions that are efficiently sampleable, assuming only the existence of iO. We obtain this result using the security of **HC1** on correlated inputs and techniques from [30]. Previously, this was known only in the ROM [6], under the assumption that UCE-secure functions exist [9], for distributions with limited correlation between messages [8, 18] or assuming lossy trapdoor functions [30]. See [12] for definitions, a precise statement of the result, and proofs.

Parameterized diO. Of independent interest, we provide a definitional framework for diO in which security is parameterized by a class of circuit samplers. This allows us to unify and capture variant notions of iO and diO in the literature [5, 53, 20, 4]. The framework makes it easy to define further variants that are weaker than full diO, yielding a language in which one can state assumptions that are closer to those actually used in the proof rather than being overkill, particularly with regard to the type of auxiliary inputs used [13, 22, 32]. This is useful in the light of work like [32] which indicates that diO with arbitrary auxiliary inputs may be implausible. The weaker notion of diO noted above and denoted diO$^-$ in Fig. 1 is formally defined in our framework as diO security for "short" auxiliary inputs, which evades the negative results of [32]. See Section 3.

Discussion and related work. Random oracles (ROs) are "ideal" hardcore functions, able to provide polynomial span for any one-way function [10]. Our results, akin to [44, 9, 41], can thus be seen as instantiating the RO in a natural ROM construction, in particular showing hardcore functions in the standard model that are just as good as those in the ROM. As a consequence, we are able to instantiate the RO in the BR93 PKE scheme [10] to obtain a standard-model IND-CPA scheme.

The hardcore function in our second construction is the reverse of the hash function used to instantiate FDH in HSW [41]: in our case, the circuit being obfuscated first applies a one-way function and then a punctured PRF, while in their case it first applies a punctured PRF and then a one-way function.

Our work adopts the standard definition of a one-way function in which any polynomial-time adversary must have negligible inversion advantage. Polynomial span is known to be achievable for any *exponentially* hard to invert function [35, 27].

Given a one-way permutation f and a polynomial n it is possible to construct another one-way permutation g that has n hardcore bits. Namely let $g(x) = f^n(x)$ and let the hardcore function on x be the result of the Blum-Micali-Yao PRG [17, 58] on seed x. A similar transform is provided in [16]. We provide hardcore functions directly for any given one-way function rather than for another function built from it.

GGHW [32] show that if what they call "special purpose" obfuscation is possible then full diO (diO for all circuits relative to arbitrary auxiliary inputs) is not possible. (Their result constructs a pathological auxiliary input that is itself a special-purpose obfuscated circuit.) Their result does not rule out the assumptions we use, namely iO or diO$^-$. GGHW go on to show that if a special-purpose obfuscation of Turing Machines is possible then there is an artificial one-way function with exponential pre-image size (in particular it is not injective) for which there is no hardcore predicate that is output-dependent. (That is, $f(x_1) = f(x_2)$ implies $h(x_1) = h(x_2)$, a property possessed by our second construction.)

Subsequent work. Our proof for **HC1** avoids hardwiring the challenge input x^* in the circuit by hardwiring $y^* = f(x^*)$ and changing the test for $x = x^*$ to $f(x) = y^*$. Brzuska and Mittelbach [23] change this step in our proof to implement the test using auxiliary-input point function obfuscation (AIPO) [23, 24, 14]. As a result, under the additional assumption that AIPO is possible, they obtain a proof for our **HC1** construction which applies to arbitrary (not just injective) one-way functions. Zhandry [59] shows how to replace diO in our constructions with extractable witness PRFs.

2 Preliminaries

We recall definitions for one-way functions, hardcore predicates and punctured PRFs.

Notation. We denote by $\lambda \in \mathbb{N}$ the security parameter and by 1^λ its unary representation. We denote the size of a finite set X by $|X|$, and the length of a string $x \in \{0,1\}^*$ by $|x|$. We let ε denote the empty string. If C is a circuit then $|\mathrm{C}|$ denotes its size, and if s is an integer then $\mathsf{Pad}_s(\mathrm{C})$ denotes C padded to have size s. If X is a finite set, we let $x \leftarrow\$ X$ denote picking an element of X uniformly at random and assigning it to x. Algorithms may be randomized unless otherwise indicated. Running time is worst case. "PT" stands for "polynomial-time," whether for randomized algorithms or deterministic ones. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with random coins r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \leftarrow\$ A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$. We say that $f : \mathbb{N} \to \mathbb{R}$ is negligible if for every positive polynomial p, there exists $n_p \in \mathbb{N}$ such that $f(n) < 1/p(n)$ for all $n > n_p$. We use the code based game playing framework of [11]. (See Fig. 2 for examples of games.) By $\mathrm{G}^{\mathcal{A}}(\lambda)$ we denote the event that the execution of game G with adversary $\mathcal{A}$ and security parameter λ results in the game returning true.

Function families. A family of functions F specifies the following. PT key generation algorithm $\mathsf{F.Kg}$ takes 1^λ and possibly another input to return a key $fk \in \{0,1\}^{\mathsf{F.kl}(\lambda)}$, where $\mathsf{F.kl}\colon \mathbb{N} \to \mathbb{N}$ is the key length function associated to F. Deterministic, PT evaluation algorithm $\mathsf{F.Ev}$ takes key fk and an input $x \in \{0,1\}^{\mathsf{F.il}(\lambda)}$ to return an output $\mathsf{F.Ev}(fk, x) \in \{0,1\}^{\mathsf{F.ol}(\lambda)}$, where $\mathsf{F.il}, \mathsf{F.ol}\colon \mathbb{N} \to \mathbb{N}$ are the input and output length functions associated to F, respectively. The pre-image size of F is the function $\textsc{PreImg}_{\mathsf{F}}$ defined for $\lambda \in \mathbb{N}$ by

$$\textsc{PreImg}_{\mathsf{F}}(\lambda) = \max_{fk, x^*} \left| \{\, x \in \{0,1\}^{\mathsf{F.il}(\lambda)} \;:\; \mathsf{F.Ev}(fk, x) = \mathsf{F.Ev}(fk, x^*) \,\} \right|$$

```
Game $\mathrm{OW}_{\mathsf{F}}^{\mathcal{F}}(\lambda)$
$fk \leftarrow\!\!\$\ \mathsf{F.Kg}(1^\lambda)$
$x^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.il}(\lambda)}$
$y^* \leftarrow \mathsf{F.Ev}(fk, x^*)$
$x' \leftarrow\!\!\$\ \mathcal{F}(1^\lambda, fk, y^*)$
Return $(y^* = \mathsf{F.Ev}(fk, x'))$
```

```
Game $\mathrm{HC}_{\mathsf{F,H}}^{\mathcal{H}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$
$fk \leftarrow\!\!\$\ \mathsf{F.Kg}(1^\lambda)$
$hk \leftarrow\!\!\$\ \mathsf{H.Kg}(1^\lambda, fk)$
$x^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.il}(\lambda)}$
$y^* \leftarrow \mathsf{F.Ev}(fk, x^*)$
If $b = 1$ then
    $r^* \leftarrow \mathsf{H.Ev}(hk, x^*)$
Else $r^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{H.ol}(\lambda)}$
$b' \leftarrow\!\!\$\ \mathcal{H}(1^\lambda, fk, hk, y^*, r^*)$
Return $(b = b')$
```

```
Game $\mathrm{PPRF}_{\mathsf{G}}^{\mathcal{G}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$
$gk \leftarrow\!\!\$\ \mathsf{G.Kg}(1^\lambda)$
$b' \leftarrow\!\!\$\ \mathcal{G}^{\mathrm{CH}}(1^\lambda)$
Return $(b = b')$

$\mathrm{CH}(x^*)$
$gk^* \leftarrow\!\!\$\ \mathsf{G.PKg}(1^\lambda, gk, x^*)$
If $b = 1$ then
    $r^* \leftarrow \mathsf{G.Ev}(gk, x^*)$
Else $r^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{G.ol}(\lambda)}$
Return $(gk^*, r^*)$
```

```
Game $\mathrm{DIFF}_{\mathcal{S}}^{\mathcal{D}}(\lambda)$
$(\mathrm{C}_0, \mathrm{C}_1, aux) \leftarrow\!\!\$\ \mathcal{S}(1^\lambda)$
$x \leftarrow\!\!\$\ \mathcal{D}(\mathrm{C}_0, \mathrm{C}_1, aux)$
Return $(\mathrm{C}_0(x) \neq \mathrm{C}_1(x))$
```

```
Game $\mathrm{IO}_{\mathsf{Obf},\mathcal{S}}^{\mathcal{O}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\}$ ; $(\mathrm{C}_0, \mathrm{C}_1, aux) \leftarrow\!\!\$\ \mathcal{S}(1^\lambda)$
$\overline{\mathrm{C}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathrm{C}_b)$ ; $b' \leftarrow\!\!\$\ \mathcal{O}(1^\lambda, \overline{\mathrm{C}}, aux)$
Return $(b = b')$
```

