

# Universal Computational Extractors and the Superfluous Padding Assumption for Indistinguishability Obfuscation

Christina Brzuska<sup>1</sup> Arno Mittelbach<sup>2</sup>

<sup>1</sup> Microsoft Research Cambridge, UK

<sup>2</sup> Cryptoplexity, Technische Universität Darmstadt, Germany  
christina.brzuska@gmail.com, mail@arno-mittelbach.de

**Abstract.** Universal Computational Extractors (UCEs), introduced by Bellare, Hoang and Keelveedhi (CRYPTO 2013), are a framework of assumptions on hash functions that allow to instantiate random oracles in a large variety of settings. Brzuska, Farshim and Mittelbach (CRYPTO 2014) showed that a large class of UCE assumptions with *computationally* unpredictable sources cannot be achieved, if indistinguishability obfuscation exists. In the process of circumventing obfuscation-based attacks, new UCE notions emerged, most notably UCEs with respect to *statistically* unpredictable sources that suffice for a large class of applications. However, the only standard model constructions of UCEs are for a small subclass considering only  $q$ -query sources which are *strongly statistically* unpredictable (Brzuska, Mittelbach; Asiacrypt 2014).

The contributions of this paper are threefold:

1. We show a surprising equivalence for the notions of strong unpredictability and (plain) unpredictability thereby lifting the construction from Brzuska and Mittelbach to achieve  $q$ -query UCEs for statistically unpredictable sources. This yields standard model instantiations for various ( $q$ -query) primitives including, deterministic public-key encryption, message-locked encryption, multi-bit point obfuscation, CCA-secure encryption, and more. For some of these, our construction yields the first standard model candidate.
2. We study the blow-up that occurs in indistinguishability obfuscation proof techniques due to puncturing and state the *Superfluous Padding Assumption* for indistinguishability obfuscation which allows us to lift the  $q$ -query restriction of our construction. We validate the assumption by showing that it holds for virtual black-box obfuscation.
3. Brzuska and Mittelbach require a strong form of point obfuscation secure in the presence of auxiliary input for their construction of UCEs. We show that this assumption is indeed necessary for the construction of injective UCEs.

**Keywords.** Universal computational extractors, standard model, superfluous padding assumption, indistinguishability obfuscation, point obfuscation

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Preliminaries</b>	<b>8</b>
2.1	Obfuscation . . . . .	8
2.1.1	General Purpose Obfuscation . . . . .	8
2.1.2	Point Obfuscation . . . . .	9
2.2	Puncturable PRFs . . . . .	11
<b>3</b>	<b>Universal Computational Extractors (UCE)</b>	<b>12</b>
3.1	The Status Quo . . . . .	12
3.2	UCE Constructions . . . . .	14
3.3	Bitwise UCEs . . . . .	14
<b>4</b>	<b>From Strong Unpredictability to (Plain) Unpredictability</b>	<b>15</b>
<b>5</b>	<b>Constructing <math>q</math>-Query <math>\text{UCE}[\mathcal{S}^{\text{sup}}]</math> Secure Functions</b>	<b>17</b>
5.1	The BM Construction . . . . .	17
5.2	Extending the BM Construction . . . . .	18
<b>6</b>	<b>Multi-key UCEs</b>	<b>19</b>
<b>7</b>	<b>The Superfluous Padding Assumption (SuPA)</b>	<b>20</b>
7.1	The Superfluous Padding Assumption . . . . .	21
7.2	Applying the SuP Assumption . . . . .	22
7.3	On the Validity of the Superfluous Padding Assumption . . . . .	23
<b>8</b>	<b>Point Obfuscation from UCEs</b>	<b>24</b>
<b>A</b>	<b>Gamehop Analysis for Theorem 5.1</b>	<b>31</b>

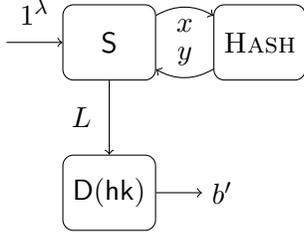


Figure 1: The schematic of the UCE game.

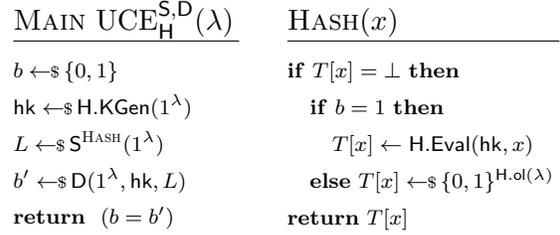


Figure 2: The pseudocode of the UCE game.

## 1 Introduction

Universal Computational Extractors (UCEs) are a definitional framework by Bellare, Hoang and Keelveedhi (BHK) [BHK13a] to model properties of random oracles. The reason why plain pseudo-random functions (PRFs) cannot be used to instantiate random oracles is that the key of a PRF is typically secret, while a hash-function needs to be publicly evaluatable and hence, the key needs to be public. A naïve model for a “standard model random oracle” is to extend the PRF definition and ask that an adversary can still not distinguish when it additionally gets the evaluation key. This definition is, of course, trivially not achievable since, using the key, an adversary can simply recompute the pseudorandom function locally for some value  $x$  and check whether the oracle’s answer is consistent with this value.

BHK give an elegant definition and split up the PRF adversary into two parts: a *source*  $S$  that gets oracle access (to either a random function or a keyed hash-function with a uniformly random key) and which provides some leakage  $L$  as well as a *distinguisher*  $D$  that gets leakage  $L$  and key  $hk$  of the hash-function (which is either the correct key in case the oracle implemented the hash function, or a random key). The task of the distinguisher is to distinguish between the case where the source has access to a random oracle and the case where the source has access to the (keyed) hash-function. We present the schematic and pseudocode of the UCE notion in Figures 1 and 2.

Splitting up the adversary alone does not prevent trivial attacks since a source can simply query its oracle on a random  $x$  to receive  $y$  and leak the pair  $(x, y)$  which can then be used by the distinguisher for a consistency check as it also has access to the hash key. To prevent trivial re-computation attacks, additional restrictions must be placed on the leakage. The main restriction that we consider in this paper and that was suggested by BHK is to require that the leakage should not “contain” any value  $x$  that the source queried to its oracle. In other words, the queries of the source need to be unpredictable from the leakage.

In their original work, BHK show that UCEs secure against *computationally* unpredictable sources suffice to instantiate random oracles in a large class of applications [BHK13a]. Brzuska, Farshim and Mittelbach [BFM14, BFM15] show that this UCE notion as well as some of the original applications cannot be instantiated, if indistinguishability obfuscation (iO) exists. A remedy against obfuscation-based attacks are UCEs secure against *statistically* unpredictable sources, as suggested in both [BFM14] and [BHK13b] independently and denoted  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  (where sup is short for Statistically UnPredictable). Here it is required that leakage  $L$  hides all of the sources queries *information-theoretically* instead of just computationally.  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  recovers many of the original applications and found further applications [BH15, DGG<sup>+</sup>15, MH14b] since. Not only does  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  have strong applications, but also it seems unlikely that iO-based attacks extend to  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  as statistically secure iO would imply that the polynomial hierarchy collapses [GR07].

In this work, we provide further evidence for the robustness of  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  against iO-based attacks. One can interpret our work as showing that if strong point function obfuscation exists then iO cannot

be used to break  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  (because we construct  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  functions from iO and point obfuscation).

**Standard-Model Constructions.** The current constructions for (general) UCE secure hash functions have only been validated in the random oracle model [BHK13a] and other idealized models [Mit14, BHK14]. Furthermore, the constructions in the ideal model also achieve UCE notions that are mutually exclusive with iO, which makes it hard to interpret the results, if we believe in indistinguishability obfuscation.

In turn, the only standard-model construction for UCEs—interestingly based on iO [BM14b]—only achieves security against  $q$ -query *strongly* statistically unpredictable sources.<sup>1</sup> That is, the notion puts two additional restrictions on the source on top of the standard  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  restrictions, namely, the source is only allowed to make an a priori bounded polynomial number of  $q$  queries and needs to be strongly statistically unpredictable. Strong (statistical) unpredictability was introduced by Brzuska and Mittelbach [BM14b] and means that not only should the leakage  $L$  hide the source’s queries  $x$  information-theoretically, but also,  $x$  should be information-theoretically hidden, even when given the oracle’s answer  $y$ . That means, in particular, that the source cannot leak  $x \oplus y$ , because, given  $y$ , one can recover  $x$ . However,  $\oplus$  is used in many constructions (for example, MLE, KDM, RKA, PFOB, DPKE and more, also see Table 1), and thus, replacing (plain) unpredictability with strong unpredictability substantially reduces the applicability of the notion.

As an example, consider message-locked encryption (MLE). MLE is a security notion for a symmetric encryption scheme where the encryption key is derived from the message to allow for deduplication [BKR13], for example, in cloud storage scenarios. A prominent scheme in the random oracle model derives the key for a message  $m$  as  $k \leftarrow \text{RO}(m)$  to then encrypt the message as  $c \leftarrow m \oplus \text{RO}(k)$  [DAB<sup>+</sup>02, BKR13]. When proving that a UCE secure function can replace the random oracle a source thus needs to leak  $m \oplus \text{HASH}(k)$ . A predictor in the strong unpredictability game which gets additionally  $k = \text{HASH}(m)$  and  $\text{HASH}(k)$  can easily recover both oracle queries and hence the source is not strongly unpredictable. As it turns out the source is, however, (plain) statistically unpredictable [BHK13b].

To summarize, the class of *strongly unpredictable* sources is smaller than the class of sources that satisfy (plain) unpredictability and hence the corresponding UCE-notion is weaker. The xor-based examples show that the UCE notion for sources that satisfy (plain) unpredictability have substantially more applications.

**Transformation from strong unpredictability to (plain) unpredictability.** Our first result is a surprising (and simple) transformation from strong to (plain) unpredictability. Namely, we start from a simple observation: when  $x$  is a long string (that is without loss of generality, because else, queries have low entropy due to their length and are easily predictable) and  $y$  is a single bit, then revealing  $y$  does not harm the predictability of  $x$ . Now, when we have several queries  $x_1, \dots, x_n$  and one is given the *set* of answers  $\{y_1, \dots, y_n\}$ , then this is still true, because the *set* of answers is either empty,  $\{0\}$ ,  $\{1\}$  or  $\{0, 1\}$ . Hence, for UCEs with a 1-bit output, strong unpredictability and (plain) unpredictability are actually equivalent. Using output length expansion for UCEs [BHK13b], one can then move to UCEs with multiple outputs. In short, we move from strongly unpredictable 1-output bit UCEs to unpredictable 1-output-bit UCEs and then to multi-output-bit UCEs. Together with the original construction by Brzuska and Mittelbach [BM14b], we yield  $q$ -query  $\text{UCE}[\mathcal{S}^{\text{sup}}]$ .<sup>2</sup> The construction by Brzuska and Mittelbach assumes

<sup>1</sup>The BM construction also achieves UCEs secure with respect to single-query strongly computationally unpredictable sources. For this work, however, we focus more on the statistical case.

<sup>2</sup>Note that, strictly speaking, the proof by BM of this construction implicitly assumes that the UCE function has long output and that output shortening in a black-box way is not easy for strong unpredictability. Therefore, we also provide a direct proof for the 1-output-bit version of the BM construction later in this paper.

indistinguishability obfuscation, puncturable PRFs and composable point obfuscation secure in the presence of statistically hard-to-invert auxiliary input. We refer to their papers [BM14b, BM14a] for an overview over the various notions of point obfuscations and here introduce only the relevant definitions for this paper (Section 2). Interestingly, although composable point obfuscation in the presence of statistically hard-to-invert auxiliary input may seem a strong assumption (alongside indistinguishability obfuscation) we observe in Section 8 that it is indeed a necessary assumption for constructing injective UCEs for (strongly) *statistically* unpredictable sources. The same holds for the computational case, that is, UCEs secure for single query (strongly) *computationally* unpredictable sources imply the existence of point obfuscation secure in the presence of *computationally* hard-to-invert auxiliary input.<sup>3</sup> This observation was independently made by Bellare, Stepanovs and Tessaro [BST15].

**UCEs with multiple keys.** We discuss the various applications of UCEs in Table 1, some of which require multiple keys. BHK conjecture that moving from single key to multi-key UCEs in a black-box way is a difficult endeavour. It turns out the construction by Brzuska and Mittelbach together with our transformation is already multi-key secure, i.e.,  $q$ -query  $\text{mUCE}[\mathcal{S}^{\text{sup}}]$ ; we provide a direct proof for this statement under the same assumptions as Brzuska and Mittelbach, that is, indistinguishability obfuscation, puncturable PRFs and point obfuscation secure in the presence of statistically hard-to-invert auxiliary input.

**Superfluous Padding Assumption and removing the  $q$ -query restriction.** Finally, we set out to remove the  $q$ -query restriction of our construction based on a novel, non-standard assumption for indistinguishability obfuscation that we propose in this paper. Firstly, let us discuss how the  $q$ -query restriction emerges within our proof technique for the construction. In the construction, the hash-key consists of an obfuscated circuit  $C$ . To prove the construction secure, we move via several game-hops from some leakage  $L$  which is generated from a source that has a hash-function as oracle to leakage  $L'$  that is generated by the same source when the oracle is a random oracle. For UCE-security, the two leakages need to be (computationally) indistinguishable in the presence of the hash-key, that is, the obfuscated circuit  $C$ . In a nutshell, we need to prove that

$$(L, \text{Obf}(C)) \approx_c (L', \text{Obf}(C)). \tag{1}$$

However, our proof only works, if we “pad” the circuit  $C$  before obfuscating it, that is, we add some redundant gates into the circuit to blow-up the size. The reason why blowing up the size of the circuit helps in the proof is that in the intermediate game hops we will encode  $q$  random strings into the circuit. Encoding  $q$  random strings, however, information-theoretically requires space that depends on the number of queries  $q$  which is why the construction at the time of key generation needs to get  $q$  as input.

Intuitively a good obfuscation should have the property that if  $\text{Obf}(\text{PAD}(C))$  hides something then so should  $\text{Obf}(C)$ . That is, an obfuscation of a padded circuit should not be better than the obfuscation of the unpadded circuit. One reason is that  $\text{Obf}$  itself could always first perform a padding operation and it seems counterintuitive that this results in a better obfuscator. The *Superfluous Padding Assumption* (SuPA) captures the above intuition and states that padding a circuit does not contribute to security. Under the SuP assumption we can, thus, move from a  $q$ -query  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  (or  $\text{mUCE}[\mathcal{S}^{\text{sup}}]$ ) secure

---

<sup>3</sup>We note that UCEs with multiple keys (and long output) suffice for implying composable point obfuscation with (computationally/statistically) hard-to-invert auxiliary information. Composable point obfuscation with computationally hard-to-invert auxiliary information is mutually exclusive with indistinguishability obfuscation [BM14a]. However, we only construct UCEs with multiple keys and long output against statistically unpredictable sources and thus do not contradict any known results.