Fig. 2. Games defining one-wayness of F, security of H as a hardcore function for F, punctured-PRF security of G, difference-security of circuit sampler $\mathcal{S}$ and iO-security of obfuscator Obf relative to circuit sampler $\mathcal{S}$.

where the maximum is over all $x^* \in \{0,1\}^{\mathsf{F.il}(\lambda)}$ and all $fk \in [\mathsf{F.Kg}(1^\lambda)]$. We say that F is injective if $\mathrm{PREIMG}_{\mathsf{F}}(\lambda) = 1$ for all $\lambda \in \mathbb{N}$, meaning $\mathsf{F.Ev}(fk, x_1) \neq \mathsf{F.Ev}(fk, x_2)$ for all distinct $x_1, x_2 \in \{0,1\}^{\mathsf{F.il}(\lambda)}$, all fk and all $\lambda \in \mathbb{N}$. We say that F has polynomial pre-image size if there is a polynomial p such that $\mathrm{PREIMG}_{\mathsf{F}}(\cdot) \leq p(\cdot)$.

ONE-WAYNESS AND HARDCORE FUNCTIONS. Function family F is one-way if $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathcal{F}}(\cdot)$ is negligible for all PT adversaries $\mathcal{F}$, where $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathcal{F}}(\lambda) = \Pr[\mathrm{OW}_{\mathsf{F}}^{\mathcal{F}}(\lambda)]$ and game $\mathrm{OW}_{\mathsf{F}}^{\mathcal{F}}(\lambda)$ is defined in Fig. 2. Let H be a family of functions with $\mathsf{H.il} = \mathsf{F.il}$. We say that H is hardcore for F if $\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F,H},\mathcal{H}}(\cdot)$ is negligible for all PT adversaries $\mathcal{H}$, where $\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F,H},\mathcal{H}}(\lambda) = 2\Pr[\mathrm{HC}_{\mathsf{F,H}}^{\mathcal{H}}(\lambda)] - 1$ and game $\mathrm{HC}_{\mathsf{F,H}}^{\mathcal{H}}(\lambda)$ is defined in Fig. 2.

PUNCTURED PRFS. A punctured function family G specifies (beyond the usual algorithms) additional PT algorithms $\mathsf{G.PKg}, \mathsf{G.PEv}$. On inputs 1^λ, a key $gk \in [\mathsf{G.Kg}(1^\lambda)]$ and target input $x^* \in \{0,1\}^{\mathsf{G.il}(\lambda)}$, algorithm $\mathsf{G.PKg}$ returns a "punctured" key gk^* such that $\mathsf{G.PEv}(gk^*, x) = \mathsf{G.Ev}(gk, x)$ for all $x \in \{0,1\}^{\mathsf{G.il}(\lambda)} \setminus \{x^*\}$. We say that G is a punctured PRF if $\mathsf{Adv}^{\mathsf{pprf}}_{\mathsf{G},\mathcal{G}}(\cdot)$ is negligible for all PT adversaries $\mathcal{G}$, where $\mathsf{Adv}^{\mathsf{pprf}}_{\mathsf{G},\mathcal{G}}(\lambda) = 2\Pr[\mathrm{PPRF}_{\mathsf{G}}^{\mathcal{G}}(\lambda)] - 1$ and game $\mathrm{PPRF}_{\mathsf{G}}^{\mathcal{G}}(\lambda)$ is defined in Fig. 2. Here $\mathcal{G}$ must make exactly one oracle query where it picks a target point x^* and gets back the corresponding punctured key together with a challenge for the value of $\mathsf{G.Ev}$ on the target point.

The concept of punctured PRFs is due to [19, 43, 21] who note that they can be built via the GGM construction [34]. This however yields a family G with $\mathsf{G.il} = \mathsf{G.ol}$. For our purposes, we need a stronger result, namely a punctured PRF with arbitrary polynomial output length:

Proposition 1. *Let ι, ℓ be polynomials and assume one-way functions exist. Then there is a punctured PRF G with $\mathsf{G.il} = \iota$ and $\mathsf{G.ol} = \ell$.*

The claimed punctured PRF G can be obtained by starting from a GGM-based punctured PRF $\overline{\mathsf{G}}$ with $\overline{\mathsf{G}}\mathsf{.il} = \overline{\mathsf{G}}\mathsf{.ol} = \iota$ and letting $\mathsf{G.Ev}(gk, x) = \mathsf{S.Ev}(\overline{\mathsf{G}}\mathsf{.Ev}(gk, x))$ where S is a PRG with input length ι and output length ℓ. We omit the details.

3 Parameterized diO framework

We present a definitional framework for diO where security is parameterized by a class of circuit samplers. This is of conceptual value in enabling us to capture and unify existing forms of iO and diO. Further, it allows one to easily define new forms of diO that are *weaker* than the full auxiliary-input diO of [20, 4] and thus obtain results under weaker assumptions. Our parameterized language leads to sharper and more fine-grained security claims in which we can state assumptions that are closer to what is actually used by the proof rather than being overkill, in particular with regard to what types of auxiliary input are used [13, 22, 32]. This allows us to circumvent the negative results of [32] which apply to diO with arbitrary auxiliary input. We note that previous definitions did parameterize the definition by a class of circuits but this is different and in particular will not capture differences related to auxiliary inputs.

Circuit samplers. A circuit sampler is a PT algorithm $\mathcal{S}$ that on input 1^λ returns a triple $(\mathrm{C}_0, \mathrm{C}_1, aux)$ where $\mathrm{C}_0, \mathrm{C}_1$ are circuits which have the same size, number of inputs and number of outputs, and aux is a string. We say that a circuit sampler $\mathcal{S}$ is difference secure if $\mathsf{Adv}^{\mathsf{diff}}_{\mathcal{S},\mathcal{D}}(\cdot)$ is negligible for every PT adversary $\mathcal{D}$, where $\mathsf{Adv}^{\mathsf{diff}}_{\mathcal{S},\mathcal{D}}(\lambda) = \Pr[\mathrm{DIFF}^{\mathcal{D}}_{\mathcal{S}}(\lambda)]$ and game $\mathrm{DIFF}^{\mathcal{D}}_{\mathcal{S}}(\lambda)$ is defined in Fig. 2. Difference security of $\mathcal{S}$ means that given $\mathrm{C}_0, \mathrm{C}_1, aux$ it is hard to find an input on which the circuits differ.

iO-security. An obfuscator is a PT algorithm Obf that on input 1^λ and a circuit C returns a circuit $\overline{\mathrm{C}}$ such that $\overline{\mathrm{C}}(x) = \mathrm{C}(x)$ for all x. If $\mathcal{S}$ is a circuit sampler and $\mathcal{O}$ is an adversary, we let $\mathsf{Adv}^{\mathsf{io}}_{\mathsf{Obf},\mathcal{S},\mathcal{O}}(\lambda) = 2\Pr[\mathrm{IO}^{\mathcal{O}}_{\mathsf{Obf},\mathcal{S}}(\lambda)] - 1$ where game $\mathrm{IO}^{\mathcal{O}}_{\mathsf{Obf},\mathcal{S}}(\lambda)$ is defined in Fig. 2. Now let $\boldsymbol{S}$ be a class (set) of circuit samplers. We say that Obf is $\boldsymbol{S}$-secure if $\mathsf{Adv}^{\mathsf{io}}_{\mathsf{Obf},\mathcal{S},\mathcal{O}}(\cdot)$ is negligible for every PT adversary $\mathcal{O}$ and every circuit sampler $\mathcal{S} \in \boldsymbol{S}$. This is our parameterized notion of security. The following obvious fact will often be useful:

Proposition 2. *Let $\boldsymbol{S}_1, \boldsymbol{S}_2$ be classes of circuit samplers and Obf an obfuscator. Suppose $\boldsymbol{S}_1 \subseteq \boldsymbol{S}_2$. Then if Obf is $\boldsymbol{S}_2$-secure it is also $\boldsymbol{S}_1$-secure.*

Capturing known notions. Different types of iO security in the literature can now be captured and unified by considering different classes of circuit samplers, as follows.

Let $\boldsymbol{S}_{\text{diff}}$ be the class of all difference-secure circuit samplers. Then Obf being $\boldsymbol{S}_{\text{diff}}$-secure means it is a differing-inputs obfuscator as per [20, 4].

Let $\boldsymbol{S}^{\overline{\text{aux}}}$ be the class of circuit samplers $\mathcal{S}$ that do not have auxiliary inputs, meaning $aux = \varepsilon$ for all $\lambda \in \mathbb{N}$ and all $(\mathrm{C}_0, \mathrm{C}_1, aux) \in [\mathcal{S}(1^\lambda)]$. Let $\boldsymbol{S}_{\text{diff}}^{\overline{\text{aux}}} = \boldsymbol{S}_{\text{diff}} \cap \boldsymbol{S}^{\overline{\text{aux}}} \subseteq \boldsymbol{S}_{\text{diff}}$ be the class of all difference-secure circuit samplers that do not have auxiliary inputs. Then Obf being $\boldsymbol{S}_{\text{diff}}^{\overline{\text{aux}}}$-secure means it is a differing-inputs obfuscator as per [5].

We say that circuits $\mathrm{C}_0, \mathrm{C}_1$ are equivalent, written $\mathrm{C}_0 \equiv \mathrm{C}_1$, if they agree on all inputs. We say that circuit sampler $\mathcal{S}$ produces equivalent circuits if $\mathrm{C}_0 \equiv \mathrm{C}_1$ for all $\lambda \in \mathbb{N}$ and all $(\mathrm{C}_0, \mathrm{C}_1, aux) \in [\mathcal{S}(1^\lambda)]$. Let $\boldsymbol{S}_{\text{eq}}$ be the class of all circuit samplers that produce equivalent circuits. Then Obf being $\boldsymbol{S}_{\text{eq}}$-secure means it is an indistinguishability obfuscator as per [53]. Let $\boldsymbol{S}_{\text{eq}}^{\overline{\text{aux}}} = \boldsymbol{S}_{\text{eq}} \cap \boldsymbol{S}^{\overline{\text{aux}}}$ be the class of circuit samplers without auxiliary inputs that produce equivalent circuits. Then Obf being $\boldsymbol{S}_{\text{eq}}^{\overline{\text{aux}}}$-secure means it is an indistinguishability obfuscator as per [5].

If $\mathcal{S}$ produces equivalent circuits it is certainly difference-secure. This means that $\boldsymbol{S}_{\text{eq}} \subseteq \boldsymbol{S}_{\text{diff}}$ and $\boldsymbol{S}_{\text{eq}}^{\overline{\text{aux}}} \subseteq \boldsymbol{S}_{\text{diff}}^{\overline{\text{aux}}}$. Hence Proposition 2 says that any $\boldsymbol{S}_{\text{diff}}$-secure obfuscator is a $\boldsymbol{S}_{\text{eq}}$-secure obfuscator and any $\boldsymbol{S}_{\text{diff}}^{\overline{\text{aux}}}$-secure obfuscator is a $\boldsymbol{S}_{\text{eq}}^{\overline{\text{aux}}}$-secure obfuscator. That is, diO implies iO, both for the case with auxiliary input and the case without, a fact we will often use.

We say that circuit sampler $\mathcal{S}$ produces d-differing circuits, where $d\colon \mathbb{N} \to \mathbb{N}$, if C_0 and C_1 differ on at most $d(\lambda)$ inputs for all $\lambda \in \mathbb{N}$ and all $(\mathrm{C}_0, \mathrm{C}_1, aux) \in [\mathcal{S}(1^\lambda)]$. Let $\boldsymbol{S}_{\text{diff}}(d)$ be the class of all difference-secure circuit samplers that produce d-differing circuits, so that $\boldsymbol{S}_{\text{eq}} \subseteq \boldsymbol{S}_{\text{diff}}(d) \subseteq \boldsymbol{S}_{\text{diff}}$. The interest of this definition is the following result of BCP [20]:

Proposition 3. *If d is a polynomial then any $\boldsymbol{S}_{\text{eq}}$-secure obfuscator is also a $\boldsymbol{S}_{\text{diff}}(d)$-secure obfuscator.*

We will exploit this to reduce our assumptions from $\boldsymbol{S}_{\text{diff}}$-secure obfuscation to $\boldsymbol{S}_{\text{eq}}$-secure obfuscation in some cases.

New classes. Above, we have used our framework to express and capture existing variants of iO and diO. We now define a new variant, via a new class of samplers. Following the definition we will explain the motivation. We say that a circuit sampler $\mathcal{S}$ has short auxiliary inputs if $|aux| < |\mathrm{C}_b|$ for all $b \in \{0,1\}$, all $\lambda \in \mathbb{N}$ and all $(\mathrm{C}_0, \mathrm{C}_1, aux) \in [\mathcal{S}(1^\lambda)]$. We let $\boldsymbol{S}^{\text{sh}}$ be the class of all circuit samplers that have short auxiliary inputs. The assumption made in Theorem 6 is a $\boldsymbol{S}$-secure obfuscator for a particular $\boldsymbol{S} \subseteq \boldsymbol{S}_{\text{diff}} \cap \boldsymbol{S}^{\text{sh}}$, meaning diO is only required relative to circuits samplers with short auxiliary inputs. This is a potentially *weaker* assumption than a $\boldsymbol{S}_{\text{diff}}$-secure obfuscator.

4 Poly-many hardcore bits for injective OWFs

In this section we consider the natural construction of a hardcore function with arbitrary span, namely an obfuscated PRF. We show that this works assuming the one-way function is injective and the obfuscation is diO-secure, yielding our first result, namely a hardcore function with arbitrary polynomial span for any injective one-way function.