<b>UCEs for strongly computationally unpredictable sources</b> ( $\text{UCE}[\mathcal{S}^{\text{scup}}]$ )			Standard model constructions for constant-query $\text{UCE}[\mathcal{S}^{\text{scup}}]$ and $q$ -query $\text{UCE}[\mathcal{S}^{\text{sup}}]$ in [BM14b].
<b>HC</b>	UCEs are hardcore for any one-way function.	[BHK13b, BM14b]	
<b>BR93</b>	UCEs are sufficient to instantiate the BR93 PKE scheme [BR93].	[BHK13b, BM14b]	
<b>UCEs for strongly statistically unpredictable sources</b> ( $\text{UCE}[\mathcal{S}^{\text{sup}}]$ )			
<b>CIH</b>	UCEs are correlated input secure hash functions.	[BHK13b, BM14b]	
<b>IMMU</b>	UCEs can be used to construct secure immunizers as countermeasure for backdoored PRGs.	[DGG <sup>+</sup> 15]	
<b>UCEs for statistically unpredictable sources</b> ( $\text{UCE}[\mathcal{S}^{\text{sup}}]$ )			
<b>MLE</b>	UCEs can instantiate message-locked encryption schemes.	[BHK13b]	
<b>DPKE</b>	UCEs can instantiate deterministic PKE schemes.	[BH15]	
<b>CCA</b>	UCEs can be used to build CCA secure PKE schemes.	[MH14b]	
<b>Multi-key UCEs for statistically unpredictable sources</b> ( $\text{mUCE}[\mathcal{S}^{\text{sup}}]$ )			
<b>KDM</b>	UCEs can instantiate the BRS scheme [BRS03] to obtain a standard-model symmetric encryption scheme which offers security against key-dependent messages.	[BHK13b]	
<b>RKA</b>	The same instantiation as for KDM can be shown to be also secure against related-key attacks.	[BHK13b]	
<b>PFOB</b>	UCEs can be used to instantiate the RO point function obfuscation scheme due to Lynn et al. [LPS04].	[BHK13b]	
<b>UCEs for statistically reset-secure sources</b> ( $\text{UCE}[\mathcal{S}^{\text{srts}}]$ )			Standard model construction in this paper.
<b>GB</b>	UCEs can be used to obtain an adaptive garbling scheme.	[BHK13b]	
<b>HPKE</b>	UCEs can be used to instantiate a PKE scheme which hedges against bad randomness.	[BH15]	

Table 1: A list of UCE notions and corresponding applications. The first block covers UCE functions that were constructed by Brzuska and Mittelbach in [BM14b]. Note that BM only constructed  $q$ -query variants for strong statistical and constant query variants for strong computational sources. In this paper we lift their construction to also cover the UCE notions of the second block, i.e., (plain) statistically unpredictable sources. Assuming the Superfluous Padding Assumption we further lift the bound on  $q$ . Finally, for the last block no standard model constructions are known. The applications HC, KDM, RKA and PFOB only needs UCEs which are secure with respect to sources making a single query.

function to a fully  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  (or  $\text{mUCE}[\mathcal{S}^{\text{sup}}]$ ) secure function. In Section 7, we validate the SuP assumption by showing that it holds for virtual-black-box obfuscation. The intuition is that virtual black-box obfuscation assures that all attacks that can be run on the obfuscated circuit can also be run by the simulator. However, the simulator only needs black-box access to the circuit and hence, whether we consider a padded circuit or a non-padded circuit does not matter. We will then discuss why it might be reasonable to also assume SuPA for indistinguishability obfuscation.

**On the Plausibility of various UCE notions in the presence of iO.** If indistinguishability obfuscation exists, then a subclass of UCE notions cannot be achieved. The focus of this paragraph is on the technical aspects of various UCE notions and how these technical aspects affect whether a notion is susceptible to iO-based attacks or not.

BFM [BFM14] show that, if iO exists, then UCEs secure against sources that satisfy (plain) computational unpredictability do not exist. Towards this goal, they construct a source that produces some leakage  $L$  which consists of an obfuscated circuit that has a query-answer pair of the source hardcoded into it. Because of this query-answer pair, the leakage only hides the query computationally,

but not statistically—unless statistically secure iO exist, which seems unlikely, because it would cause the polynomial hierarchy to collapse [GR07, GR14]. Therefore, BFM and BHK both independently suggest to consider UCEs secure against *statistically* unpredictable sources. This intuition has not been countered so far, and indeed, if iO were able to break UCEs secure against statistically unpredictable sources, then this would imply that iO is mutually exclusive with point obfuscation with statistically hard-to-invert leakage, and we currently do not believe that this statement is true. Actually, just as Bellare et al. [BST15], we do not even believe that iO is mutually exclusive with point obfuscation with *computationally* hard-to-invert leakage.

UCEs secure against statistical unpredictability recover all of the original UCE applications, except for a universal instantiation for the Encrypt-with-Hash transform and a universal hardcore function. In order to counter iO-based attacks and to recover all original applications, BHK thus also proposed several further notions with respect to *computational* versions of unpredictability, e.g., bounded parallel sources as a universal instantiation for the Encrypt-with-Hash transform. It was shown [BFM14] that bounded parallel sources are mutually exclusive with iO, but as it turns out, actually, finding a universal instantiation for the Encrypt-with-Hash transform is mutually exclusive with iO [BFM15, BH15]. The second computational unpredictability notion that BHK proposed were so-called split sources where the sources have a special form. This notion suffices to construct a universal hardcore function. In their work, Brzuska and Mittelbach [BM14b] suggest and construct UCEs secure against computationally strongly unpredictable sources, assuming iO and point obfuscation with computationally hard-to-invert leakage. They show, that this notion implies UCEs secure against split sources and thus they obtain a universal hardcore function. We note that the proof by BM only achieves constant-query security and that only single-query security is needed for the universal hardcore function application. What looks like a limitation of their proof is, indeed, an inherent obstacle. As implied by our result, UCEs secure against computationally strongly unpredictable sources with a polynomial number of queries is mutually exclusive with iO. When considering only constantly-many queries the impossibility is not known to hold. Although split sources are not known to have any application beyond universal hardcore functions, one might ask whether split sources with polynomial many queries can exist, if iO exists. Interestingly, Bellare et al. [BST15] concurrently and independently show an iO-based attack on split sources with polynomial-many queries. Note that their result is a stronger negative result and subsumes ours, because split sources are a smaller source class than computationally strongly unpredictable sources. Our work and [BST15] show that all current definitional propositions for computational variants of unpredictability can have at most constantly (or logarithmically) many queries, if iO exist.

As far as we know this query-limitation does not apply to *statistical* unpredictability, and for UCEs secure against statistically unpredictable sources, regardless of the number of queries, iO-based attacks seem unlikely, as discussed above. However, we still need an additional assumption (the superfluous padding assumption) to move from a positive result for a bounded polynomial to a positive result for an unbounded polynomial number of queries. In conclusion, bounds on the queries affect both, feasibility and infeasibility results. We thus suggest that, whenever one proposes an application for UCEs, to carefully investigate the dependency of the application w.r.t. the number of queries, most importantly, when one is concerned with variants of computational unpredictability which we feel we understand less well.

**Outline.** We present preliminaries on different notions of obfuscation as well as puncturable PRFs in Section 2. In Section 3, we present the current state-of-the-art in UCEs. In Section 4, we present our transformation from strong unpredictability to (plain) unpredictability and give a proof of a  $q$ -query UCE[ $S^{\text{sup}}$ ]-secure function—part of the proof is delegated to Appendix A. In Section 6, we discuss and construct multi-key UCEs. In Section 7 we introduce and discuss the SuP assumption and finally we

discuss the relationship between strong point obfuscators secure in the presence of auxiliary information and UCEs in Section 8.

## 2 Preliminaries

NOTATION. By  $\lambda \in \mathbb{N}$ , we denote the security parameter that we give to all algorithms implicitly in unary representation  $1^\lambda$ . By  $\{0, 1\}^\ell$  we denote the set of all bit-strings of length  $\ell$ , and by  $\{0, 1\}^*$  the set of all bit-strings of finite length. If  $x, y \in \{0, 1\}^*$  are two bit strings of the same length, then we denote their inner product over  $\mathbb{GF}(2)$  by  $\langle x, y \rangle$ . The length of  $x$  is denoted by  $|x|$ . For a finite set  $X$ , we denote the action of sampling  $x$  uniformly at random from  $X$  by  $x \leftarrow_{\$} X$ , and denote the cardinality of  $X$  by  $|X|$ . We denote by  $[i]$  the set  $\{1, \dots, i\}$ . Algorithms are assumed to be randomized, unless otherwise stated. We call an algorithm efficient or PPT if it runs in time polynomial in the security parameter. We say a function  $\text{negl}(\lambda)$  is negligible if  $\text{negl}(\lambda) \in \lambda^{-\omega(1)}$ . We say a function  $\text{poly}(\lambda)$  is polynomial if  $\text{poly}(\lambda) \in \lambda^{\mathcal{O}(1)}$ .

### 2.1 Obfuscation

#### 2.1.1 General Purpose Obfuscation

We begin with recalling the notions of indistinguishability obfuscation and differing-inputs obfuscation originally proposed by Barak et al. [BGI<sup>+</sup>01, BGI<sup>+</sup>12]. While indistinguishability obfuscation intuitively captures that the obfuscation of two functionally equivalent circuits cannot be distinguished, differing-inputs obfuscation says that if an efficient distinguisher for two obfuscated circuits exists then one can efficiently find an input on which the two circuits differ. We here give a game-based definition, following the definitional framework of [BST14] which captures indistinguishability obfuscation based notions via the IO security game and a class of samplers  $\text{Sam}$ .

**Definition 2.1** A PPT algorithm  $\text{Obf}$  is called an obfuscation scheme if, on input the security parameter  $1^\lambda$  and a description of a circuit  $C$ , it returns (a description of) a circuit  $C'$  such that  $\forall x : C(x) = C'(x)$ . We call a PPT algorithm  $\text{Sam}$  a circuit sampler if on input the security parameter  $1^\lambda$  sampler  $\text{Sam}$  outputs  $(C_0, C_1, \text{aux})$  where  $C_0$  and  $C_1$  are descriptions of circuits and  $\text{aux}$  is a string. If  $\text{Sam}$  is a circuit sampler and  $\text{Obf}$  is an obfuscation scheme we define the advantage  $\text{Adv}_{\text{Obf}, \text{Sam}, \text{D}}^{\text{io}}(\cdot)$  for a distinguisher  $\text{D}$  relative to game IO:

$$\text{Adv}_{\text{Obf}, \text{Sam}, \text{D}}^{\text{io}}(\lambda) := 2 \cdot \Pr \left[ \text{IO}_{\text{Obf}, \text{Sam}}^{\text{D}}(\lambda) \right] - 1$$

$$\begin{array}{l} \text{IO}_{\text{D}, \text{Sam}}^{\text{Obf}}(\lambda) \\ \hline b \leftarrow_{\$} \{0, 1\} \\ (C_0, C_1, \text{aux}) \leftarrow_{\$} \text{Sam}(1^\lambda) \\ \bar{C} \leftarrow_{\$} \text{Obf}(1^\lambda, C_b) \\ b' \leftarrow_{\$} \text{D}(1^\lambda, \bar{C}, \text{aux}) \\ \mathbf{return} \ (b = b') \end{array}$$

If  $\mathcal{S}$  is a class of circuit samplers, then we call  $\text{Obf}$   $\mathcal{S}$ -secure if for all  $\text{Sam} \in \mathcal{S}$  and all efficient distinguishers  $\text{D}$  advantage  $\text{Adv}_{\text{Obf}, \text{Sam}, \text{D}}^{\text{io}}(\lambda)$  is negligible in  $\lambda$ .

INDISTINGUISHABILITY OBFUSCATION. We can now capture the notions of indistinguishability obfuscation as well as differing-inputs obfuscation via restricting the class of samplers to *equality samplers* and *differing-inputs samplers*, respectively.

**Definition 2.2 (Equality circuit sampler)** We call a non-uniform algorithm  $\text{Sam}$  an equality circuit sampler if for all security parameters  $\lambda \in \mathbb{N}$  it outputs a triple  $(C_0, C_1, \text{aux})$  consisting of two circuit descriptions and a string such that with overwhelming probability over the coins of  $\text{Sam}$  we have that the circuits  $C_0$  and  $C_1$  have the same size, number of inputs and number of outputs and are functionally equivalent, that is

$$\Pr_{(C_0, C_1, \text{aux}) \leftarrow \text{Sam}(1^\lambda)} [ |C_0| = |C_1| \wedge \forall x : C_0(x) = C_1(x) ] \geq 1 - \text{negl}(\lambda).$$

Let  $\mathcal{S}_{\text{eq}}$  be the class of all equality circuit samplers (including non-efficient samplers). Then we capture the notion of indistinguishability obfuscation by Barak et al. [BGI<sup>+</sup>01] (which is used in the candidate construction by Garg et al. [GGH<sup>+</sup>13]) by an obfuscation scheme  $\text{Obf}$  secure for  $\mathcal{S}_{\text{eq}}$ . For this note, that a sequence of pairs of functionally equivalent circuits can be described by a deterministic non-uniform sampler. The converse direction follows via an averaging argument.

**DIFFERING-INPUTS OBFUSCATION.** Next we consider the class of differing-inputs circuit samplers that allows us to capture the notion of differing-inputs obfuscation.

**Definition 2.3 (Differing-inputs circuit sampler)** We call a non-uniform algorithm  $\text{Sam}$  a differing-inputs circuit sampler if for all security parameters  $\lambda \in \mathbb{N}$  it outputs a triple  $(C_0, C_1, \text{aux})$  consisting of two circuit descriptions and a string such that advantage  $\text{Adv}_{\text{Sam}, \text{Ext}}^{\text{diff}}(\cdot)$  is negligible for all PPT algorithms  $\text{Ext}$  and where the advantage is defined as (relative to game  $\text{Diff}$  on the right):

$$\text{Adv}_{\text{Sam}, \text{Ext}}^{\text{diff}}(\lambda) := \Pr \left[ \text{Diff}_{\text{Sam}}^{\text{Ext}}(\lambda) \right]$$

$$\frac{\text{Diff}_{\text{Sam}}^{\text{Ext}}(\lambda)}{(C_0, C_1, \text{aux}) \leftarrow \text{Sam}(1^\lambda)$$

$$x \leftarrow \text{Ext}(1^\lambda, C_0, C_1, \text{aux})$$

$$\text{return } (C_0(x) \neq C_1(x))$$

We capture differing-inputs obfuscation by considering obfuscators  $\text{Obf}$  secure for the class of all (not necessarily efficient) differing-inputs samplers  $\mathcal{S}_{\text{diff}}$ .

The notion of differing-inputs obfuscation recently also gained much attention [ABG<sup>+</sup>13, BCP14, BP13]. In particular, we will build on the work by Boyle, Chung and Pass [BCP14] who show that any general indistinguishability obfuscator also yields a mild version of a differing-inputs obfuscator. That is, any indistinguishability obfuscator for all circuits in  $\text{P/poly}$  is also a differing-inputs obfuscator for circuits that differ on at most polynomially many inputs.

**Theorem 2.4 ([BCP14])** Let  $\text{iO}$  be an indistinguishability obfuscator for  $\text{P/poly}$ . Let  $\text{Sam}$  be a differing-inputs circuit sampler for which there exists a polynomial  $d : \mathbb{N} \rightarrow \mathbb{N}$ , such that

$$\Pr \left[ |\{x : C_0(x) \neq C_1(x)\}| \leq d(\lambda) \mid (C_0, C_1, \text{aux}) \leftarrow \text{Sam}(1^\lambda) \right] \geq 1 - \text{negl}(\lambda).$$

Then  $\text{iO}$  is a differing-inputs obfuscator for  $\text{Sam}$ , i.e., obfuscator  $\text{iO}$  is  $\{\text{Sam}\}$ -secure.

### 2.1.2 Point Obfuscation

Besides the general purpose obfuscation notions of indistinguishability obfuscation and differing-inputs obfuscation we employ special purpose obfuscators for point functions  $p_x$  which for a value  $x \in \{0, 1\}^*$  are defined as

$$p_x(s) := \begin{cases} 1 & \text{if } s = x \\ \perp & \text{o/w} \end{cases}$$

We consider a variant of point function obfuscators under auxiliary input which were first formalized by Canetti [Can97]. We here give the definition from [BP12] presented in a game based formulation. The first definition formalizes unpredictable distributions which are in turn used to define obfuscators for point functions.

**Definition 2.5 (Unpredictable distribution)** *An algorithm  $\mathcal{B}$  that on input the security parameter  $1^\lambda$  outputs two strings  $(x, \mathbf{aux})$  is called computationally unpredictable (resp. statistically unpredictable) if no PPT algorithm (resp. unbounded algorithm) can predict  $x$  from  $\mathbf{aux}$ . That is, for every PPT (resp. unbounded) algorithm  $\mathcal{P}$  and for all large enough  $\lambda$ :*

$$\Pr_{(\mathbf{aux}, x) \leftarrow \mathcal{D}_\lambda} \left[ \mathcal{P}(1^\lambda, \mathbf{aux}) = x \right] \leq \text{negl}(\lambda)$$

We let  $\mathcal{D}^{\text{cup}}$  denote all efficient computationally distributions and denote by  $\mathcal{D}^{\text{sup}}$  all efficient statistically unpredictable distributions.

In the following we define point obfuscators secure in the presence of auxiliary inputs relative to a class of distributions  $\mathcal{D}$ . We write  $\text{AIPO}[\mathcal{D}]$  to capture the class of obfuscators that are secure against all distributions in  $\mathcal{D}$ .

**Definition 2.6 (Auxiliary input point obfuscation (AIPO))** *A PPT algorithm AIPO is a point obfuscator if on input a string  $x$  it outputs a polynomial-size circuit that returns 1 on  $x$  and 0 everywhere else. It is called secure under auxiliary input for distribution class  $\mathcal{D}$  if it satisfies the following secrecy property: for any distribution  $\mathcal{B}_1 \in \mathcal{D}$  it holds for any PPT algorithm  $\mathcal{B}_2$  that the probability that the following experiment outputs true for  $(\mathcal{B}_1, \mathcal{B}_2)$  is negligibly close to  $\frac{1}{2}$ :*

$$\text{Adv}_{\text{AIPO}, \mathcal{B}_1, \mathcal{B}_2}^{\text{po}}(\lambda) = 2 \cdot \Pr \left[ \text{PO}_{\text{AIPO}}^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \right] - 1 \leq \text{negl}(\lambda)$$

---

```

POAIPO $\mathcal{B}_1, \mathcal{B}_2$ ( $\lambda$ )
 $b \leftarrow \{0, 1\}$ 
 $(x_0, \mathbf{aux}) \leftarrow \mathcal{B}_1(1^\lambda)$ 
 $x_1 \leftarrow \{0, 1\}^{|x_0|}$ 
 $p \leftarrow \text{AIPO}(x_b)$ 
 $b' \leftarrow \mathcal{B}_2(1^\lambda, p, \mathbf{aux})$ 
return  $b = b'$ 

```

The probability is over the coins of adversary  $(\mathcal{B}_1, \mathcal{B}_2)$ , the coins of AIPO and the choices of  $x_1$  and  $b$ . We'll denote by  $\text{AIPO}[\mathcal{D}]$  the class of all PPT point obfuscators secure against distributions in  $\mathcal{D}$ .

**COMPOSABLE AIPOS.** We can define a composable variant of the above definition by considering distributions that output a vector  $\mathbf{x}$  rather than a point  $x$ . Computational (resp. statistical) unpredictability then requires that for all PPT (resp. unbounded) algorithms  $\mathcal{P}$  and for all large enough  $\lambda$ :

$$\Pr_{(\mathbf{aux}, \mathbf{x}) \leftarrow \mathcal{D}_\lambda} \left[ \mathbf{x}[i] = x : x, i \leftarrow \mathcal{P}(1^\lambda, \mathbf{aux}) \right] \leq \text{negl}(\lambda)$$

Hence the AIPO security game would change accordingly to

```

 $b \leftarrow \{0, 1\}$ 
 $(\mathbf{x}, \mathbf{aux}) \leftarrow \mathcal{B}_1(1^\lambda)$ 
for  $i = 1, \dots, |\mathbf{x}|$  do
  if  $b = 1$  then
     $\mathbf{x}[i] \leftarrow \{0, 1\}^{|\mathbf{x}[i]|}$ 
     $\mathbf{p}[i] \leftarrow \text{AIPO}(\mathbf{x}[i])$ 
 $b' \leftarrow \mathcal{B}_2(1^\lambda, \mathbf{p}, \mathbf{aux})$ 
return  $b = b'$ 

```

Brzuska and Mittelbach have recently shown that in the computational setting (where predictor  $P$  is restricted to be PPT) such composable point obfuscators do not exist if indistinguishability obfuscation exists [BM14a]. In the statistical setting where auxiliary input  $\text{aux}$  must hide the points in  $X$  statistically we have a candidate construction of such a composable point obfuscator by Canetti [Can97]. Bitansky and Canetti [BC10] show that the point obfuscation scheme of Canetti [Can97] is a so-called  $t$ -composable VGB point obfuscator under the (non standard)  $t$ -Strong Vector Decision Diffie–Hellman assumption. Matsuda and Hanaoka [MH14a] show that such composable VGB point obfuscators imply the existence of composable AIPO with respect to statistically unpredictable distributions. Jumping ahead, in this paper we show that (injective) UCEs with respect to statistically unpredictable sources imply composable AIPO with respect to statistically unpredictable distributions.

## 2.2 Puncturable PRFs

Besides point function obfuscation schemes, our main ingredient in the upcoming proofs are so-called puncturable pseudorandom functions (PRF) [SW14]. We consider families of functions  $F$  consisting of algorithms  $F.\text{KGen}$ ,  $F.\text{kl}$ ,  $F.\text{Eval}$ ,  $F.\text{il}$  and  $F.\text{ol}$ . Algorithm  $F.\text{KGen}$  is a PPT algorithm taking the security parameter  $1^\lambda$  and outputting a key  $k \in \{0, 1\}^{F.\text{kl}(\lambda)}$  where  $F.\text{kl} : \mathbb{N} \rightarrow \mathbb{N}$  denotes the key length. Functions  $F.\text{il} : \mathbb{N} \rightarrow \mathbb{N}$  and  $F.\text{ol} : \mathbb{N} \rightarrow \mathbb{N}$  denote the input and output length functions associated to  $F$  and for any  $x \in \{0, 1\}^{F.\text{il}(\lambda)}$  and  $k \leftarrow_{\$} F.\text{KGen}(1^\lambda)$  we have that  $F.\text{Eval}(k, x) \in \{0, 1\}^{F.\text{ol}(\lambda)}$ , where the PPT algorithm  $F.\text{Eval}$  denotes the “evaluation” function associated to  $F$ .

A family of puncturable PRFs  $G := (G.\text{KGen}, G.\text{Puncture}, G.\text{kl}, G.\text{Eval}, G.\text{il}, G.\text{ol})$  consists of functions that specify input length, output length and key length as well as a key generation algorithm and a deterministic evaluation algorithm. Additionally, there is a PPT puncturing algorithm  $G.\text{Puncture}$  which on input a polynomial-size set  $S \subseteq \{0, 1\}^{G.\text{il}(\lambda)}$ , outputs a special key  $k_S$ . We here give a game based variant of selectively secure puncturable PRFs where the adversary may adaptively request challenge points (via oracle **challenge**) while it is only allowed to see a single punctured key [BW13]. Via a standard hybrid argument it can be shown that this formulation is equivalent to allowing only a single challenge query.

**Definition 2.7** *A family of functions  $G := (G.\text{KGen}, G.\text{Puncture}, G.\text{kl}, G.\text{Eval}, G.\text{il}, G.\text{ol})$  is called puncturable PRF if the following properties are observed*

- **Functionality preserved under puncturing.** *For every PPT adversary  $\mathcal{A}$  such that  $\mathcal{A}(1^\lambda)$  outputs a polynomial-size set  $S \subseteq \{0, 1\}^{G.\text{il}(\lambda)}$ , it holds for all  $x \in \{0, 1\}^{G.\text{il}(\lambda)}$  where  $x \notin S$  that:*

$$\Pr \left[ G.\text{Eval}(k, x) = G.\text{Eval}(k_S, x) : k \leftarrow_{\$} G.\text{KGen}(1^\lambda), k_S \leftarrow_{\$} G.\text{Puncture}(k, S) \right] = 1$$

- **Pseudorandom at punctured points.** *For every PPT adversary  $(\mathcal{A}_1, \mathcal{A}_2)$  the advantage  $\text{Adv}_{\mathcal{A}_1, \mathcal{A}_2}^{\text{pprf}}(\cdot)$  is negligible:*

$$\text{Adv}_{\mathcal{A}}^{\text{pprf}}(\lambda) = 2 \cdot \Pr \left[ \text{PPRF}_{\mathcal{A}_1, \mathcal{A}_2}^G(\lambda) \right] - 1$$

where game PPRF is defined as

$\text{PPRF}_{\mathcal{A}_1, \mathcal{A}_2}^G(\lambda)$	<b>challenge</b> ( $x$ )
$S \leftarrow \{\}; b \leftarrow_{\$} \{0, 1\}$	<b>if</b> $x \in S$ <b>then return</b> $\perp$
$k \leftarrow_{\$} G.\text{KGen}(1^\lambda)$	$S \leftarrow S \cup \{x\}$
$\text{state} \leftarrow_{\$} \mathcal{A}_1^{\text{challenge}}(1^\lambda)$	<b>if</b> $b = 0$ <b>then</b>
$k^* \leftarrow_{\$} G.\text{Puncture}(k, S)$	$y \leftarrow G.\text{Eval}(k, x)$
$b' \leftarrow_{\$} \mathcal{A}_2(1^\lambda, \text{state}, k^*)$	<b>else</b> $y \leftarrow_{\$} \{0, 1\}^{G.\text{ol}(\lambda)}$
<b>return</b> $(b = b')$	<b>return</b> $y$

$\text{UCE}_{\mathbf{H}}^{\mathbf{S},\mathbf{D}}(\lambda)$	$\text{HASH}(x)$	$\text{Pred}_{\mathbf{S}}^{\mathbf{P}}(\lambda)$	$\text{HASH}(x)$
$b \leftarrow_{\mathbf{S}} \{0, 1\}$	<b>if</b> $T[x] = \perp$ <b>then</b>	<b>done</b> $\leftarrow$ <b>false</b> ; $Q \leftarrow \{\}$	<b>if</b> <b>done</b> = <b>false</b> <b>then</b>
$\text{hk} \leftarrow_{\mathbf{S}} \text{H.KGen}(1^\lambda)$	<b>if</b> $b = 1$ <b>then</b>	$L \leftarrow_{\mathbf{S}} \mathbf{S}^{\text{HASH}}(1^\lambda)$	$Q \leftarrow Q \cup \{x\}$
$L \leftarrow_{\mathbf{S}} \mathbf{S}^{\text{HASH}}(1^\lambda)$	$T[x] \leftarrow \text{H.Eval}(\text{hk}, x)$	<b>done</b> $\leftarrow$ <b>true</b>	<b>if</b> $T[x] = \perp$ <b>then</b>
$b' \leftarrow_{\mathbf{S}} \mathbf{D}(1^\lambda, \text{hk}, L)$	<b>else</b> $T[x] \leftarrow_{\mathbf{S}} \{0, 1\}^{\text{H.ol}(\lambda)}$	$Q' \leftarrow_{\mathbf{S}} \mathbf{P}^{\text{HASH}}(1^\lambda, L)$	$T[x] \leftarrow_{\mathbf{S}} \{0, 1\}^{\text{H.ol}(\lambda)}$
<b>return</b> $(b = b')$	<b>return</b> $T[x]$	<b>return</b> $(Q \cap Q' \neq \{\})$	<b>return</b> $T[x]$

Figure 3: The UCE security game together with the unpredictability game (on the right). In the UCE game source  $\mathbf{S}$  has access to  $\text{HASH}$ , which returns real or ideal hash values, and leaks  $L$  to a distinguisher  $\mathbf{D}$ . The latter additionally gets the hash key and outputs a bit  $b'$ . On the right we give the unpredictability game.

As observed by [BW13, BGI14, KPTZ13] puncturable PRFs can, for example, be constructed from pseudorandom generators via the GGM tree-based construction [GGM84]. We note that, as AIPO implies one-way functions AIPO, thus, also implies the existence of puncturable PRFs [BM14b].

### 3 Universal Computational Extractors (UCE)

The UCE Framework by Bellare, Hoang, and Keelveedhi (BHK; [BHK13a]) provides a standardized method to define security properties for keyed hash functions that model extractor capabilities of random oracles. Loosely speaking, UCEs are PRF-like assumptions that split a distinguisher (which should differentiate whether it operates relative to a UCE function or relative to a random oracle) into two parts: a first adversary  $\mathbf{S}$  that gets access to a keyed hash function or a random oracle (and which is called the *source*), and a second adversary  $\mathbf{D}$  that gets the hash key  $\text{hk}$  (and which is called the *distinguisher*). The two algorithms together try to guess whether the source was given access to a keyed hash function or to a random oracle.

Concretely, the UCE notions are defined via a two-stage UCE game (we present the pseudocode in Figure 3 on the left). First, the source  $\mathbf{S}$  is run with oracle access to  $\text{HASH}$  to output some leakage  $L$ . Subsequently, distinguisher  $\mathbf{D}$  is run on the leakage  $L$  and hash key  $\text{hk}$  but without access to oracle  $\text{HASH}$ . Distinguisher  $\mathbf{D}$  outputs a single bit  $b$  indicating whether oracle  $\text{HASH}$  implements a random oracle or hash function  $\mathbf{H}$  with key  $\text{hk}$ .

**FORMAL UCE DEFINITION.** We denote hash function (families) by  $\mathbf{H}$ . Let  $\mathbf{H} = (\text{H.KGen}, \text{H.Eval}, \text{H.kl}, \text{H.il}, \text{H.ol})$  be a hash-function family and let  $(\mathbf{S}, \mathbf{D})$  be a pair of PPT algorithms. We define the UCE advantage of a pair  $(\mathbf{S}, \mathbf{D})$  against  $\mathbf{H}$  through

$$\text{Adv}_{\mathbf{H}, \mathbf{S}, \mathbf{D}}^{\text{uce}}(\lambda) := 2 \cdot \Pr \left[ \text{UCE}_{\mathbf{H}}^{\mathbf{S}, \mathbf{D}}(\lambda) \right] - 1,$$

where game  $\text{UCE}_{\mathbf{H}}^{\mathbf{S}, \mathbf{D}}(\lambda)$  is shown in Figure 3 on the left.

#### 3.1 The Status Quo

Without any restrictions the UCE game models that a concrete function is indistinguishable from a random oracle and hence it is not surprising that the pair  $(\mathbf{S}, \mathbf{D})$  can easily win the UCE game. For example, say, source  $\mathbf{S}$  makes a random query  $x$  to receive  $y \leftarrow \text{HASH}(x)$  and outputs  $(x, y)$  as leakage. As distinguisher  $\mathbf{D}$  knows the hash key  $\text{hk}$  as well as the leakage  $(x, y)$ , it can recompute the hash value and check whether  $y = \text{H}(\text{hk}, x)$ . In order to get a concrete UCE notion it is thus necessary to define a restriction on sources and distinguishers. Given this flexibility a key issue is to come up with *good UCE notions* that are not too strong (that is, in particular, they should be instantiable in the standard

model) and, on the other hand, are not too weak in order to be useful in applications. We say a hash function  $H$  is UCE secure for sources  $S \in \mathcal{S}$  denoted by  $\text{UCE}[\mathcal{S}]$ , if for all PPT sources  $S \in \mathcal{S}$  and all PPT distinguishers  $D$  the advantage  $\text{Adv}_{H,S,D}^{\text{uce}}(\lambda)$  is negligible.

BHK present several possible restrictions giving rise to various UCE notions and show that these allow to instantiate random oracles in a wide range of interesting applications. The first two UCE notions to be defined were called UCEs for (computationally) unpredictable sources (denoted by  $\text{UCE}[\mathcal{S}^{\text{cup}}]$ ) and UCEs for reset-secure sources (denoted by  $\text{UCE}[\mathcal{S}^{\text{crs}}]$ ) [BHK13a].<sup>4</sup> The two notions allowed to instantiate random oracles in a wide range of interesting applications: deterministic public-key encryption (DPKE), message-locked encryption (MLE), universal hardcore functions (HC), point function obfuscation (MBPO), key dependent message security (KDM), related key security (RKA), correlation input secure hash functions (CIH) and more. In particular the (weaker) notion of  $\text{UCE}[\mathcal{S}^{\text{cup}}]$  was fascinating as it allowed to instantiate the random oracle in all previously mentioned applications. (Reset-security, UCE2 aka.  $\text{UCE}[\mathcal{S}^{\text{crs}}]$ , was only used in a single application and could hence be regarded as a special purpose assumption.) Formally, we denote by  $\mathcal{S}^{\text{cup}}$  the class of all computationally unpredictable sources and call a source  $S$  *computationally unpredictable* if the advantage of any PPT predictor  $P$ , defined by

$$\text{Adv}_{S,P}^{\text{pred}}(\lambda) := \Pr \left[ \text{Pred}_S^P(\lambda) \right],$$

is negligible, where game  $\text{Pred}_S^P(\lambda)$  is shown in Figure 3 on the right. For a formal definition of reset-security we refer to [BHK13b].