Construction. Let G be a function family. Let Obf be an obfuscator and let $s\colon \mathbb{N} \to \mathbb{N}$. We define function family $\mathsf{H} = \mathbf{HC1}[\mathsf{G}, \mathsf{Obf}, s]$ as follows, with $\mathsf{H.il} = \mathsf{G.il}$ and $\mathsf{H.ol} = \mathsf{G.ol}$:

```
$\underline{\mathsf{H.Kg}(1^\lambda, fk)}$
$gk \leftarrow\$ \mathsf{G.Kg}(1^\lambda)$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$
$\overline{\mathrm{C}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathrm{C})$
$hk \leftarrow \overline{\mathrm{C}}$ ; Return $hk$

$\underline{\mathsf{H.Ev}(hk, x)}$
$\overline{\mathrm{C}} \leftarrow hk$ ; $r \leftarrow \overline{\mathrm{C}}(x)$
Return $r$
```

We give $\mathsf{H.Kg}$ two inputs because this is required by the syntax, but the second is ignored. Here $\mathsf{G.Ev}(gk, \cdot)$ represents the circuit that given x returns $\mathsf{G.Ev}(gk, x)$, and C is obtained by padding $\mathsf{G.Ev}(gk, \cdot)$ to size $s(\lambda)$. The padding length function s is a parameter of the construction that will depend on the one-way function F for which H will be hardcore. The description hk of the hardcore function is an obfuscation of circuit C. The hardcore function output is the result of this obfuscated circuit on x, which is simply $\mathsf{G.Ev}(gk, x)$, the result of the original circuit on x. The output of the hardcore function is thus the output of a PRF, the key for the latter embedded in an obfuscated circuit to prevent its being revealed.

Results. Recall that a $\boldsymbol{S}_{\text{diff}}(1)$-secure obfuscator is weaker than a full $\boldsymbol{S}_{\text{diff}}$-secure obfuscator (see Section 3 for definitions) since it is only required to work on circuits that differ on at most one (hard to find) input. We have:

Theorem 4. *Let F be an injective one-way function family. Let G be a punctured PRF with $\mathsf{G.il} = \mathsf{F.il}$. Then there is a polynomial s such that the following is true. Let Obf be any $\boldsymbol{S}_{\text{diff}}(1)$-secure obfuscator. Then the function family $\mathsf{H} = \mathbf{HC1}[\mathsf{G}, \mathsf{Obf}, s]$ defined above is hardcore for F.*

Proposition 1 yields punctured PRFs with as long an output as desired, so that the above allows extraction of an arbitrary polynomial number of hardcore bits. The one-way function assumption made in Proposition 1 is already implied by the assumption that F is one way. Theorem 4 assumes a differing-inputs obfuscator only for circuits that differ on at most one input, which by Proposition 3 can be obtained from an indistinguishability obfuscator, making the latter the only assumption beyond one-wayness of F needed to extract polynomially-many hardcore bits. Formally, we have:

Corollary 5. *Let ℓ be any polynomial. Let F be any injective one-way function. If there exists a $\boldsymbol{S}_{\text{eq}}$-secure obfuscator then there exists a hardcore function H for F with $\mathsf{H.ol} = \ell$.*

```
Games G_0–G_4
$fk \leftarrow\!\!\$\ \mathsf{F.Kg}(1^\lambda)$ ; $gk \leftarrow\!\!\$\ \mathsf{G.Kg}(1^\lambda)$ ; $x^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.il}(\lambda)}$
$y^* \leftarrow \mathsf{F.Ev}(fk, x^*)$ ; $gk^* \leftarrow\!\!\$\ \mathsf{G.PKg}(1^\lambda, gk, x^*)$
    $r^* \leftarrow \mathsf{G.Ev}(gk, x^*)$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$        // $\mathrm{G}_0$
    $r^* \leftarrow \mathsf{G.Ev}(gk, x^*)$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^1_{gk^*,x^*,r^*})$        // $\mathrm{G}_1$
    $r^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{G.ol}(\lambda)}$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^1_{gk^*,x^*,r^*})$        // $\mathrm{G}_2$
    $r^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{G.ol}(\lambda)}$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^2_{fk,gk,y^*,r^*})$        // $\mathrm{G}_3$
    $r^* \leftarrow\!\!\$\ \{0,1\}^{\mathsf{G.ol}(\lambda)}$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$        // $\mathrm{G}_4$
$\overline{\mathrm{C}} \leftarrow\!\!\$\ \mathsf{Obf}(1^\lambda, \mathrm{C})$ ; $hk \leftarrow \overline{\mathrm{C}}$ ; $b' \leftarrow\!\!\$\ \mathcal{H}(1^\lambda, fk, hk, y^*, r^*)$ ; Return $(b' = 1)$

Circuit $\mathrm{C}^1_{gk^*,x^*,r^*}(x)$
If $x \neq x^*$ then return $\mathsf{G.PEv}(gk^*, x)$
Else return $r^*$

Circuit $\mathrm{C}^2_{fk,gk,y^*,r^*}(x)$
If $\mathsf{F.Ev}(fk, x) \neq y^*$ then return $\mathsf{G.Ev}(gk, x)$
Else return $r^*$
```

Fig. 3. Games for proof of Theorem 4.

Proof (Theorem 4). We define s as follows: For any $\lambda \in \mathbb{N}$ let $s(\lambda)$ be a polynomial upper bound on $\max(|\mathsf{G.Ev}(gk,\cdot)|, |\mathrm{C}^1_{gk^*,x^*,r^*}|, |\mathrm{C}^2_{fk,gk,y^*,r^*}|)$ where the last two circuits are in Fig. 3 and the maximum is over all $gk \in [\mathsf{G.Kg}(1^\lambda)]$, $x^* \in \{0,1\}^{\mathsf{G.il}(\lambda)}$, $gk^* \in [\mathsf{G.PKg}(1^\lambda, gk, x^*)]$, $fk \in [\mathsf{F.Kg}(1^\lambda)]$, $y^* \in \{0,1\}^{\mathsf{F.ol}(\lambda)}$ and $r^* \in \{0,1\}^{\mathsf{G.ol}(\lambda)}$. Now let $\mathcal{H}$ be a PT adversary. Consider the games and associated circuits of Fig. 3. Lines not annotated with comments are common to all five games. The games begin by picking keys fk, gk for the one-way function F and the punctured-PRF G, respectively. They pick the challenge input x^* and then create a punctured PRF key gk^* for it. Then the games differ in how they define r^* and the circuit C to be obuscated. These defined, the games re-unite to obfuscate the circuit and run $\mathcal{H}$.

Game G_0 does not use the punctured keys, and is equivalent to the $b = 1$ case of $\mathrm{HC}^{\mathcal{H}}_{\mathsf{F,H}}(\lambda)$ while G_4, similarly, is its $b = 0$ case, so

$$\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F,H},\mathcal{H}}(\lambda) = \Pr[\mathrm{G}_0] - \Pr[\mathrm{G}_4] \ . \tag{1}$$