Somewhat regrettably, shortly after the UCE framework was published, Brzuska, Farshim and Mittelbach (BFM) showed that the notion  $\text{UCE}[\mathcal{S}^{\text{cup}}]$  is mutually exclusive with the existence of indistinguishability obfuscators [BFM14]. As candidate constructions for indistinguishability obfuscation are known while no candidate constructions for  $\text{UCE}[\mathcal{S}^{\text{cup}}]$ -secure hash functions are known the BFM result is usually interpreted in favor of indistinguishability obfuscation and thus raising the question as to the existence of UCE-secure functions. Subsequent to the BFM result several new (and weaker) notions of UCE security have been proposed to work around the BFM impossibility result. Most notably are the notions

**UCEs for statistical unpredictable sources.** The notion denoted by  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  is defined analogously to  $\text{UCE}[\mathcal{S}^{\text{cup}}]$  with the exception that sources need to be statistically unpredictable, that is, the predictor  $P$  in game  $\text{Pred}$  is allowed to run in unbounded time. The notion was proposed by BFM [BFM14] and independently by BHK [BHK13b] and turns out to be almost as universal as UCE1 back when UCEs were introduced.

**UCEs for strongly unpredictable sources.** Strong unpredictability is a strengthening of the unpredictability notion requiring that the source’s queries remain unpredictable even if the predictor gets to see a set containing all of the HASH oracle’s answers. Strong unpredictability was defined by Brzuska and Mittelbach [BM14b] who also show that strong unpredictability is a strictly larger class than the class defined via split sources (a structural requirement on sources defined in a later version by BHK [BHK13b]).

Formally, the restriction on sources captured by strong unpredictability is defined via the security game  $\text{stPred}$ :

---

<sup>4</sup>In the original formulation the UCE notions were called UCE1 (for computationally unpredictable sources) and UCE2 (for computationally reset-secure sources).

$\text{stPred}_{\mathcal{S}}^{\text{P}}(\lambda)$	$\text{HASH}(x)$
$X^*, Y^* \leftarrow \{\}$	$X^* \leftarrow X^* \cup \{x\}$
$L \leftarrow_{\mathcal{S}} \mathcal{S}^{\text{HASH}}(1^\lambda)$	$y \leftarrow_{\mathcal{S}} \{0, 1\}^{\text{H.ol}(\lambda)}$
$x' \leftarrow_{\mathcal{S}} \mathcal{P}^{\text{HASH}}(1^\lambda, L, Y^*)$	$Y^* \leftarrow Y^* \cup \{y\}$
<b>return</b> $(x' \in X^*)$	<b>return</b> $y$

Similarly to plain unpredictability one can further differentiate between PPT predictors and unbounded predictors which gives rise to the UCE notions  $\text{UCE}[\mathcal{S}^{\text{s-cup}}]$  (UCEs for computationally strong-unpredictable sources) and its statistical counter-part  $\text{UCE}[\mathcal{S}^{\text{s-sup}}]$  (UCEs for statistically strong-unpredictable sources).

**UCEs for statistical reset-secure sources.** Similarly to unpredictability, reset-security can be further split into computationally and statistically secure sources as proposed by both BFM [BFM14] and BHK [BHK13b]. Reset-security is a strictly stronger notion than unpredictability [BHK13a] which allows sources to make predictable queries to some extent. For a detailed description we refer to [BHK13b].

In Table 1 we present the most prominent applications for UCEs and list the corresponding UCE notions.

### 3.2 UCE Constructions

BHK showed that the simple random oracle construction  $\text{H}^{\text{RO}}(\text{hk}, x) := \text{RO}(\text{hk}||x)$  is  $\text{UCE}[\mathcal{S}^{\text{cup}}]$  secure and suggested that also HMAC is  $\text{UCE}[\mathcal{S}^{\text{cup}}]$  when treating the compression function as a fixed-input length random oracle. The latter was later shown by Mittelbach to be the case [Mit14].

Standard model constructions for UCEs were first presented by Brzuska and Mittelbach [BM14b] who constructed a  $\text{UCE}[\mathcal{S}^{1\text{-query}} \cap \mathcal{S}^{\text{s-cup}}]$  secure function from indistinguishability obfuscation and point obfuscation secure in the presence of auxiliary inputs as well as a  $\text{UCE}[\mathcal{S}^{q\text{-query}} \cap \mathcal{S}^{\text{s-sup}}]$  secure function from indistinguishability obfuscation and a weaker variant of point obfuscation where the auxiliary information is required to hide the obfuscated point statistically. That is, in the computational strong unpredictability setting BM showed how to obtain a UCE function which is secure against sources that make a constant number of queries yielding a standard model construction of a universal hardcore function for any one-way function. In the statistical strong unpredictability setting BM constructed a UCE secure function coping with sources making  $q$ -many queries for an arbitrary polynomial  $q$ .

### 3.3 Bitwise UCEs

In this paper we make a UCE notion explicit which was implicitly already considered by BHK in their original formulation of UCEs [BHK13a], namely, UCEs that output only a single bit, i.e., for which  $\text{ol}(\lambda) = 1$ . As the output length of UCEs will play a crucial factor in this paper we extend the UCE notation to denote the output length as subscript: we write  $\text{UCE}_1$  to denote functions that have a single bit output and simply UCE for functions which have arbitrary (polynomial) output length.

BHK show that for unpredictable UCEs (both for computational unpredictability and statistical unpredictability) we can extend the output length of the function by running it in counter mode. That is, given a function  $\text{H}$  with output length  $\text{H.ol}$  we can construct a function  $\text{H}$  with output length  $\ell$  as

```


$$\overline{H}(\mathbf{hk}, x)$$



---


for  $i = 1, \dots, \lceil \ell / H.\text{ol}(\lambda) \rceil$  do
   $y_i \leftarrow \langle i \rangle_{\log \lceil \ell / H.\text{ol}(\lambda) \rceil} \| x$ 
   $h_i \leftarrow H.\text{Eval}(\mathbf{hk}, y_i)$ 
 $h \leftarrow h_1 \| \dots \| h_\ell$ 
return  $h[1, \ell]$ 

```

**Theorem 3.1 (Theorem 4.6 [BHK13b])** *For any function  $H \in \text{UCE}_{H.\text{ol}}[\mathcal{S}^{\text{cup}}]$  we can construct  $\overline{H} \in \text{UCE}_\ell[\mathcal{S}^{\text{cup}}]$  for an arbitrary function  $\ell$  that is upper bounded by a polynomial. The same holds for statistical unpredictability, i.e.,  $\text{UCE}[\mathcal{S}^{\text{sup}}]$ .*

When we consider query-restricted sources such as  $\mathcal{S}^{1\text{-query}}$  or  $\mathcal{S}^{q\text{-query}}$  then one needs to be careful with the above theorem. In case we consider sources making  $q$ -queries then we get that

$$H \in \text{UCE}_{H.\text{ol}}[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{q\text{-query}}] \implies \overline{H} \in \text{UCE}_\ell[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{p\text{-query}}]$$

where  $p$  depends on  $q$  and the extension factor, i.e.,

$$p(\lambda) = \left\lfloor \frac{q(\lambda)}{\lceil \ell / H.\text{ol}(\lambda) \rceil} \right\rfloor.$$

In particular this means, that when we want to move from a constant output length to a polynomial output length this is not possible if we consider sources which are restricted to making a constant number of queries. By the above theorem we cannot argue that  $H \in \text{UCE}_1[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{1\text{-query}}]$  implies that  $\overline{H} \in \text{UCE}[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{1\text{-query}}]$  and, indeed, it seems unlikely that this implication holds.

## 4 From Strong Unpredictability to (Plain) Unpredictability

In the following section we present the first main result of this paper which, although simple to prove, has intriguing consequences. We show that for bit-wise UCEs, i.e.,  $\text{UCE}_1$  the strong unpredictability and (plain) unpredictability restrictions are equivalent. This holds both for the statistical case and for the computational case.

**Theorem 4.1** *For any function  $H$  with  $H.\text{ol}(\lambda) = 1$  it holds that*

1.  $H$  is UCE secure against computationally unpredictable sources  $\mathcal{S}^{\text{cup}}$  if and only if it is UCE secure against computatioanlly strong unpredictable sources  $\mathcal{S}^{\text{s-cup}}$ :

$$H \in \text{UCE}_1[\mathcal{S}^{\text{cup}}] \iff H \in \text{UCE}_1[\mathcal{S}^{\text{s-cup}}]$$

2.  $H$  is UCE secure against statistically unpredictable sources  $\mathcal{S}^{\text{sup}}$  if and only if it is UCE secure against statistically strong unpredictable sources  $\mathcal{S}^{\text{s-sup}}$ :

$$H \in \text{UCE}_1[\mathcal{S}^{\text{sup}}] \iff H \in \text{UCE}_1[\mathcal{S}^{\text{s-sup}}]$$

The two statements above hold also in the case the number of queries a source can make is restricted to some number  $q$ , that is

$$\begin{aligned} H \in \text{UCE}_1[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{q\text{-query}}] &\iff H \in \text{UCE}_1[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}] \\ H \in \text{UCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}] &\iff H \in \text{UCE}_1[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}] \end{aligned}$$

*Proof.* In order to prove the above theorem recall that the difference between unpredictability and strong unpredictability resides solely in the information the predictor is given. The following description captures both forms of unpredictability, the boxed statements showing the additional information given to the predictor in the strong unpredictability game:

$\text{Pred}_S^P(\lambda)$	$\boxed{\text{stPred}_S^{P_*}(\lambda)}$	$\text{HASH}(x)$
$X^* \leftarrow \{\};$	$\boxed{Y^* \leftarrow \{\}}$	$X^* \leftarrow X^* \cup \{x\}$
$L \leftarrow_{\S} \mathcal{S}^{\text{HASH}}(1^\lambda)$		$y \leftarrow_{\S} \{0, 1\}^{\text{H.oI}(\lambda)}$
$x' \leftarrow_{\S} \mathcal{P}_{\boxed{*}}^{\text{HASH}}(1^\lambda, L, \boxed{Y^*})$		$\boxed{Y^* \leftarrow Y^* \cup \{y\}}$
<b>return</b> $(x' \in X^*)$		<b>return</b> $y$

In the strong unpredictability game predictor  $P_*$  gets a set containing the answers from the HASH oracle. It is important to emphasize that this is indeed a set and not a list. Hence when we consider bit-UCEs  $Y^*$  can only take one of 4 forms:

$$Y^* \in \left\{ \{\}, \{0\}, \{1\}, \{0, 1\} \right\}$$

As the correct  $Y^*$  can be guessed with probability  $\frac{1}{4}$  we hence get that from any predictor  $P_*$  in game  $\text{stPred}$  we can construct a predictor  $P$  in game  $\text{Pred}$  such that

$$\text{Adv}_{S, P_*}^{\text{stpred}}(\lambda) = \frac{1}{4} \cdot \text{Adv}_{S, P}^{\text{pred}}(\lambda).$$

Similarly, any *good* predictor  $P$  in game  $\text{Pred}$  immediately yields a *good* predictor in game  $\text{stPred}$  which concludes the proof of Theorem 4.1.  $\square$

Combining Theorems 3.1 and Theorem 4.1 we obtain several interesting corollaries. Given that we can extend the length of a UCE function by sacrificing on the number of queries that a source can make we get

**Corollary 4.2** *It holds that*

$$\begin{aligned} \text{UCE}_1[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}] &\implies \text{UCE}[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{p\text{-query}}] \\ \text{UCE}_1[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}] &\implies \text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{p\text{-query}}] \end{aligned}$$

*The implications are to be read as: if there exists  $H$  that is  $\text{UCE}_1[X]$  secure, then there exists  $\bar{H}$  that is  $\text{UCE}[Y]$  secure.*

Brzuska, Farshim and Mittelbach [BFM14] show that indistinguishability obfuscation and  $\text{UCE}[\mathcal{S}^{\text{cup}} \cap \mathcal{S}^{1\text{-query}}]$  are mutually exclusive. Combining this with Corollary 4.2 yields that also  $\text{UCE}_1[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}]$  and in particular  $\text{UCE}[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}]$  are mutually exclusive with indistinguishability obfuscation.

**Corollary 4.3** *If indistinguishability obfuscation exists, then  $\text{UCE}_1[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}]$  (and  $\text{UCE}[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{q\text{-query}}]$ ) security cannot be achieved in the standard model.*

Brzuska and Mittelbach [BM14b] construct a  $\text{UCE}[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{1\text{-query}}]$  secure function from indistinguishability obfuscation and strong variants of point obfuscation. Note that the above negative results only apply to UCEs secure against strongly computationally unpredictable sources as soon as there is a superlogarithmic number of queries, because then, the transformation via  $\text{UCE}_1$  yields UCEs secure against computationally unpredictable sources, where the UCE function has a superpolynomial number

of output bits which is crucial for the iO-based attack to work (because it applies a PRG to the output). Hence, the BM construction for computationally strongly unpredictable sources, that make only a *constant* number of queries is, in some sense, optimal. There is a small gap that might allow for an additional positive result for computationally strongly unpredictable sources that make a *logarithmic* number of queries. However, it is not known that a logarithmic number of queries would allow for more applications.

## 5 Constructing $q$ -Query $\text{UCE}[\mathcal{S}^{\text{sup}}]$ Secure Functions

Having established how bit-wise UCEs for strongly unpredictable sources relate to normal UCEs for unpredictable sources it remains to construct a  $\text{UCE}_1[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}]$  in order to obtain a  $q$ -query  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  secure function, that is, a  $\text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  secure function.  $q$ -query  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  suffices to build variants of key-dependent message secure encryption schemes, related-key secure encryption schemes, CCA-secure public key encryption, PRG immunizers and multi-bit output point-function obfuscation as well as  $q$ -query variants of correlated input-secure functions, message-locked encryption and deterministic public-key encryption [BHK13b, MH14b, BH15]. Noteworthy is the relation between  $\text{UCE}[\mathcal{S}^{\text{sup}}]$  and point function obfuscation, which we will discuss in detail in Section 8.

Brzuska and Mittelbach (BM) construct a  $\text{UCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}]$  secure function from indistinguishability obfuscation and a variant of point function obfuscation, namely, composable AIPO for statistically unpredictable distributions. It is, however, not clear whether we can simply truncate their construction to obtain a  $\text{UCE}_1[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}]$  and hence a  $q$ -query  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$  secure function. At least, such a transformation is not obvious if one wants to use a black-box proof of security. Note that when the hash functions have large outputs (which is the case considered by BM) then it is equivalent to consider whether the predictor is given the *set* of oracle answers (which is as in the definition of strong unpredictability) or the *multi-set* (that is, for each answer it is also conveyed how often it appeared). When considering hash functions with long outputs the question of set vs. multi-set does not come up since collisions on oracle answers occur only with negligible probability. However, for shorter output lengths where collisions may appear the proof strategy by BM only works when considering multi-sets.

Interestingly, when considering hash functions with a single bit, we can simulate the multi-set view by guessing the number of ones and the number of zeroes in the multi-set, which allows us to prove that the BM construction, indeed yields a  $q$ -query  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$  secure function. Subsequently in Section 6 we show how to extend the proof in order to obtain also a multi-key UCE version, i.e.,  $\text{mUCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$ .

### 5.1 The BM Construction

Let us recall the construction of a  $\text{UCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{q\text{-query}}]$  presented by Brzuska and Mittelbach [BM14b]. Key generation of their construction crucially depends on a polynomial  $q$  which describes the number of queries a source is allowed to make.

**Construction 1** *Let  $q : \mathbb{N} \rightarrow \mathbb{N}$  and  $s : \mathbb{N} \rightarrow \mathbb{N}$  be polynomials. Let  $G$  be a puncturable PRF and let  $\text{iO}$  be an indistinguishability obfuscator for all circuits in  $\text{P/poly}$ . We define our hash function family  $\text{H}$  as*

$\text{H.KGen}(1^\lambda)$	$\text{H.Eval}(\text{hk}, x)$
$k \leftarrow_{\S} G.\text{KGen}(1^\lambda)$	$\bar{C} \leftarrow \text{hk}$
$\bar{C} \leftarrow_{\S} \text{iO}(\text{PAD}(s(\lambda), G.\text{Eval}(k, \cdot)))$	<b>return</b> $\bar{C}(x)$
$\text{hk} \leftarrow \bar{C}$	
<b>return</b> $\text{hk}$	

where  $\text{PAD} : \mathbb{N} \times \{0, 1\}^* \rightarrow \{0, 1\}^*$  denotes a deterministic padding algorithm that takes as input an integer and a description of a circuit  $C$  and outputs a functionally equivalent circuit padded to length  $|C| + s(\lambda)$ . Function  $s$  needs to be chosen in accordance with the puncturable PRF and a point obfuscation scheme AIPO to allow for puncturing  $G$  on  $q$  points and embedding  $q$  many point obfuscations within circuit  $\bar{C}$ .