We now show that $\Pr[\mathrm{G}_{i-1}] - \Pr[\mathrm{G}_i]$ is negligible for $i = 1, 2, 3, 4$, which by Equation (1) implies that $\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F,H},\mathcal{H}}(\cdot)$ is negligible and proves the theorem. We begin with some intuition. In game G_1, we switch the circuit being obfuscated to one that uses the punctured key when $x \neq x^*$ and returns an embedded $r^* = \mathsf{G.Ev}(gk, x^*)$ otherwise. This circuit is equivalent to $\mathsf{G.Ev}(gk, \cdot)$ so iO-security, implied by diO, will tell us that the adversary $\mathcal{H}$ will hardly notice. This switch puts us in position to use the security of the punctured-PRF, based on which G_2 replaces r^* with a random value. These steps are direct applications of the Sahai-Waters technique [53], but now things get more difficult. Having made r^* random is not enough because we must now revert the circuit being obfuscated back to $\mathsf{G.Ev}(gk, \cdot)$. We also realize that we have not yet used the one-wayness of F, so this reversion must rely on it. But the circuit in G_2 embeds x^*, the point a one-wayness adversary would be trying to find, and it is not clear how we can simulate the construction of this circuit in the design of an adversary violating one-wayness. To address this, instead of testing whether x equals x^*, the circuit in G_3 tests whether the values of $\mathsf{F.Ev}(fk, \cdot)$ on these points agree, which can be done given only $y^* = \mathsf{F.Ev}(fk, x^*)$ and avoids putting x^* in the circuit. But we need this to not alter the functionality of the circuit. This is where we make crucial use of the assumption that $\mathsf{F.Ev}(gk, \cdot)$ is injective. Finally, in game G_4 we revert the circuit back to $\mathsf{G.Ev}(gk, \cdot)$. But the circuits in $\mathrm{G}_3, \mathrm{G}_4$ are now manifestedly *not* equivalent. However, the input on which they differ is x^*. We show that the one-wayness of F implies that this point is hard to find, whence diO (here iO is not enough) allows us to conclude. We now proceed to the details.

Below, we (simultaneously) define three circuit samplers that differ at the commented lines and have the uncommented lines in common, and then also define an iO-adversary:

```
Circuit Samplers $\mathcal{S}_1(1^\lambda)$, $\mathcal{S}_3(1^\lambda)$, $\mathcal{S}_4(1^\lambda)$
$fk \leftarrow\$ \mathsf{F.Kg}(1^\lambda)$ ; $gk \leftarrow\$ \mathsf{G.Kg}(1^\lambda)$
$x^* \leftarrow\$ \{0,1\}^{\mathsf{F.il}(\lambda)}$ ; $y^* \leftarrow \mathsf{F.Ev}(fk, x^*)$ ; $gk^* \leftarrow\$ \mathsf{G.PKg}(1^\lambda, gk, x^*)$
$r^* \leftarrow \mathsf{G.Ev}(gk, x^*)$ ; $\mathrm{C}_1 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$ ; $\mathrm{C}_0 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^1_{gk^*,x^*,r^*})$   // $\mathcal{S}_1$
$r^* \leftarrow\$ \{0,1\}^{\mathsf{G.ol}(\lambda)}$ ; $\mathrm{C}_1 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^1_{gk^*,x^*,r^*})$ ; $\mathrm{C}_0 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^2_{fk,gk,y^*,r^*})$   // $\mathcal{S}_3$
$r^* \leftarrow\$ \{0,1\}^{\mathsf{G.ol}(\lambda)}$ ; $\mathrm{C}_1 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^2_{fk,gk,y^*,r^*})$ ; $\mathrm{C}_0 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$   // $\mathcal{S}_4$
$aux \leftarrow (fk, y^*, r^*)$ ; Return $(\mathrm{C}_0, \mathrm{C}_1, aux)$
```

```
Adversary $\mathcal{O}(1^\lambda, \overline{\mathrm{C}}, aux)$
$(fk, y^*, r^*) \leftarrow aux$ ; $hk \leftarrow \overline{\mathrm{C}}$ ; $b' \leftarrow\$ \mathcal{H}(1^\lambda, fk, hk, y^*, r^*)$
Return $b'$
```

Now we have

$$\Pr[\mathrm{G}_{i-1}] - \Pr[\mathrm{G}_i] = \mathsf{Adv}^{\mathsf{io}}_{\mathsf{Obf},\mathcal{S}_i,\mathcal{O}}(\lambda) \qquad \text{for } i \in \{1,3,4\} \ . \tag{2}$$

We now make three claims: (1) $\mathcal{S}_1 \in \boldsymbol{S}_{\mathrm{eq}}$ (2) $\mathcal{S}_3 \in \boldsymbol{S}_{\mathrm{eq}}$ (3) $\mathcal{S}_4 \in \boldsymbol{S}_{\mathrm{diff}}(1)$. Since $\boldsymbol{S}_{\mathrm{eq}} \subseteq \boldsymbol{S}_{\mathrm{diff}}(1)$ and Obf is assumed $\boldsymbol{S}_{\mathrm{diff}}(1)$-secure, the RHS of Equation (2) is negligible in all three cases.

We now establish claim (1). If $x \neq x^*$ then $\mathrm{C}^1_{gk^*,x^*,r^*}(x) = \mathsf{G.PEv}(gk^*, x) = \mathsf{G.Ev}(gk, x)$. If $x = x^*$ then $\mathrm{C}^1_{gk^*,x^*,r^*}(x) = r^*$, but $\mathcal{S}_1$ sets $r^* = \mathsf{G.Ev}(gk, x^*)$. This means that $\mathcal{S}_1$ produces equivalent circuits, and hence $\mathcal{S}_1 \in \boldsymbol{S}_{\mathrm{eq}}$.

Next we establish claim (2). The assumed injectivity of F implies that circuits $\mathrm{C}^1_{gk^*,x^*,r^*}$ and $\mathrm{C}^2_{fk,gk,y^*,r^*}$ are equivalent when $y^* = \mathsf{F.Ev}(fk, x^*)$, and hence $\mathcal{S}_3 \in \boldsymbol{S}_{\mathrm{eq}}$.

To establish claim (3), given any PT difference adversary $\mathcal{D}$ for $\mathcal{S}_4$, we build one-wayness adversary $\mathcal{F}$ via

```
Adversary $\mathcal{F}(1^\lambda, fk, y^*)$
$gk \leftarrow\$ \mathsf{G.Kg}(1^\lambda)$ ; $r^* \leftarrow\$ \{0,1\}^{\mathsf{G.ol}(\lambda)}$
$\mathrm{C}_1 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^2_{fk,gk,y^*,r^*})$ ; $\mathrm{C}_0 \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \cdot))$
$aux \leftarrow (fk, y^*, r^*)$ ; $x \leftarrow\$ \mathcal{D}(\mathrm{C}_0, \mathrm{C}_1, aux)$ ; Return $x$
```

If $\mathrm{C}_1(x) \neq \mathrm{C}_0(x)$ then it must be that $\mathsf{F.Ev}(fk, x) = y^*$. Thus $\mathsf{Adv}^{\mathsf{diff}}_{\mathcal{S}_4,\mathcal{D}}(\cdot) \leq \mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathcal{F}}(\cdot)$. The assumed one-wayness of F thus means that $\mathcal{S}_4$ is difference-secure. But we also observe that, due to the injectivity of F, circuits $\mathrm{C}_0, \mathrm{C}_1$ differ on only one input, namely x^*. So $\mathcal{S}_4 \in \boldsymbol{S}_{\mathrm{diff}}(1)$.

One transition remains, namely that from G_1 to G_2. Here we have

$$\Pr[\mathrm{G}_1] - \Pr[\mathrm{G}_2] = \mathsf{Adv}^{\mathsf{pprf}}_{\mathsf{G},\mathcal{G}}(\lambda) \tag{3}$$

where adversary $\mathcal{G}$ is defined via

```
Adversary $\mathcal{G}^{\mathrm{CH}}(1^\lambda)$
$fk \leftarrow\$ \mathsf{F.Kg}(1^\lambda)$ ; $x^* \leftarrow\$ \{0,1\}^{\mathsf{F.il}(\lambda)}$ ; $y^* \leftarrow \mathsf{F.Ev}(fk, x^*)$ ; $(gk^*, r^*) \leftarrow\$ \mathrm{CH}(x^*)$
$\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathrm{C}^1_{gk^*,x^*,r^*})$ ; $hk \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathrm{C})$ ; $b' \leftarrow\$ \mathcal{H}(1^\lambda, fk, hk, y^*, r^*)$
Return $b'$
```

The RHS of Equation (3) is negligible by the assumption that G is a punctured PRF. This concludes the proof.