## 5.2 Extending the BM Construction

In the following we show that the BM construction when implemented with a puncturable PRF that has a single output bit also yields a secure  $q$ -query  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$  function. The security proof is similar to the proof in [BM14b] with some simplifications and some modifications especially in the last game hop.

**Construction 2** *The construction is identical to Construction 1 with the exception that PRF  $G$  has an output length of  $G.\text{ol}(\lambda) = 1$ .*

**Theorem 5.1** *If indistinguishability obfuscation exists and if composable AIPO for statistically unpredictable distributions (composable  $\text{AIPO}[\mathcal{D}^{\text{sup}}]$ ) exist then Construction 2 is  $q$ -query  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$  (i.e.,  $\text{UCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$ ) secure.*

We prove the theorem via a sequence of 5 games (depicted in Figure 4) where game  $\text{Game}_1$  denotes the original  $\text{UCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  game with hidden bit  $b$  fixed to 1. We first present the games and subsequently the analysis of the individual game hops. Let  $\mathbf{S} \in \mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}$ . Without loss of generality we assume that source  $\mathbf{S}$  does not repeat queries to its oracle  $\text{HASH}$ .

**Game<sub>1</sub>:** The first game is the original  $\text{UCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$ -game with hidden bit  $b$  set to 1. Here, the hash key  $\text{hk}$  is an obfuscation of the circuit  $C_1[\mathbf{k}](x) := G.\text{Eval}(\mathbf{k}, x)$  where  $\mathbf{k}$  is a key for the puncturable PRF.

**Game<sub>2</sub>:** Let  $X_0^*$  denote the queries by source  $\mathbf{S}$  to its  $\text{HASH}$  oracle that were answered with 0 and let  $X_1^*$  denote the queries answered with 1.  $\text{Game}_2$  is identical to  $\text{Game}_1$  with the exception that circuit  $C_2[\mathbf{k}^*, P_0, P_1]$  is obfuscated instead of circuit  $C_1[\mathbf{k}]$ . Here  $\mathbf{k}^*$  is the punctured key for set  $X_0^* \cup X_1^*$  (i.e., punctured on all of the queries made by  $\mathbf{S}$ ). Sets  $P_d$  contain point obfuscations for the queries in  $X_d$ . Finally, circuit  $C_2[\mathbf{k}^*, P_0, P_1]$  is functionally equivalent to  $C_1[\mathbf{k}]$ . On input  $x$  it first checks whether any point obfuscation in  $P_d$  outputs 1 on  $x$  (note this can be at most one). If so it outputs  $d$ . Otherwise it uses the punctured key to output  $G.\text{Eval}(\mathbf{k}^*, x)$ . Note that in this case  $x \notin X_0^* \cup X_1^*$  and hence  $G.\text{Eval}(\mathbf{k}^*, x) = G.\text{Eval}(\mathbf{k}, x)$ .

**Game<sub>3</sub>:** The game is equivalent to  $\text{Game}_2$  except that oracle  $\text{HASH}$  now samples  $y$  uniformly at random instead of invoking  $G.\text{Eval}(\mathbf{k}, \cdot)$ .

**Game<sub>4</sub>:** The game is equivalent to the previous game except that we now use an obfuscation of circuit  $C_3[\mathbf{k}, P_0, P_1]$ . The circuit is functionally equivalent to  $C_2[\mathbf{k}^*, P_0, P_1]$  as the puncturable PRF is only called on values that were not punctured out.

**Game<sub>5</sub>:** The game is equivalent to the previous game except that now an obfuscation of circuit  $C_4[\mathbf{k}]$ . Circuit  $C_4[\mathbf{k}]$  is our original circuit again, that is,  $C_4[\mathbf{k}](\cdot) := G.\text{Eval}(\mathbf{k}, \cdot)$ .  $\text{Game}_5$  is our intended target. It is the UCE-security game for our construction in the random oracle world (that is, oracle  $\text{HASH}$  implements a random oracle).

In  $\text{Game}_5$  we have reached the target setting, i.e., the UCE-game with the hidden bit set to 0. It remains to show that the individual games are negligibly close. In Figure 4 we present the reduction target for each of the game hops above the pseudocode for each game. We present a formal analysis of the the individual game hops in Appendix A.

	iO	PRF	iO	AIPO[ $\mathcal{D}^{\text{sup}}$ ] + iO+ [BCP14]
Game <sub>1</sub> ( $\lambda$ )	Game <sub>2</sub> ( $\lambda$ )	Game <sub>3</sub> ( $\lambda$ )	Game <sub>4</sub> ( $\lambda$ )	Game <sub>5</sub> ( $\lambda$ )
1 :	$X_0^*, X_1^*, P_0, P_1 \leftarrow \{\}$	$X_0^*, X_1^*, P_0, P_1 \leftarrow \{\}$	$X_0^*, X_1^*, P_0, P_1 \leftarrow \{\}$	$X_0^*, X_1^*, P_0, P_1 \leftarrow \{\}$
2 :	$k \leftarrow \$ G.\text{KGen}(1^\lambda)$	$k \leftarrow \$ G.\text{KGen}(1^\lambda)$	$k \leftarrow \$ G.\text{KGen}(1^\lambda)$	$k \leftarrow \$ G.\text{KGen}(1^\lambda)$
3 :	$L \leftarrow \$ \mathcal{S}^{\text{HASH}}(1^\lambda)$	$L \leftarrow \$ \mathcal{S}^{\text{HASH}}(1^\lambda)$	$L \leftarrow \$ \mathcal{S}^{\text{HASH}}(1^\lambda)$	$L \leftarrow \$ \mathcal{S}^{\text{HASH}}(1^\lambda)$
4 :		<b>for</b> $d \in \{0, 1\}, x \in X_d^*$ <b>do</b>	<b>for</b> $d \in \{0, 1\}, x \in X_d^*$ <b>do</b>	
5 :		$p \leftarrow \$ \text{AIPO}(x)$	$p \leftarrow \$ \text{AIPO}(x)$	
6 :		$P_d \leftarrow P_d \cup \{p\}$	$P_d \leftarrow P_d \cup \{p\}$	
7 :		$k^* \leftarrow \$ G.\text{Puncture}(k, X_0 \cup X_1)$		
8 :	$\bar{C} \leftarrow \$ \text{iO}(C_1[k])$	$\bar{C} \leftarrow \$ \text{iO}(C_2[k^*, P_0, P_1])$	$\bar{C} \leftarrow \$ \text{iO}(C_3[k, P_0, P_1])$	$\bar{C} \leftarrow \$ \text{iO}(C_4[k])$
9 :	$\text{hk} \leftarrow \bar{C}$	$\text{hk} \leftarrow \bar{C}$	$\text{hk} \leftarrow \bar{C}$	$\text{hk} \leftarrow \bar{C}$
10 :	$b' \leftarrow \$ \mathcal{D}(1^\lambda, \text{hk}, L)$	$b' \leftarrow \$ \mathcal{D}(1^\lambda, \text{hk}, L)$	$b' \leftarrow \$ \mathcal{D}(1^\lambda, \text{hk}, L)$	$b' \leftarrow \$ \mathcal{D}(1^\lambda, \text{hk}, L)$
11 :	<b>return</b> $(1 = b')$	<b>return</b> $(1 = b')$	<b>return</b> $(1 = b')$	<b>return</b> $(1 = b')$
<hr/>				
HASH( $x$ )	HASH( $x$ )	HASH( $x$ )	HASH( $x$ )	HASH( $x$ )
1 :	$y \leftarrow G.\text{Eval}(k, x)$	$y \leftarrow G.\text{Eval}(k, x);$ <span style="border: 1px solid black; padding: 2px;"><math>y \leftarrow \\$ \{0, 1\}</math></span>	$y \leftarrow \$ \{0, 1\}$	$y \leftarrow \$ \{0, 1\}$
2 :		$X_y^* \leftarrow X_y^* \cup \{x\}$	$X_y^* \leftarrow X_y^* \cup \{x\}$	
3 :	<b>return</b> $y$	<b>return</b> $y$	<b>return</b> $y$	<b>return</b> $y$
<hr/>				
CIRCUIT $C_1[k](x)$	CIRCUIT $C_2[k^*, P_0, P_1](x)$	CIRCUIT $C_3[k, P_0, P_1](x)$	CIRCUIT $C_4[k](x)$	
1 :	<b>for</b> $d \in \{0, 1\}, p \in P_d$ <b>do</b>	<b>for</b> $d \in \{0, 1\}, p \in P_d$ <b>do</b>		
2 :	<b>if</b> $p(x) = 1$ <b>then</b>	<b>if</b> $p(x) = 1$ <b>then</b>		
3 :	<b>return</b> $d$	<b>return</b> $d$		
4 :	<b>return</b> $G.\text{Eval}(k, x)$	<b>return</b> $G.\text{Eval}(k^*, x)$	<b>return</b> $G.\text{Eval}(k, x)$	<b>return</b> $G.\text{Eval}(k, x)$

Figure 4: A pseudo code description of the game steps in Theorem 5.1. The boxed Game<sub>3</sub> is to be understood with the boxed statement included. Furthermore the highlighted lines are emphasize the difference to the previous game.

## 6 Multi-key UCEs

Besides UCEs with a single hash key  $\text{hk}$ , BHK also define a multi-key version called mUCE which works analogously as plain UCEs with the exception that source  $\mathcal{S}$  can decide with how many keys it wants to work and oracle HASH takes as input an index specifying the key [BHK13a]. Also similarly to plain UCEs the source needs to be restricted and we can consider similar restrictions, namely unpredictable and strongly unpredictable sources both their statistical and their computational variants. In Figure 5 we give the pseudocode of the mUCE game and the accompanying unpredictability game mPred.

The relationship between UCEs and multi-key UCEs is still an open research question. While it is easy to see that  $\text{mUCE}[\mathcal{S}^{\text{cup}}]$  (and its statistical variant) imply the corresponding single key variants the other direction is not known to hold. On the other hand, we are not aware of any separating example and believe that such an example would be interesting. The construction in this paper as well as the UCE constructions in idealized models [BHK13a, Mit14] both also achieve multi-key security.

**Theorem 6.1** *If indistinguishability obfuscation exists and if composable AIPO for statistically unpredictable distributions (composable  $\text{AIPO}[\mathcal{D}^{\text{sup}}]$ ) exist then Construction 2 is  $\text{mUCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  secure.*

Note that we do not need to take care of the number of keys in the construction since for each key we allow at most  $q$ -many queries and each key corresponds to a new circuit.

$\text{mUCE}_{\mathbb{H}}^{\text{S,D}}(\lambda)$	$\text{HASH}(x, i)$	$\text{mPred}_{\mathbb{S}}^{\text{P}}(\lambda)$	$\text{HASH}(x, i)$
$(1^n, \text{state}) \leftarrow_{\mathbb{S}} \mathbb{S}(1^\lambda, \varepsilon)$ $b \leftarrow_{\mathbb{S}} \{0, 1\}$ <b>for</b> $i = 1, \dots, n$ , <b>do</b> $\text{hk}[i] \leftarrow_{\mathbb{S}} \text{H.KGen}(1^\lambda)$ $L \leftarrow_{\mathbb{S}} \mathbb{S}^{\text{HASH}}(1^n, \text{state})$ $b' \leftarrow_{\mathbb{S}} \mathbb{D}(1^\lambda, \text{hk}, L)$ <b>return</b> $(b = b')$	<b>if</b> $T[x, i] = \perp$ <b>then</b> <b>if</b> $b = 1$ <b>then</b> $T[x, i] \leftarrow \text{H.Eval}(\text{hk}[i], x)$ <b>else</b> $T[x, i] \leftarrow_{\mathbb{S}} \{0, 1\}^{\text{H.ol}(\lambda)}$ <b>return</b> $T[x, i]$	$(1^n, \text{state}) \leftarrow_{\mathbb{S}} \mathbb{S}(1^\lambda, \varepsilon)$ <b>done</b> $\leftarrow$ <b>false</b> ; $Q \leftarrow \{\}$ $L \leftarrow_{\mathbb{S}} \mathbb{S}^{\text{HASH}}(1^n, \text{state})$ <b>done</b> $\leftarrow$ <b>true</b> $Q' \leftarrow_{\mathbb{S}} \mathbb{P}^{\text{HASH}}(1^\lambda, 1^n, L)$ <b>return</b> $(Q \cap Q' \neq \{\})$	<b>if</b> <b>done</b> = <b>false</b> <b>then</b> $Q \leftarrow Q \cup \{x\}$ <b>if</b> $T[x, i] = \perp$ <b>then</b> $T[x, i] \leftarrow_{\mathbb{S}} \{0, 1\}^{\text{H.ol}(\lambda)}$ <b>return</b> $T[x, i]$

Figure 5: The multi-key UCE game (mUCE) on the left and the corresponding unpredictability game on the right. Similarly to the plain UCE game we consider two variants of unpredictability: computationally unpredictable sources where predictor  $\mathbb{P}$  is restricted to be PPT and statistically unpredictable sources where predictor  $\mathbb{P}$  is unbounded.

The proof is analogous to the proof for the single-keyed version. Instead of a single key, now  $t$  keys need to be handled and constructed. The number  $t$  of keys in the system can, for example, be upper bounded by the runtime of the source  $\mathbb{S}$  allowing us to keep it fixed (the source may use only a fraction of the keys, if it so chooses). In the game steps, the sets  $X_0^*, X_1^*, P_0, P_1$  are multiplied by the number of keys in the system. The first game hop works analogously while the second game hop can be done one key at a time. The third game hop again is analogously as in the single-key version and the final game hop can again be done one key at a time. We have depicted the final game hop in Figure 6. Game  $\text{Game}_4^j$  is an intermediate game with  $\text{Game}_4^1 = \text{Game}_4$  and  $\text{Game}_4^{t+1} = \text{Game}_5$ . In  $\text{Game}_4^j$  the first  $j - 1$  keys are already transformed to be generated as an obfuscation of the correct circuit  $C_4[k]$ . The difference between  $\text{Game}_4^j$  and  $\text{Game}_4^{j+1}$  is hence that key  $\text{hk}[j]$  is either generated as

$$\text{iO}(C_3[\mathbf{k}[i], P_0^i, P_1^i])$$

or as

$$\text{iO}(C_4[\mathbf{k}[i]])$$

which is almost an identical situation as in the single-key game. Hence we have that

$$\left| \Pr \left[ \text{Game}_4^j(\lambda) \right] - \Pr \left[ \text{Game}_4^{j+1}(\lambda) \right] \right| \leq \text{Adv}_{\text{iO}, \text{Sam}, \text{Dist}}^{\text{iO}}(\lambda) + [\text{BCP14}] \leq \text{negl}(\lambda)$$

## 7 The Superfluous Padding Assumption (SuPA)

In the following section we explore the role of padding for the security of Construction 2 and propose the *Superfluous Padding Assumption* (SuPA). Consider a circuit  $C$  and the same circuit padded to size  $|C| + s$ , which we denote by  $\text{PAD}(s, C)$ . Intuitively, if for some obfuscation scheme  $\text{Obf}(\text{PAD}(s, C))$  hides some information about an auxiliary value  $\text{aux}$  then so should an obfuscation of the unpadded circuit. In other words, if we consider pairs two pairs  $(\text{aux}, C)$  and  $(\text{aux}', C')$  and we can prove that if we first pad the circuit to some length  $|C| + s$  and then apply obfuscation that then the two resulting distributions are indistinguishable then also the distributions where we omit the padding step should be indistinguishable. More formally, we would like to argue that

$$(\text{aux}, \text{Obf}(\text{PAD}(s, C))) \approx_c (\text{aux}', \text{Obf}(\text{PAD}(s, C'))) \implies (\text{aux}, \text{Obf}(C)) \approx_c (\text{aux}', \text{Obf}(C')) \quad (2)$$

Coming back to our construction of UCEs it is easy to see that if the implication in (2) holds, then this allows us to argue that padding in Construction 2 is superfluous and the construction is, indeed, secure without any bound  $q$ . In the following we make the above intuition formal.