5 Poly-many hardcore bits for any OWF

The proof of Theorem 4 makes crucial use of the assumed injectivity of F. To remove this assumption, we modify the construction so that the obfuscated PRF is applied, not to x, but to the result of the one-way function on x.

Construction. Let F, G be function families with $\mathsf{G.il} = \mathsf{F.ol}$. Let Obf be an obfuscator and let $s\colon \mathbb{N} \to \mathbb{N}$. We define function family $\mathsf{H} = \mathbf{HC2}[\mathsf{F}, \mathsf{G}, \mathsf{Obf}, s]$ as follows, with $\mathsf{H.il} = \mathsf{F.il}$ and $\mathsf{H.ol} = \mathsf{G.ol}$:

```
$\mathsf{H.Kg}(1^\lambda, fk)$
$gk \leftarrow\$ \mathsf{G.Kg}(1^\lambda)$ ; $\mathrm{C} \leftarrow \mathsf{Pad}_{s(\lambda)}(\mathsf{G.Ev}(gk, \mathsf{F.Ev}(fk, \cdot)))$
$\overline{\mathrm{C}} \leftarrow\$ \mathsf{Obf}(1^\lambda, \mathrm{C})$
$hk \leftarrow \overline{\mathrm{C}}$ ; Return $hk$

$\mathsf{H.Ev}(hk, x)$
$\overline{\mathrm{C}} \leftarrow hk$ ; $r \leftarrow \overline{\mathrm{C}}(x)$
Return $r$
```

This time the result of the hardcore function output on input x is $\mathsf{G.Ev}(gk, y)$ where $y = \mathsf{F.Ev}(fk, x)$.

Results. The following says that $\boldsymbol{S}_{\mathrm{diff}}$-security of the obfuscator suffices for the security of the above hardcore function, but that in fact the assumption is weaker, the circuit samplers for which security is required being further restricted to have short auxiliary input as captured and to produce circuits that differ at a number of points bounded by the pre-image size d of the one-way function, formally $(\boldsymbol{S}_{\mathrm{diff}}(d) \cap \boldsymbol{S}^{\mathrm{sh}})$-security, where the classes were defined in Section 3. The proof is in [12]:

Theorem 6. *Let* F *be a one-way function family. Let* G *be a punctured PRF with* $\mathsf{G.il} = \mathsf{F.ol}$. *Let* $d = \textsc{PreImg}_{\mathsf{F}}$. *Then there is a polynomial* s *such that the following is true. Let* $\boldsymbol{S} = \boldsymbol{S}_{\mathrm{diff}}(d) \cap \boldsymbol{S}^{\mathrm{sh}}$. *Let* Obf *be any* $\boldsymbol{S}$*-secure obfuscator. Then the function family* $\mathsf{H} = \mathbf{HC2}[\mathsf{F}, \mathsf{G}, \mathsf{Obf}, s]$ *defined above is hardcore for* F.

This means that in the most general case we have:

Corollary 7. *Let ℓ be any polynomial. Let* F *be any one-way function. If there exists a* $(\boldsymbol{S}_{\text{diff}} \cap \boldsymbol{S}^{\text{sh}})$*-secure obfuscator then there exists a hardcore function* H *for* F *with* $\mathsf{H.ol} = \ell$*.*

When the pre-image size of F is polynomial, however, we can again exploit Proposition 3 to obtain our conclusion assuming nothing beyond iO:

Corollary 8. *Let ℓ be any polynomial. Let* F *be any one-way function with polynomially-bounded pre-image size. If there exists a* $\boldsymbol{S}_{\text{eq}}$*-secure obfuscator then there exists a hardcore function* H *for* F *with* $\mathsf{H.ol} = \ell$*.*

6 Hardcore functions for correlated inputs

We show that our hardcore functions are able to extract random bits even on sequences of inputs that are arbitrarily correlated. Somewhat more precisely, draw a vector $\mathbf{x}$ from an arbitrary distribution, in particuclar allowing its components to be arbitrarily correlated. Then applying our hardcore function componentwise to $\mathbf{x}$ results in a vector whose components look random *and independent* even given the result of applying f componentwise to $\mathbf{x}$, making only the necessary assumption that f remains one-way on the distribution from which $\mathbf{x}$ was selected. This is an unusual property, not possessed by all constructions of hardcore functions. The ability to extract polynomially-many bits on correlated inputs leads to new constructions of deterministic PKE schemes.

Notation. We denote vectors by boldface lowercase letters, for example $\mathbf{x}$. If $\mathbf{x}$ is a vector then $|\mathbf{x}|$ denotes the number of components of $\mathbf{x}$ and $\mathbf{x}[i]$ denotes the i-th component of $\mathbf{x}$, for any $1 \leq i \leq |\mathbf{x}|$. We write $x \in \mathbf{x}$ to mean that $x = \mathbf{x}[i]$ for some $1 \leq i \leq |\mathbf{x}|$. If F is a family of functions, $\mathbf{x}$ is a vector over $\{0,1\}^{\mathsf{F.il}(\lambda)}$ and $fk \in \{0,1\}^{\mathsf{F.kl}(\lambda)}$, then we let $\mathsf{F.Ev}(fk, \mathbf{x}) = (\mathsf{F.Ev}(fk, \mathbf{x}[1]), \ldots, \mathsf{F.Ev}(fk, \mathbf{x}[|\mathbf{x}|]))$. Let $\mathbf{Rnd}$ denote the algorithm that on input a vector $\mathbf{x}$ and an integer ℓ returns a vector $\mathbf{r}$ of the same length as $\mathbf{x}$ whose entries are random ℓ-bit strings except that if two entries of $\mathbf{x}$ are the same, the same is true of the corresponding entries of $\mathbf{r}$. In detail, $\mathbf{Rnd}(\mathbf{x}, \ell)$ creates table T via: For $i = 1, \ldots, |\mathbf{x}|$ do: If not $T[\mathbf{x}[i]]$ then $T[\mathbf{x}[i]] \leftarrow_{\$} \{0,1\}^{\ell}$. Then it sets $\mathbf{r}[i] \leftarrow T[\mathbf{x}[i]]$ for $i = 1, \ldots, |\mathbf{x}|$ and returns the vector $\mathbf{r}$.

Definitions. An input sampler is an algorithm $\mathcal{I}$ that on input 1^λ returns a $\mathcal{I}.\mathsf{vl}(\lambda)$-vector of strings over $\{0,1\}^{\mathcal{I}.\mathsf{il}(\lambda)}$, where the vector-length $\mathcal{I}.\mathsf{vl}\colon \mathbb{N} \to \mathbb{N}$ and the input length $\mathcal{I}.\mathsf{il}\colon \mathbb{N} \to \mathbb{N}$ are polynomials associated to $\mathcal{I}$. We say that a function family F is one-way with respect to input sampler $\mathcal{I}$ if $\mathsf{F.il} = \mathcal{I}.\mathsf{il}$ and $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathcal{I},\mathcal{F}}(\cdot)$ is negligible for all PT adversaries $\mathcal{F}$, where $\mathsf{Adv}^{\mathsf{ow}}_{\mathsf{F},\mathcal{I},\mathcal{F}}(\lambda) = \Pr[\mathrm{OW}^{\mathcal{F}}_{\mathsf{F},\mathcal{I}}(\lambda) = 1]$ and game $\mathrm{OW}^{\mathcal{F}}_{\mathsf{F},\mathcal{I}}(\lambda)$ is defined in Fig. 4. We stress that for $\mathcal{F}$ to win in this game it needs to find the inverse under $\mathsf{F.Ev}(fk, \cdot)$ of *some* component of $\mathbf{y}^*$, not all components. Let H be a family of functions with $\mathsf{H.il} = \mathsf{F.il}$. We say that H is hardcore for F with respect to input sampler $\mathcal{I}$ if $\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F},\mathsf{H},\mathcal{I},\mathcal{H}}(\cdot)$

```
Game $\mathrm{OW}^{\mathcal{F}}_{\mathsf{F},\mathcal{I}}(\lambda)$
$fk \leftarrow\$ \mathsf{F.Kg}(1^\lambda)$
$\mathbf{x}^* \leftarrow\$ \mathcal{I}(1^\lambda)$
$\mathbf{y}^* \leftarrow \mathsf{F.Ev}(fk, \mathbf{x}^*)$
$x' \leftarrow\$ \mathcal{F}(1^\lambda, fk, \mathbf{y}^*)$
Return $(\mathsf{F.Ev}(fk, x') \in \mathbf{y}^*)$
```

```
Game $\mathrm{HC}^{\mathcal{H}}_{\mathsf{F},\mathsf{H},\mathcal{I}}(\lambda)$
$b \leftarrow\$ \{0,1\}$
$fk \leftarrow\$ \mathsf{F.Kg}(1^\lambda)$
$hk \leftarrow\$ \mathsf{H.Kg}(1^\lambda, fk)$
$\mathbf{x}^* \leftarrow\$ \mathcal{I}(1^\lambda)$
$\mathbf{y}^* \leftarrow \mathsf{F.Ev}(fk, \mathbf{x}^*)$
If $b = 1$ then
    $\mathbf{r}^* \leftarrow \mathsf{H.Ev}(hk, \mathbf{x}^*)$
Else
    $\mathbf{r}^* \leftarrow\$ \mathbf{Rnd}(\mathbf{x}^*, \mathsf{H.ol}(\lambda))$
$b' \leftarrow\$ \mathcal{H}(1^\lambda, fk, hk, \mathbf{y}^*, \mathbf{r}^*)$
Return $(b = b')$
```

```
Game $\mathrm{PPRF}^{\mathcal{G}}_{\mathsf{G}}(\lambda)$
$b \leftarrow\$ \{0,1\}$
$gk \leftarrow\$ \mathsf{G.Kg}(1^\lambda)$
$b' \leftarrow\$ \mathcal{G}^{\mathrm{CH}}(1^\lambda)$
Return $(b = b')$

$\mathrm{CH}(\mathbf{x}^*)$
$gk^* \leftarrow\$ \mathsf{G.PKg}(1^\lambda, gk, \mathbf{x}^*)$
If $b = 1$ then
    $\mathbf{r}^* \leftarrow \mathsf{G.Ev}(gk, \mathbf{x}^*)$
Else
    $\mathbf{r}^* \leftarrow\$ \mathbf{Rnd}(\mathbf{x}^*, \mathsf{G.ol}(\lambda))$
Return $(gk^*, \mathbf{r}^*)$
```

Fig. 4. Games defining one-wayness of F with respect to an input sampler $\mathcal{I}$, security of H as a hardcore function for F with respect to $\mathcal{I}$ and punctured-PRF security of G on multiple inputs.

is negligible for all PT adversaries $\mathcal{H}$, where $\mathsf{Adv}^{\mathsf{hc}}_{\mathsf{F},\mathsf{H},\mathcal{I},\mathcal{H}}(\lambda) = 2\Pr[\mathrm{HC}^{\mathcal{H}}_{\mathsf{F},\mathsf{H},\mathcal{I}}(\lambda)] - 1$ and game $\mathrm{HC}^{\mathcal{H}}_{\mathsf{F},\mathsf{H},\mathcal{I}}(\lambda)$ is defined in Fig. 4.

We extend punctured PRFs as defined in Section 2 to allow puncturing at multiple points. In a punctured function family G, algorithm $\mathsf{G.PKg}$ now takes 1^λ, a key $gk \in [\mathsf{G.Kg}(1^\lambda)]$ and a vector $\mathbf{x}^*$ over $\{0,1\}^{\mathsf{G.il}(\lambda)}$ (the target inputs) to return a "punctured" key gk^* such that $\mathsf{G.PEv}(gk^*, x) = \mathsf{G.Ev}(gk, x)$ for all $x \in \{0,1\}^{\mathsf{G.il}(\lambda)}$ such that $x \notin \mathbf{x}^*$. G is a punctured PRF if $\mathsf{Adv}^{\mathsf{pprf}}_{\mathsf{G},\mathcal{G}}(\cdot)$ is negligible for all PT adversaries $\mathcal{G}$, where $\mathsf{Adv}^{\mathsf{pprf}}_{\mathsf{G},\mathcal{G}}(\lambda) = 2\Pr[\mathrm{PPRF}^{\mathcal{G}}_{\mathsf{G}}(\lambda)] - 1$ and game $\mathrm{PPRF}^{\mathcal{G}}_{\mathsf{G}}(\lambda)$ is defined in Fig. 4. Here $\mathcal{G}$ must make exactly one oracle query consisting of a vector over $\{0,1\}^{\mathsf{G.il}(\lambda)}$. Proposition 1 extends, and we exploit this below.

Results. The following says that our construction for injective functions F extracts hardcore bits for any PT input sampler $\mathcal{I}$ with respect to which F is one way, meaning even for arbitrarily correlated inputs. The proof is in [12]:

Theorem 9. *Let F be an injective function family. Let G be a punctured PRF with $\mathsf{G.il} = \mathsf{F.il}$. Let d: $\mathbb{N} \to \mathbb{N}$ be a polynomial. Then there is a polynomial s such that the following is true. Let $\mathcal{I}$ be any PT input sampler with $\mathcal{I}.\mathsf{vl} = d$ and $\mathcal{I}.\mathsf{il} = \mathsf{F.il}$. Let Obf be any $\mathbf{S}_{\mathrm{diff}}(d)$-secure obfuscator. Assume F is one-way with respect to $\mathcal{I}$. Then the function family $\mathsf{H} = \mathbf{HC1}[\mathsf{G}, \mathsf{Obf}, s]$ defined in Section 4 is hardcore for F with respect to $\mathcal{I}$.*

A subtle point here is that s depends on $d = \mathcal{I}.\mathsf{vl}$ in addition to F and G, which means that the size of the key hk describing the hardcore function grows with the number of correlated inputs d on which we want the function to be hardcore. This is expected and due to [57] may not be avoidable under falsifiable assumptions.

Importantly for our applications, H does not depend on $\mathcal{I}$ beyond depending on $d = \mathcal{I}.\mathsf{vl}$ and $\mathsf{F}.\mathsf{il} = \mathcal{I}.\mathsf{il}$.

Since d in Theorem 9 is a polynomial, we may apply Proposition 3 to obtain the analog of Corollary 5 for correlated inputs, namely that, assuming only a $\boldsymbol{S}_{\mathrm{eq}}$-secure obfuscator (i.e. iO), there exists, for any injective F and any input sampler $\mathcal{I}$, a function family that returns polynomially-many bits and is hardcore for F with respect to $\mathcal{I}$. In [12] we discuss the application of Theorem 9 to the design of new D-PKE schemes that are PRIV-secure for arbitrarily correlated messages.