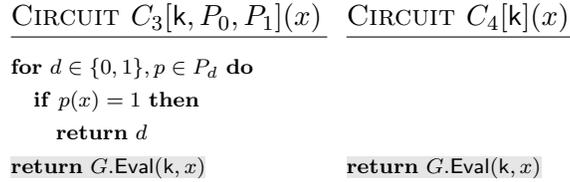
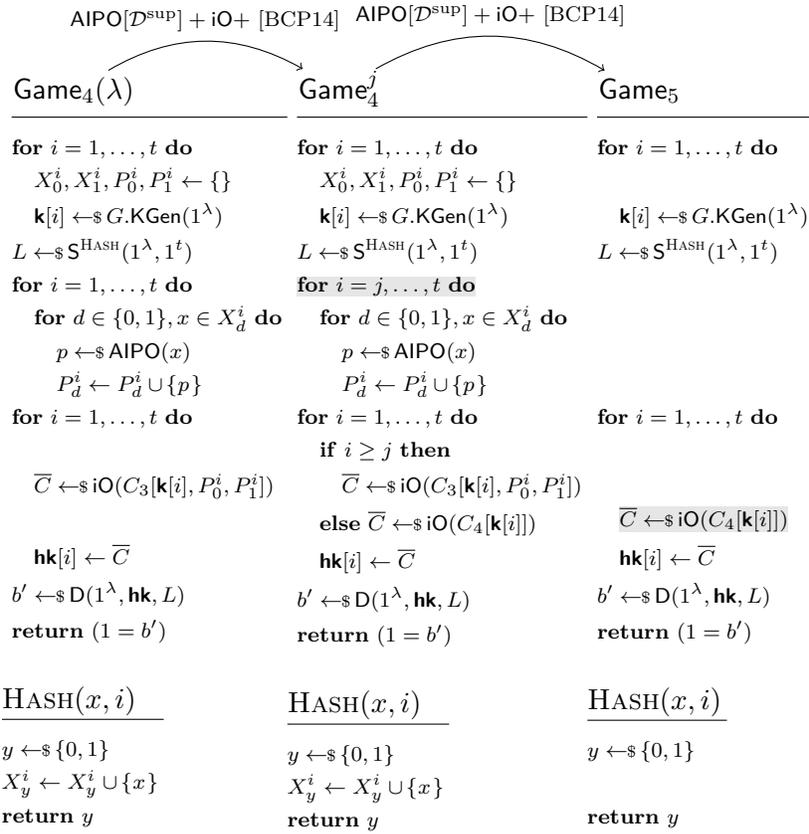


Figure 6: Relevant game steps for proof of Theorem 6.1.

## 7.1 The Superfluous Padding Assumption

In the following we explore the above intuition and present a novel and non-standard assumption called the *Superfluous Padding Assumption* (SuPA) which captures the above intuition: if an obfuscation for a padded circuit is *good* then so is the obfuscation of the unpadded circuit. As the naming suggest we are not able to prove the assumption (yet) for indistinguishability obfuscation, the case required in order to lift the  $q$ -bound from Construction 2. We do, however, make a first step in showing that the assumption holds in the stronger virtual black-box setting. The assumption itself can be stated for any obfuscator and is parameterized by a class of efficient samplers.

We state our assumption in two steps. In the first step we define admissible samplers, where an efficient sampler outputs some auxiliary information  $\text{aux}$  and a circuit  $C$ . We call two samplers admissible if their auxiliary informations are computationally indistinguishable relative to an obfuscation of the padded circuit for some polynomial padding.

**Definition 7.1** *Let Obf be an obfuscation scheme and let  $\text{PAD} : \mathbb{N} \times \{0, 1\}^* \rightarrow \{0, 1\}^*$  be a deterministic padding algorithm that takes as input an integer  $s$  and a description of a circuit  $C$  and outputs a*

$$\frac{\text{SuP}[s]_{\text{Obf}, \text{Sam}_0, \text{Sam}_1}^{\text{D}}(\lambda)}{}$$

```

b  $\leftarrow_s \{0, 1\}$ 
(aux, C)  $\leftarrow_s \text{Sam}_b(1^\lambda)$ 
if  $s(\lambda) > 0$  then
   $\bar{C} \leftarrow_s \text{Obf}(\text{PAD}(s(\lambda), C))$ 
else  $\bar{C} \leftarrow_s \text{Obf}(C)$ 
b'  $\leftarrow_s \text{D}(1^\lambda, s, \text{aux}, \bar{C}, |C|)$ 
return  $b = b'$ 

```

Figure 7: The SuP game is parameterized by a polynomial  $s$ . It runs one of two samplers which output auxiliary information  $\text{aux}$  and a circuit  $C$ . Then according to  $s$  the circuit is padded (if  $s = 0$  then the original circuit is used) before it is obfuscated and given to distinguisher  $D$  which additionally gets as input auxiliary input  $\text{aux}$  as well as  $s$  and the size of the original circuit  $|C|$ . The task of distinguisher  $D$  is to guess from which sampler auxiliary input  $\text{aux}$  and circuit  $C$  came from.

*functionally equivalent circuit of size  $s + |C|$ . We say that a pair of PPT samplers  $(\text{Sam}_0, \text{Sam}_1)$  is SuP-admissible for obfuscator  $\text{Obf}$ , if there exists a polynomial  $s$  such that for any PPT distinguisher  $D$  its advantage in the  $\text{SuP}[s]$  game (formalized in Figure 7) is negligible:*

$$\text{Adv}_{\text{Obf}, \text{Sam}_0, \text{Sam}_1, \text{D}}^{\text{sup}[s]}(\lambda) = 2 \cdot \Pr \left[ \text{SuP}[s]_{\text{Obf}, \text{Sam}_0, \text{Sam}_1}^{\text{D}}(\lambda) \right] - 1 \leq \text{negl}(\lambda)$$

SuP-admissibility does not place restrictions on samplers beyond indistinguishability, but potentially, one can put various reasonable additional restrictions on the samplers. One could, for example, require that the marginal distribution on the circuits is identical, or that  $\text{aux}$  is generated only with oracle access to the functionality provided by circuit  $C$  and thus cannot depend on the description of  $C$ , but only on the functionality of  $C$ . In the following we state the *Superfluous Padding Assumption* (SuP assumption) in its most general form, without requiring that samplers are from some specific subclass of admissible samplers.

**Assumption 7.2** *Let  $\text{Obf}$  be an obfuscation scheme and let  $\text{Sam}_0$  and  $\text{Sam}_1$  be two SuP-admissible samplers. Then, the Superfluous Padding Assumption states that no efficient distinguisher  $D$  has a non-negligible advantage in the  $\text{SuP}[0]$  game without padding:*

$$\text{Adv}_{\text{Obf}, \text{Sam}_0, \text{Sam}_1, \text{D}}^{\text{sup}[0]}(\lambda) = 2 \cdot \Pr \left[ \text{SuP}[0]_{\text{Obf}, \text{Sam}_0, \text{Sam}_1}^{\text{D}}(\lambda) \right] - 1 \leq \text{negl}(\lambda).$$

## 7.2 Applying the SuP Assumption

Intuitively the SuP assumption states that if two distributions are indistinguishable relative to an obfuscated circuit  $C$  which was padded before obfuscation, then the two distributions are also indistinguishable relative to the obfuscated circuit  $C$  without padding. Or in other words, if an obfuscation of a padded circuit hides something, then so does an obfuscation of the unpadded circuit. Before we further analyze the assumption let us establish how it lifts Construction 2 from achieving  $\text{UCE}_1[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  to achieving  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$ .

In the proof of Theorem 5.1 we show that for any source  $S$ , there exists a polynomial  $s$  such that for no adversary can distinguish the following two distributions

$$\left( L_1, \text{iO}(\text{PAD}(s(\lambda), G.\text{Eval}(k, \cdot))) \right) \approx_c \left( L_5, \text{iO}(\text{PAD}(s(\lambda), G.\text{Eval}(k, \cdot))) \right)$$

where  $L_1$  is generated as in  $\text{Game}_1$  by running source  $S$  relative to the actual construction and  $L_5$  is generated as in  $\text{Game}_5$  by running source  $S$  relative to a random oracle. Applying the SuP assumption

for indistinguishability obfuscation and identifying  $\text{Sam}_0$  with the sampler that runs  $\text{Game}_1$  until, and including, line 9 and outputting  $(L, \text{hk})$  (see Figure 4) and, similarly,  $\text{Sam}_1$  with the sampler that runs  $\text{Game}_5$  until line 9 we get that also

$$\left( L_1, \text{iO}(G.\text{Eval}(k, \cdot)) \right) \approx_c \left( L_5, \text{iO}(G.\text{Eval}(k, \cdot)) \right)$$

are computationally indistinguishable. Hence we get that:

**Theorem 7.3** *If indistinguishability obfuscation exists and if composable AIPO for statistically unpredictable distributions (composable AIPO $[\mathcal{D}^{\text{sup}}]$ ) exist, if the superfluous padding assumption holds for a secure indistinguishability obfuscation scheme, then Construction 2 is  $\text{UCE}_1[\mathcal{S}^{\text{sup}}]$  secure.*

Note that, in particular, it might be that indistinguishability obfuscation generically implies SuPA just as VBB obfuscation generically implies SuPA, as we will see soon.

**Remark.** Let us note that the full SuP assumption as stated above is not necessary for our UCE use-case as there we only consider a single pair of samplers  $(S_1, S_5)$  (where  $S_1$  is the source in  $\text{Game}_1$  and  $S_5$  is the source defined in  $\text{Game}_5$  in the proof of Theorem 5.1) where auxiliary information  $L$  is generated only with oracle access to the functionality of the circuits without using the description of the circuits. Furthermore, for the UCE case the circuit distribution is identical, that is, both samplers  $\text{Sam}_0$  and  $\text{Sam}_1$  generate circuit  $C$  as  $G.\text{Eval}(k, \cdot)$  for uniformly random key. Finally, for Construction 2 we can fix the padding operation and thus need the SuP assumption also only to hold for this fixed choice of padding.

### 7.3 On the Validity of the Superfluous Padding Assumption

The SuP assumption is a highly non-standard assumption, and we need to study it further to deepen our understanding and, hopefully, gain confidence in it. In this subsection, we validate the assumption in an idealized model, namely, we show that it holds for virtual black-box obfuscation. For indistinguishability obfuscation, different techniques are needed and we will discuss how and why we believe that SuPA might be reasonable to assume also for indistinguishability obfuscation.

**Theorem 7.4** *The Superfluous Padding Assumption holds for virtual black-box obfuscators.*

*Proof.* The idea of the proof is to exploit that the simulator of the virtual black-box obfuscator only has black-box access to the obfuscated circuit (and its size) and does not get the circuit itself. Hence, given an *obfuscated* version of the circuit, one can simulate the oracle for the simulator regardless of how the obfuscation looks like and whether the circuit was padded before being obfuscated or not.

Let  $\text{Obf}$  be a virtual black-box obfuscator and let  $\text{Sam}_0$  and  $\text{Sam}_1$  be a pair of SuP-admissible samplers. Let  $D$  be an efficient distinguisher. We need to prove that

$$\left| \Pr \left[ \left( \begin{array}{l} (\text{aux}_0, C_0) \leftarrow \text{Sam}_0(1^\lambda) \\ \overline{C}_0 \leftarrow \text{Obf}(C_0) \\ \text{return } D(1^\lambda, \text{aux}_0, \overline{C}_0, |C_0|) \end{array} \right) = 1 \right] - \Pr \left[ \left( \begin{array}{l} (\text{aux}_1, C_1) \leftarrow \text{Sam}_1(1^\lambda) \\ \overline{C}_1 \leftarrow \text{Obf}(C_1) \\ \text{return } D(1^\lambda, \text{aux}_1, \overline{C}_1, |C_1|) \end{array} \right) = 1 \right] \right| \quad (3)$$

is negligible. We start with the first distribution and transform it into the second via several game-hops. Let  $\text{Sim}$  be the virtual black-box simulator for  $D$ . Note that  $\text{Sim}$  depends on  $D$ , but that  $\text{Sim}$  does not depend on the circuit. By the security of virtual black-box obfuscation, we have that

$$\left| \Pr \left[ \left( \begin{array}{l} (\text{aux}_0, C_0) \leftarrow \text{Sam}_0(1^\lambda) \\ \overline{C}_0 \leftarrow \text{Obf}(C_0) \\ \text{return } D(1^\lambda, \text{aux}_0, \overline{C}_0, |C_0|) \end{array} \right) = 1 \right] - \Pr \left[ \left( \begin{array}{l} (\text{aux}_0, C_0) \leftarrow \text{Sam}_0(1^\lambda) \\ \text{return } \text{Sim}^{C_0}(1^\lambda, \text{aux}_0, |C_0|) \end{array} \right) = 1 \right] \right| \quad (4)$$

is negligible. Now, if we can prove that

$$\epsilon := \left| \Pr \left[ \left( \begin{array}{l} (\mathbf{aux}_0, C_0) \leftarrow \text{Sam}_0(1^\lambda) \\ \text{return Sim}^{C_0}(1^\lambda, \mathbf{aux}_0, |C_0|) \end{array} \right) = 1 \right] - \Pr \left[ \left( \begin{array}{l} (\mathbf{aux}_1, C_1) \leftarrow \text{Sam}_1(1^\lambda) \\ \text{return Sim}^{C_1}(1^\lambda, \mathbf{aux}_1, |C_1|) \end{array} \right) = 1 \right] \right| \quad (5)$$

is negligible, then we can use that (4) is negligible in the “backward” direction, i.e., we can replace the subscript 0 of  $\mathbf{aux}_0$ ,  $C_0$  and  $\text{Sam}_0$  in (4) by 1, because the simulator  $\text{Sim}$  works for all circuits  $C$  and only depends on  $D$ . Thereby, we then yield the left distribution in (3). Thus, it remains to show that  $\epsilon$  is negligible. Assume not, then we show that  $\text{Sam}_0$  and  $\text{Sam}_1$  are not SuP-admissible. Towards this goal, we construct a distinguisher  $D^*$  as follows. Distinguisher  $D^*$  receives as inputs  $(1^\lambda, \mathbf{s}, \mathbf{aux}, \overline{C}, |C|)$ . It then runs  $\text{Sim}^{\overline{C}}(1^\lambda, \mathbf{aux}, |C|)$  to receive a bit  $b'$  and outputs  $b'$ .

As the simulator only has black-box access to the circuit,  $\text{Sim}^{C_b}(1^\lambda, \mathbf{aux}_b, |C_b|)$  has exactly the same behaviour as  $\text{Sim}^{\text{Obf}(\text{PAD}(\mathbf{s}(\lambda), C_b))}(1^\lambda, \mathbf{aux}_b, |C_b|)$ , and thus, the advantage  $\text{SuP}[\mathbf{s}]_{\text{Obf}, \text{Sam}_0, \text{Sam}_1}^{D^*}(\lambda)$  is equal to  $\epsilon$ .  $\square$

**On SuPA for indistinguishability obfuscation.** Besides being able to validate SuPA for VBB obfuscators there is (at least) one curiosity pointing towards that SuPA holds. Consider the case that it does not hold and that indistinguishability obfuscators exist. Let us assume an obfuscator always outputs circuits which are at least twice the size of the input circuit. Then we have constructions (such as the UCE construction from this paper) which might be insecure when instantiated with the above obfuscator but which become secure when the obfuscator is run repeatedly, for example, if we instantiate the construction with  $\text{iO}(\text{iO}(C))$ . Such a “security amplification” (from an inverse polynomial advantage to negligible advantage for any efficient adversary), however, seems unreasonable for any good obfuscator and indeed seems rare for a security notion based on a distinguishing game.

One hope to prove SuPA for indistinguishability obfuscation is the characterization by Goldwasser and Rothblum [GR07] as best-possible obfuscation. The definition of best-possible obfuscation is learning-based and states that everything that can be learned from an  $\text{iO}$ -obfuscated circuit  $\text{iO}(C)$  can also be learned from any functionally equivalent circuit  $C'$  of the same size, i.e.,  $|C| = |C'|$ . Intuitively, using best-possible obfuscation might be one way to prove SuPA. However, it is not clear how to remove the size-restriction by Goldwasser and Rothblum. Of course, proving (or even understanding) SuPA is one of the main questions that are left open by our work.

## 8 Point Obfuscation from UCEs

For our construction of a  $\text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  secure function (Section 5) we used indistinguishability obfuscation and composable AIPOs (for statistically unpredictable distributions). In this section we show that the latter is indeed a necessary condition for the existence of injective  $\text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{q\text{-query}}]$  secure functions. In the original UCE paper, Bellare, Hoang and Keelveedhi [BHK13a] show how to build a variant of point obfuscation from UCEs secure against computationally unpredictable sources. Brzuska et al. [BFM14] and Bellare et al. [BHK13b] independently suggested UCEs relative to statistically unpredictable sources and showed that the original point obfuscation scheme can be lifted to this form of UCEs.

BHK show that the random oracle in the obfuscation scheme by Lynn, Prabhakaran, and Sahai [LPS04] can be instantiated with a UCE secure function. LPS consider multi-bit output point obfuscation for point functions  $p_{\alpha, \beta}$  defined as

$$p_{\alpha, \beta}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ 0 & \text{otherwise} \end{cases}$$

Note that these are different from the “simple” point functions that we considered for AIPO obfuscation which were simply defined as

$$p_\alpha(x) = \begin{cases} 1 & \text{if } x = \alpha \\ 0 & \text{otherwise} \end{cases}$$

Multi-bit output point obfuscation is closely related to composable point obfuscation [CD08]. For an overview of point obfuscation and their relation to indistinguishability obfuscation we refer to [BM14a].

Let us recall the LPS obfuscation scheme can be instantiated with a  $\text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{2\text{-query}}]$  secure function. To obfuscate a point  $(\alpha, \beta)$  one computes

```

hk ←s H.KGen( $1^\lambda$ )
 $\bar{\alpha}$  ← H.Eval(hk, 0|| $\alpha$ )
 $\bar{\beta}$  ← H.Eval(hk, 1|| $\alpha$ )  $\oplus$   $\beta$ 
return (hk,  $\bar{\alpha}$ ,  $\bar{\beta}$ )

```

BHK show that this indeed yields a secure (self-)composable point obfuscation scheme when employing a  $\text{UCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{2\text{-query}}]$  secure function, and show that it is composable when employing its multi-key variant  $\text{mUCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{2\text{-query}}]$  that is a function supporting multi-keys against sources that use each key at most once. Note that for this application we explicitly require UCEs with long outputs and correctness of the obfuscation requires that the UCE function is injective.

In the following we show that an injective  $\text{mUCE}[\mathcal{S}^{\text{sup}} \cap \mathcal{S}^{1\text{-query}}]$  in fact also suffices to construct a composable AIPO for statistically unpredictable auxiliary input—note that BHK consider a notion of point obfuscation without auxiliary information. We can actually give the stronger statement, that already  $\text{mUCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{1\text{-query}}]$ , that is, strong statistical unpredictability is sufficient to imply AIPO.

**Lemma 8.1** *The existence of secure composable AIPO $[\mathcal{D}^{\text{sup}}]$  (for statistically unpredictable distributions) is a necessary condition for the existence of secure  $\text{mUCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{1\text{-query}}]$  functions that are injective.*

We note that the above Lemma similarly holds for the computational case. That is,  $\text{mUCE}[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{1\text{-query}}]$  implies the existence of secure composable AIPO $[\mathcal{D}^{\text{cup}}]$  and  $\text{UCE}[\mathcal{S}^{\text{s-cup}} \cap \mathcal{S}^{1\text{-query}}]$  implies the existence of (not necessarily composable) AIPO $[\mathcal{D}^{\text{cup}}]$ . Since we know that indistinguishability obfuscation and composable AIPO $[\mathcal{D}^{\text{cup}}]$  are mutually exclusive [BM14a] this yields a natural separation between  $\text{UCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{1\text{-query}}]$  and its multi-key variant  $\text{UCE}[\mathcal{S}^{\text{s-sup}} \cap \mathcal{S}^{1\text{-query}}]$ .

*Proof (Lemma 8.1).* We will give a reduction from an adversary  $(\mathcal{B}_1, \mathcal{B}_2)$  against the AIPO scheme to an adversary against a  $\text{mUCE}[\mathcal{S}^{\text{s-sup}}]$  secure function. Instead of proving the multi-bit LPS scheme secure, we will use a simple plain AIPO scheme which obfuscates a point  $x$  simply as

```

hk ←s H.KGen( $1^\lambda$ )
 $\alpha$  ← H.Eval(hk,  $x$ )
 $p$  ← (hk,  $\alpha$ )
return  $p$ 

```

Suppose there exists an adversary  $(\mathcal{B}_1, \mathcal{B}_2)$  against the above obfuscation scheme. We will prove the claim via three game hops visualized in Figure 8. The first game  $\text{Game}_1$  is the original AIPO game with hidden bit  $b$  set to 0, that is, the point functions are generated for the points output by  $\mathcal{B}_1$ . From there we gradually move to  $\text{Game}_4$  which is the AIPO game with hidden bit  $b$  set to 1. In the following we first describe the games and then show that the steps reduce to  $\text{mUCE}[\mathcal{S}^{\text{s-sup}}]$  security:

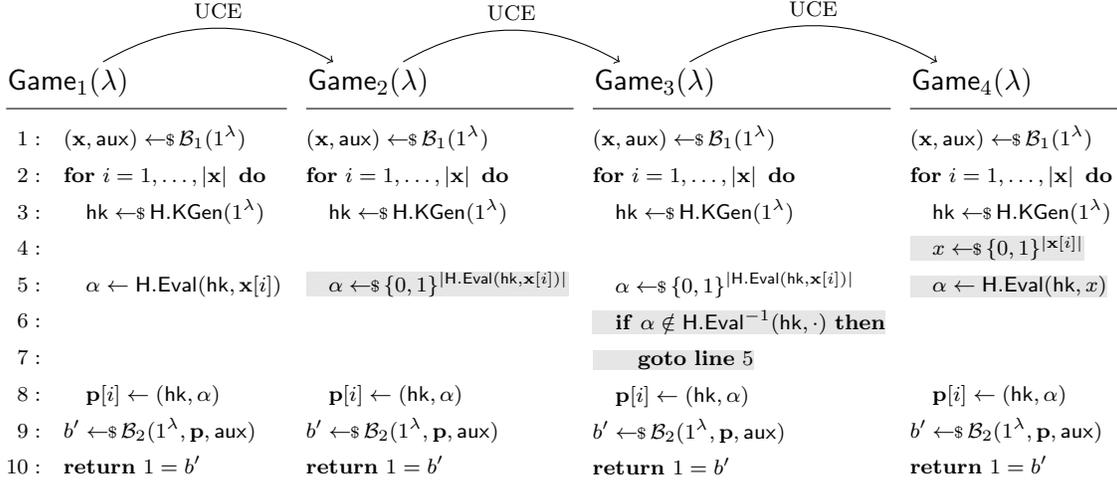


Figure 8: Game hops for proof of Lemma 8.1.

**Game<sub>1</sub>** The original AIPO game with hidden bit  $b$  set to 0, that is, the point functions are generated for the points output by  $\mathcal{B}_1$ .

**Game<sub>2</sub>** The game is as before but in line 5 value  $\alpha$  is generated as a uniformly random bit string of the same length  $|\text{H.Eval}(\text{hk}, \mathbf{x}[i])|$  (for the  $i$ -th value).

**Game<sub>3</sub>** The game is as before but it is ensured that  $\alpha$  has a (unique) preimage under  $\text{H.Eval}(\text{hk}, \cdot)$ .

**Game<sub>4</sub>** The original AIPO game with hidden bit  $b$  set to 1, that is, the point functions are generated for uniformly random points.

We can write the advantage of adversary  $\mathcal{B}_1, \mathcal{B}_2$  as

$$\begin{aligned}
\text{Adv}_{\mathcal{B}_1, \mathcal{B}_2, \text{H}}^{\text{PO}}(\lambda) &= \Pr \left[ \text{PO}_{\text{H}}^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \mid b = 0 \right] + \Pr \left[ \text{PO}_{\text{H}}^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \mid b = 1 \right] - 1 \\
&= \Pr \left[ \text{Game}_1^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \right] - \Pr \left[ \text{Game}_5^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \right] \\
&\leq \sum_{i=1}^3 \left| \Pr \left[ \text{Game}_i^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \right] - \Pr \left[ \text{Game}_{i+1}^{\mathcal{B}_1, \mathcal{B}_2}(\lambda) \right] \right|
\end{aligned}$$

In the following we show that the distance between any two games is negligible.

**Game<sub>1</sub> TO Game<sub>2</sub>.** We construct a UCE adversary  $(\text{S}, \text{D})$ . Without loss of generality we assume that  $\mathcal{B}_1$  outputs  $t$  many points (where  $t$  is some polynomial in  $\lambda$ ) and can hence consider a source that always works on  $t$  many keys. We consider  $(\text{S}, \text{D})$  as

$\text{S}^{\text{HASH}}(1^\lambda)$	$\text{D}(1^\lambda, \text{hk}, L)$
$(\mathbf{x}, \text{aux}) \leftarrow \mathcal{B}_1(1^\lambda)$	$(\alpha, \text{aux}) \leftarrow L$
<b>for</b> $i = 1, \dots, t$ <b>do</b>	<b>for</b> $i = 1, \dots, t$ <b>do</b>
$\alpha[i] \leftarrow \text{HASH}(\mathbf{x}[i], i)$	$\mathbf{p}[i] \leftarrow (\text{hk}[i], \alpha[i])$
$L \leftarrow (\alpha, \text{aux})$	$b' \leftarrow \mathcal{B}_2(\mathbf{p}, \text{aux})$
<b>return</b> $L$	<b>return</b> $b'$

In case HASH implements the actual hash function, then  $(\text{S}, \text{D})$  simulate **Game<sub>1</sub>** and in case HASH implements a random function they simulate **Game<sub>2</sub>**. Thus, we have that

$$|\Pr[\text{Game}_1(\lambda)] - \Pr[\text{Game}_2(\lambda)]| \leq \text{Adv}_{\text{H}, \text{S}, \text{D}}^{\text{UCE}}(\lambda)$$

Game<sub>2</sub> TO Game<sub>3</sub> TO Game<sub>4</sub>. Before we consider the distance between games Game<sub>2</sub> and Game<sub>3</sub> we will argue that games Game<sub>3</sub> and Game<sub>4</sub> are negligibly close. For this, note that the distributions on  $\alpha$  is identical in games Game<sub>3</sub> and Game<sub>4</sub> as for an injective function it is irrelevant if we sample from the preimage space or from the image space. Thus we have that

$$\Pr[\text{Game}_3(\lambda)] = \Pr[\text{Game}_2(\lambda)].$$

We can now bound the distance between games Game<sub>2</sub> and Game<sub>3</sub> again down to UCE. For this we consider (S, D), where D is as before, but S now asks random queries (as in Game<sub>4</sub>).

$S^{\text{HASH}}(1^\lambda)$	$D(1^\lambda, \mathbf{hk}, L)$
$(\mathbf{x}, \text{aux}) \leftarrow_{\$} \mathcal{B}_1(1^\lambda)$	$(\alpha, \text{aux}) \leftarrow L$
<b>for</b> $i = 1, \dots, t$ <b>do</b>	<b>for</b> $i = 1, \dots, t$ <b>do</b>
$x \leftarrow_{\$} \{0, 1\}^{ \mathbf{x}[i] }$	$\mathbf{p}[i] \leftarrow (\mathbf{hk}[i], \alpha[i])$
$\alpha[i] \leftarrow \text{HASH}(, i)$	$b' \leftarrow_{\$} \mathcal{B}_2(\mathbf{p}, \text{aux})$
$L \leftarrow (\alpha, \text{aux})$	<b>return</b> $b'$
<b>return</b> $L$	

Now, the view given by (S, D) when HASH is implemented as a random function is exactly as in Game<sub>2</sub> and in case it is implemented by the actual hash function HASH it is as in Game<sub>4</sub> (which is thus an identical view towards  $\mathcal{B}_2$  as given in Game<sub>3</sub>).

This concludes the proof. □

## Acknowledgments

We thank Victoria Fehr, Marc Fischlin, Peter Gaži, Felix Günther and Markulf Kohlweiss for helpful comments on the presentation. Arno Mittelbach was supported by CASED ([www.cased.de](http://www.cased.de)) and the German Research Foundation (DFG) SPP 1736.

## References

- [ABG<sup>+</sup>13] Prabhanjan Ananth, Dan Boneh, Sanjam Garg, Amit Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. *Cryptology ePrint Archive*, Report 2013/689, 2013. <http://eprint.iacr.org/2013/689>. (Cited on page 9.)
- [BC10] Nir Bitansky and Ran Canetti. On strong simulation and composable point obfuscation. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 520–537, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. (Cited on page 11.)
- [BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 52–73, San Diego, CA, USA, February 24–26, 2014. Springer, Berlin, Germany. (Cited on pages 9, 19, 20, 21, and 32.)
- [BFM14] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 188–205, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany. (Cited on pages 3, 6, 7, 13, 14, 16, and 24.)

- [BFM15] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Random-oracle uninstantiability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 428–455, Warsaw, Poland, March 23–25, 2015. Springer, Berlin, Germany. (Cited on pages 3 and 7.)
- [BGI<sup>+</sup>01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. (Cited on pages 8 and 9.)
- [BGI<sup>+</sup>12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, May 2012. (Cited on page 8.)
- [BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Berlin, Germany. (Cited on page 12.)
- [BH15] Mihir Bellare and Viet Tung Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 627–656, Sofia, Bulgaria, April 26–30, 2015. Springer, Berlin, Germany. (Cited on pages 3, 6, 7, and 17.)
- [BHK13a] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. Instantiating random oracles via UCEs. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 398–415, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany. (Cited on pages 3, 4, 12, 13, 14, 19, and 24.)
- [BHK13b] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. Instantiating random oracles via UCEs. Cryptology ePrint Archive, Report 2013/424, 2013. <http://eprint.iacr.org/2013/424>. (Cited on pages 3, 4, 6, 13, 14, 15, 17, and 24.)
- [BHK14] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. Cryptography from compression functions: The UCE bridge to the ROM. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 169–187, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany. (Cited on page 4.)
- [BKR13] Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. Message-locked encryption and secure deduplication. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 296–312, Athens, Greece, May 26–30, 2013. Springer, Berlin, Germany. (Cited on page 4.)
- [BM14a] Christina Brzuska and Arno Mittelbach. Indistinguishability obfuscation versus multi-bit point obfuscation with auxiliary input. In Palash Sarkar and Tetsu Iwata, editors, *Advances*

*in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 142–161, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Berlin, Germany. (Cited on pages 5, 11, and 25.)

- [BM14b] Christina Brzuska and Arno Mittelbach. Using indistinguishability obfuscation via UCEs. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 122–141, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Berlin, Germany. (Cited on pages 4, 5, 6, 7, 12, 13, 14, 16, 17, 18, and 33.)
- [BP12] Nir Bitansky and Omer Paneth. Point obfuscation and 3-round zero-knowledge. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 190–208, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Berlin, Germany. (Cited on page 10.)
- [BP13] Elette Boyle and Rafael Pass. Limits of extractability assumptions with distributional auxiliary input. Cryptology ePrint Archive, Report 2013/703, 2013. <http://eprint.iacr.org/2013/703>. (Cited on page 9.)
- [BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. (Cited on page 6.)
- [BRS03] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002: 9th Annual International Workshop on Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75, St. John’s, Newfoundland, Canada, August 15–16, 2003. Springer, Berlin, Germany. (Cited on page 6.)
- [BST14] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 102–121, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Berlin, Germany. (Cited on page 8.)
- [BST15] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. Contention in cryptoland: Obfuscation, leakage and UCE. Cryptology ePrint Archive, Report 2015/487, 2015. <http://eprint.iacr.org/>. (Cited on pages 5 and 7.)
- [BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300, Bangalore, India, December 1–5, 2013. Springer, Berlin, Germany. (Cited on pages 11 and 12.)
- [Can97] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO’97*, volume 1294 of *Lecture Notes in Computer Science*, pages 455–469, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. (Cited on pages 10 and 11.)