7 Application to D-PKE

A PKE scheme is deterministic if its encryption function is deterministic. Deterministic public-key encryption (D-PKE) is useful for many applications. However, it cannot provide IND-CPA security. BBO [6] define what it means for a D-PKE scheme to provide PRIV-security over an input sampler $\mathcal{I}$, the latter returning vectors of *arbitrarily correlated* messages to be encrypted. We restrict attention to distributions that are admissible, meaning that there exists a family of injective, trapdoor functions that is one-way relative to $\mathcal{I}$. The basic question that emerges is, for which admissible distributions $\mathcal{I}$ does there exist a D-PKE scheme that is PRIV-secure over $\mathcal{I}$? We show that this is true for all admissible distributions that are efficiently sampleable, assuming only the existence of iO. We obtain this result by combining Theorem 9 with techniques from [30]. Previously, this was known only in the ROM [6], under the assumption that UCE-secure functions exist [9], for distributions with limited correlation between messages [8, 18] or assuming lossy trapdoor functions [30]. See [12] for details.

Acknowledgments

We thank Daniel Wichs and Elette Boyle for comments and discussions. Bellare and Stepanovs were supported in part by grants NSF CNS-1116800 and NSF CNS-1228890. Tessaro was supported in part by grant NSF CNS-1423566.

References

1. A. Akavia, S. Goldwasser, and S. Safra. Proving hard-core predicates using list decoding. In *44th FOCS*, pages 146–159. IEEE Computer Society Press, Oct. 2003.
2. A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 474–495. Springer, Mar. 2009.
3. W. Alexi, B. Chor, O. Goldreich, and C. P. Schnorr. RSA and Rabin functions: Certain parts are as hard as the whole. *SIAM Journal on Computing*, 17(2):194–209, 1988.
4. P. Ananth, D. Boneh, S. Garg, A. Sahai, and M. Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689, Version 20131031:060024, October 31, 2013.

5. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Aug. 2001.
6. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Aug. 2007.
7. M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 232–249. Springer, Dec. 2009.
8. M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 360–378. Springer, Aug. 2008.
9. M. Bellare, V. T. Hoang, and S. Keelveedhi. Instantiating random oracles via UCEs. Cryptology ePrint Archive, Report 2013/424, 2013. Preliminary version in CRYPTO 2013.
10. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, Nov. 1993.
11. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006.
12. M. Bellare, I. Stepanovs, and S. Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. Cryptology ePrint Archive, Report 2013/873, 2013. `http://eprint.iacr.org/2013/873`.
13. N. Bitansky, R. Canetti, O. Paneth, and A. Rosen. On the existence of extractable one-way functions. In D. B. Shmoys, editor, *46th ACM STOC*, pages 505–514. ACM Press, May / June 2014.
14. N. Bitansky and O. Paneth. Point obfuscation and 3-round zero-knowledge. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 190–208. Springer, Mar. 2012.
15. L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on computing*, 15(2):364–383, 1986.
16. M. Blum and S. Goldwasser. An efficient probabilistic public-key encryption scheme which hides all partial information. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 289–302. Springer, Aug. 1984.
17. M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM journal on Computing*, 13(4):850–864, 1984.
18. A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 335–359. Springer, Aug. 2008.
19. D. Boneh and B. Waters. Constrained pseudorandom functions and their applications. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Dec. 2013.
20. E. Boyle, K.-M. Chung, and R. Pass. On extractability obfuscation. In Y. Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 52–73. Springer, Feb. 2014.
21. E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In H. Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Mar. 2014.
22. E. Boyle and R. Pass. Limits of extractability assumptions with distributional auxiliary input. Cryptology ePrint Archive, Report 2013/703, 2013.

23. C. Brzuska and A. Mittelbach. Using indistinguishability obfuscation via UCEs. Cryptology ePrint Archive, Report 2014/381, 2014. `http://eprint.iacr.org/2014/381`.
24. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 455–469. Springer, Aug. 1997.
25. D. Catalano, R. Gennaro, and N. Howgrave-Graham. The bit security of Paillier's encryption scheme and its applications. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 229–243. Springer, May 2001.
26. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer, Aug. 1998.
27. Y. Dodis, S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 361–381. Springer, Feb. 2010.
28. R. Fischlin and C.-P. Schnorr. Stronger security proofs for RSA and Rabin bits. *Journal of Cryptology*, 13(2):221–244, 2000.
29. D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. *Journal of Cryptology*, 26(1):39–74, Jan. 2013.
30. B. Fuller, A. O'Neill, and L. Reyzin. A unified approach to deterministic encryption: New constructions and a connection to computational entropy. In R. Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 582–599. Springer, Mar. 2012.
31. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, Oct. 2013.
32. S. Garg, C. Gentry, S. Halevi, and D. Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 518–535. Springer, Aug. 2014.
33. R. Gennaro. An improved pseudo-random generator based on the discrete logarithm problem. *Journal of Cryptology*, 18(2):91–110, Apr. 2005.
34. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33:792–807, 1986.
35. O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.
36. O. Goldreich and V. Rosen. On the security of modular exponentiation with application to the construction of pseudorandom generators. *Journal of Cryptology*, 16(2):71–93, Mar. 2003.
37. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
38. V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 182–200. Springer, Mar. 2011.
39. J. Håstad and M. Näslund. The security of individual RSA bits. In *39th FOCS*, pages 510–521. IEEE Computer Society Press, Nov. 1998.
40. J. Håstad, A. Schrift, and A. Shamir. The discrete logarithm modulo a composite hides $O(n)$ bits. *Journal of Computer and System Sciences*, 47(3):376–404, 1993.
41. S. Hohenberger, A. Sahai, and B. Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 201–220. Springer, May 2014.

42. B. S. Kaliski Jr. A pseudo-random bit generator based on elliptic logarithms. In A. M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 84–103. Springer, Aug. 1986.
43. A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 13*, pages 669–684. ACM Press, Nov. 2013.
44. E. Kiltz, A. O'Neill, and A. Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 295–313. Springer, Aug. 2010.
45. M. Lewko, A. O'Neill, and A. Smith. Regularity of lossy RSA on subdomains and its applications. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 55–75. Springer, May 2013.
46. D. L. Long and A. Wigderson. The discrete logarithm hides $O(\log n)$ bits. *SIAM journal on Computing*, 17(2):363–372, 1988.
47. P. D. MacKenzie and S. Patel. Hard bits of the discrete log with applications to password authentication. In A. Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 209–226. Springer, Feb. 2005.
48. M. Näslund. All bits of $ax+b \bmod p$ are hard. In N. Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 114–128. Springer, Aug. 1996.
49. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, May 1999.
50. S. Patel and G. S. Sundaram. An efficient discrete log pseudo random generator. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 304–317. Springer, Aug. 1998.
51. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *40th ACM STOC*, pages 187–196. ACM Press, May 2008.
52. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signature and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978.
53. A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In D. B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.
54. C.-P. Schnorr and W. Alexi. RSA-bits are 0.5 + epsilon secure. In T. Beth, N. Cot, and I. Ingemarsson, editors, *EUROCRYPT'84*, volume 209 of *LNCS*, pages 113–126. Springer, Apr. 1984.
55. R. Steinfeld, J. Pieprzyk, and H. Wang. On the provable security of an efficient RSA-based pseudorandom generator. In X. Lai and K. Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 194–209. Springer, Dec. 2006.
56. U. V. Vazirani and V. V. Vazirani. Efficient and secure pseudo-random number generation. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 193–202. Springer, Aug. 1984.
57. D. Wichs. Barriers in cryptography with weak, correlated and leaky sources. In R. D. Kleinberg, editor, *ITCS 2013*, pages 111–126. ACM, Jan. 2013.
58. A. C.-C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, Nov. 1982.
59. M. Zhandry. How to avoid obfuscation using witness PRFs. Cryptology ePrint Archive, Report 2014/301, 2014. `http://eprint.iacr.org/2014/301`.