- [CD08] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating point functions with multibit output. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 489–508, Istanbul, Turkey, April 13–17, 2008. Springer, Berlin, Germany. (Cited on page 25.)
- [DAB<sup>+</sup>02] John R. Douceur, Atul Adya, William J. Bolosky, Dan Simon, and Marvin Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In *ICDCS*, pages 617–624, 2002. (Cited on page 4.)
- [DGG<sup>+</sup>15] Yevgeniy Dodis, Chaya Ganesh, Alexander Golovnev, Ari Juels, and Thomas Ristenpart. A formal treatment of backdoored pseudorandom generators. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 101–126, Sofia, Bulgaria, April 26–30, 2015. Springer, Berlin, Germany. (Cited on pages 3 and 6.)
- [GGH<sup>+</sup>13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press. (Cited on page 9.)
- [GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th Annual Symposium on Foundations of Computer Science*, pages 464–479, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press. (Cited on page 12.)
- [GR07] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 194–213, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany. (Cited on pages 3, 7, and 24.)
- [GR14] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. *Journal of Cryptology*, 27(3):480–505, July 2014. (Cited on page 7.)
- [KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13: 20th Conference on Computer and Communications Security*, pages 669–684, Berlin, Germany, November 4–8, 2013. ACM Press. (Cited on page 12.)
- [LPS04] Ben Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 20–39, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany. (Cited on pages 6 and 24.)
- [MH14a] Takahiro Matsuda and Goichiro Hanaoka. Chosen ciphertext security via point obfuscation. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 95–120, San Diego, CA, USA, February 24–26, 2014. Springer, Berlin, Germany. (Cited on page 11.)
- [MH14b] Takahiro Matsuda and Goichiro Hanaoka. Chosen ciphertext security via UCE. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 56–76,

Buenos Aires, Argentina, March 26–28, 2014. Springer, Berlin, Germany. (Cited on pages 3, 6, and 17.)

- [Mit14] Arno Mittelbach. Salvaging indistinguishability in a multi-stage setting. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 603–621, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany. (Cited on pages 4, 14, and 19.)
- [SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 475–484, New York, NY, USA, May 31 – June 3, 2014. ACM Press. (Cited on page 11.)

## A Gamehop Analysis for Theorem 5.1

In this Section we present a detailed analysis of the individual game hops within the proof of Theorem 5.1.

**Analysis of game hops.** In  $\text{Game}_5$  we have reached the target setting, i.e., the UCE-game with the hidden bit set to 0 while  $\text{Game}_1$  denotes the real UCE-game with the hidden bit set to 1. That is, the HASH oracle answers with randomly chosen values independent of the hash key. Further note, that  $C_4$  and  $C_1$  are identical, that is they are as in the construction. Thus, we can write the advantage of an adversary  $(S, D)$  in the UCE-security game as

$$\begin{aligned} \text{Adv}_{S,D,H}^{\text{uce}}(\lambda) &= \Pr \left[ \text{UCE}_H^{S,D}(\lambda) \mid b = 1 \right] + \Pr \left[ \text{UCE}_H^{S,D}(\lambda) \mid b = 0 \right] - 1 \\ &= \Pr \left[ \text{Game}_1^{S,D}(\lambda) \right] - \Pr \left[ \text{Game}_5^{S,D}(\lambda) \right] \\ &\leq \sum_{i=1}^4 \left| \Pr \left[ \text{Game}_i^{S,D}(\lambda) \right] - \Pr \left[ \text{Game}_{i+1}^{S,D}(\lambda) \right] \right| \end{aligned}$$

In the following we show that the individual games are negligibly close. In Figure 4 we present the reduction target for each of the game hops above the pseudocode for each game.

$\text{Game}_1(\lambda)$  TO  $\text{Game}_2(\lambda)$ . In order to reduce to the security of the indistinguishability obfuscator  $iO$ , we show that, by construction, the circuits  $C_1[k]$  and  $C_2[k^*, P_0, P_1]$  compute the same function. If  $p(x) = \perp$ , then  $C_2[k^*, P_0, P_1]$  returns  $G.\text{Eval}(k^*, x)$ . If  $p(x) = 1$  for some  $p \in P_d$  then  $d$  is returned. Note that in this case  $p$  was placed in  $P_d$  only if  $G.\text{Eval}(k, x) = d$ . Hence, on all inputs  $x$ ,  $C_2[k^*, P_0, P_1]$  returns  $G.\text{Eval}(k, x)$  and so does  $C_1[k]$ . Having established the functional equivalence between the two circuit we can bound the difference between games  $\text{Game}_1$  and  $\text{Game}_2$  by the distinguishing advantage against the indistinguishability obfuscator  $iO$ . We now formalize this intuition.

We consider a circuit sampler  $\text{Sam}$  which runs the steps of  $\text{Game}_2$  up to and including line 7. Sampler  $\text{Sam}$  then outputs (the functionally equivalent) circuits  $C_1[k]$  and  $C_2[k^*, P_0, P_1]$  and auxiliary information  $\text{aux} \leftarrow L$ . Obfuscation distinguisher  $\text{Dist}$  gets as input  $\text{aux}$  and an obfuscated circuit  $\bar{C}$  which is either an obfuscation of  $C_1[k]$  or of  $C_2[k^*, P_0, P_1]$ . It sets  $\text{hk} \leftarrow \bar{C}$ ,  $L \leftarrow \text{aux}$  and runs  $D(1^\lambda, \text{hk}, L)$ . It outputs whatever  $D$  outputs.

If  $C = C_1[k]$  then adversaries  $(\text{Sam}, \text{Dist})$  perfectly simulate game  $\text{Game}_1(\lambda)$  and if  $C = C_2[k^*, P_0, P_1]$  then the adversaries simulate  $\text{Game}_2(\lambda)$ . Thus, we can rewrite the difference between the two games' distributions

$$\left| \Pr[\text{Game}_1(\lambda)] - \Pr[\text{Game}_2(\lambda)] \right| \leq \text{Adv}_{iO, \text{Sam}, \text{Dist}}^{iO}(\lambda)$$

$\text{Game}_2(\lambda)$  TO  $\text{Game}_3(\lambda)$ . We reduce the difference between  $\text{Game}_2$  and  $\text{Game}_3$  to the security of the puncturable PRF  $G$ . We define an adversary  $(\mathcal{A}_1, \mathcal{A}_2)$  against the puncturable PRF as follows. Adversary  $\mathcal{A}_1$  runs source  $S(1^\lambda)$  on the security parameter answering its queries to  $\text{HASH}$  with its own oracle challenge. It records the queries of  $S$  in sets  $X_0^*$  and  $X_1^*$  (as in procedure  $\text{HASH}$ ) storing queries that were answered with a 0 in  $X_0^*$  and queries that were answered with 1 in  $X_1^*$ . When  $S$  stops it records leakage  $L$ . It then executes lines 4 to (but not including) line 7. It then stops and outputs  $\text{state} \leftarrow (L, P_0, P_1)$ . Adversary  $\mathcal{A}_2$  is then run on input  $\text{state}$  and the punctured key  $k^*$ . It parses  $(L, P_0, P_1) \leftarrow \text{state}$  and constructs circuit  $C_2[k^*, P_0, P_1]$ . It sets  $\text{hk} \leftarrow_{\$} \text{iO}(C_2[k^*, P_0, P_1])$  and runs distinguisher  $D$  on input  $(1^\lambda, \text{hk}, L)$ . It outputs whatever  $D$  outputs.

If the challenge oracle answers honestly using the puncturable PRF  $G$  then adversary  $(\mathcal{A}_1, \mathcal{A}_2)$  perfectly simulates  $\text{Game}_2$  and otherwise it perfectly simulates  $\text{Game}_3$ . Thus, we have that

$$\Pr[\text{Game}_2(\lambda)] - \Pr[\text{Game}_3(\lambda)] \leq \text{Adv}_{G, \mathcal{A}_1, \mathcal{A}_2}^{\text{pprf}}(\lambda)$$

which by the security of the puncturable PRF  $G$  is negligible.

$\text{Game}_3(\lambda)$  TO  $\text{Game}_4(\lambda)$ . As circuits  $C_2[k^*, P_0, P_1]$  and  $C_3[k, P_0, P_1]$  use key  $k$  (resp.  $k^*$ ) only on values  $x \notin X_0^* \cup X_1^*$  the two circuits are functionally equivalent. An analogous analysis as from  $\text{Game}_1$  to  $\text{Game}_2$  hence yields that

$$|\Pr[\text{Game}_3(\lambda)] - \Pr[\text{Game}_4(\lambda)]| \leq \text{Adv}_{\text{iO}, \text{Sam}, \text{Dist}}^{\text{io}}(\lambda)$$

$\text{Game}_4(\lambda)$  TO  $\text{Game}_5(\lambda)$ . By construction, the circuits  $C_3[k, P_0, P_1]$  and  $C_4[k]$  only differ on points  $x \in X_0^* \cup X_1^*$ . We will bound the difference between games  $\text{Game}_4$  and  $\text{Game}_5$  by the differing-inputs security of the indistinguishability obfuscator  $\text{iO}$ . For this, we build on a result by Boyle, Chung and Pass (given as Theorem 2.4) who show that any indistinguishability obfuscator is also a differing-inputs obfuscator for differing-inputs circuits which differ on at most polynomially many points [BCP14]. As explained above, our circuits can differ only on points  $x \in X_0^* \cup X_1^*$  with  $|X_0^* \cup X_1^*| \in \text{poly}$  and hence we can apply their theorem.

In order to argue with the security property of differing-inputs obfuscation, we need to present a differing-inputs circuit sampler  $\text{Sam}$  generating circuits  $(C_3[k, P_0, P_1], C_4[k])$ . For this we consider  $\text{Sam}$  that runs the same steps as game  $\text{Game}_4$  up-to line 7 and which sets  $\text{aux} \leftarrow L$  and outputs  $(C_3[k, P_0, P_1], C_4[k], \text{aux})$ .

**Claim 1** *If AIPO is a secure composable AIPO[ $\mathcal{D}^{\text{sup}}$ ] obfuscator (see Definition 2.6), then  $\text{Sam}$  is a differing-inputs circuit sampler which outputs circuits that differ on at most  $q$  many points.*

Before proving Claim 1, we show how to use it to prove that the difference between  $\text{Game}_4(\lambda)$  and  $\text{Game}_5(\lambda)$  is small. Theorem 2.4 by Boyle et al. [BCP14] says that, if a family is differing-inputs and only differs on at most polynomially many points, then their indistinguishability obfuscations are indistinguishable. Claim 1 establishes that the family  $\text{Sam}$  is a differing-inputs sampler, and we already observed that circuits  $C_3[k, P_0, P_1]$  and  $C_4[k]$  only differ on polynomially many points. Hence, Theorem 2.4 allows us to do an analysis similar to the one from the first game hop and we get that

$$|\Pr[\text{Game}_4(\lambda)] - \Pr[\text{Game}_5(\lambda)]| \leq \text{Adv}_{\text{iO}, \text{Sam}, \text{Dist}}^{\text{io}}(\lambda) + [\text{BCP14}] \leq \text{negl}(\lambda)$$

We now proceed to proving Claim 1. Assume there exists an adversary (i.e., an extractor)  $\text{Ext}$  against the differing-inputs sample  $\text{Sam}$ . Then, intuitively, if  $\text{Ext}$  succeeds to find some target value  $\tau$  this should

help in breaking the obfuscation scheme AIPO since there must be one obfuscation  $p \in P_0 \cup P_1$  such that  $p(\tau) = 1$ . Let us now make this intuition formal.

We construct adversary  $(\mathcal{B}_1, \mathcal{B}_2)$  where  $\mathcal{B}_1$  describes a statistically unpredictable distribution. On input the security parameter,  $\mathcal{B}_1$  runs the steps of `Game4` up-to line 4. It constructs a sequence of points  $X$  which first contains all points from  $X_0^*$  and then all the points from  $X_1^*$ . It sets  $j \leftarrow |X_0|$ , that is,  $j$  is set to the number of points that were answered with 0. Adversary  $\mathcal{B}_2$  chooses an index  $\ell \leftarrow_{\$} [|X|]$  at random, samples a uniformly random value  $r \leftarrow_{\$} \{0, 1\}^{\text{H.il}(\lambda)}$  and sets  $b := \langle r, X[\ell] \rangle$ . It then sets  $\text{aux} \leftarrow (\ell, j, r, b, L)$  and outputs  $(X, \text{aux})$ .

Adversary  $\mathcal{B}_2$  gets as input the security parameter, auxiliary input  $(\ell, j, r, b, L) \leftarrow \text{aux}$  and a list of obfuscations  $\mathbf{p}$  which either contains point obfuscations for points in  $X$  or for randomly generated points. Adversary  $\mathcal{B}_2$  then constructs sets  $P_0$  and  $P_1$  as

```

 $P_0, P_1 \leftarrow \{\}$ 
for  $i = 1, \dots, |\mathbf{p}|$  do
  if  $i \leq j$  then
     $P_0 \leftarrow P_0 \cup \{\mathbf{p}[i]\}$ 
  else  $P_1 \leftarrow P_1 \cup \{\mathbf{p}[i]\}$ 

```

Adversary  $\mathcal{B}_2$  then samples a random key  $\mathbf{k} \leftarrow_{\$} G.\text{KGen}(\lambda)$  and constructs circuits  $C_3[\mathbf{k}, P_0, P_1]$  and  $C_4[\mathbf{k}]$ .

Adversary  $\mathcal{B}_2$  then calls `Ext` on input  $(C_3[\mathbf{k}, P_0, P_1], C_4[\mathbf{k}], L)$  to receive a value  $\tau$ . If `Ext` outputs  $\tau = \perp$ , then  $\mathcal{B}_2$  flips a bit and returns the outcome of the bitflip. Else, if  $\tau$  is such that  $C_3[\mathbf{k}, P_0, P_1](\tau) \neq C_4[\mathbf{k}](\tau)$  but  $\mathbf{p}[\ell](\tau) \neq 1$  then  $\mathcal{B}_2$  also flips a bit and returns the outcome of the bitflip. Finally, if  $\mathbf{p}[\ell](\tau) = 1$  then  $\mathcal{B}_2$  outputs 1 if  $\langle r, \tau \rangle$  equals  $b$  and 0 otherwise.

If  $\mathbf{p}$  is an obfuscation of the points in  $X$  and `Ext` outputs  $\tau$  such that  $\mathbf{p}[\ell](\tau) = 1$  then  $\mathcal{B}_2$  will output 1 with probability 1. If, on the other hand,  $\mathbf{p}$  is a sequence of obfuscations of random points and `Ext` outputs  $\tau$  such that  $\mathbf{p}[\ell](\tau) = 1$  then  $\mathcal{B}$  will only output 1 with probability  $\frac{1}{2}$  (since  $\Pr[\langle u, r \rangle = b] = \frac{1}{2}$  for a random point  $u$ ). A formal analysis is analogous to the analysis conducted in [BM14b] with an additional loss of factor  $\frac{1}{q}$  for guessing the right index  $\ell$ .

To finish the proof of Claim 1, we need to argue that  $\mathcal{B}_1$  implements a statistically unpredictable distribution. By assumption, the source  $\mathbf{S}$  is statistically unpredictable (i.e.,  $\mathbf{S} \in \mathcal{S}^{\text{sup}}$ ) and hence leakage  $L$  hides  $X$ . Thus, to see that  $\mathcal{B}_1$  defines an unpredictable distribution, we need to argue that  $X$  remains unpredictable if additionally given  $(\ell, j, r, b)$ . But a single bit  $b$  and two indexes  $\ell, j \in [q]$  can be guessed with probability  $\frac{1}{2q^2}$  and  $r$  is a uniformly random value. Hence,  $(\mathcal{B}_1, \mathcal{B}_2)$  breaks the security of the AIPO obfuscation, which concludes the proof of Claim 1 and the proof of Theorem 5.1.