Black-Box Uselessness: Composing Separations in Cryptography

Geoffroy Couteau, Pooya Farshim, Mohammad Mahmoody

Black-Box Uselessness: Composing Separations in Cryptography

Geoffroy Couteau* Pooya Farshim† Mohammad Mahmoody‡

Abstract

Black-box separations have been successfully used to identify the limits of a powerful set of tools in cryptography, namely those of black-box reductions. They allow proving that a large set of techniques are not capable of basing one primitive $\mathcal{P}$ on another $\mathcal{Q}$. Such separations, however, do not say anything about the power of the *combination* of primitives $\mathcal{Q}_1, \mathcal{Q}_2$ for constructing $\mathcal{P}$, even if $\mathcal{P}$ cannot be based on $\mathcal{Q}_1$ or $\mathcal{Q}_2$ alone.

By introducing and formalizing the notion of *black-box uselessness*, we develop a framework that allows us to make such conclusions. At an informal level, we call primitive $\mathcal{Q}$ black-box useless (BBU) for primitive $\mathcal{P}$ if $\mathcal{Q}$ cannot help constructing $\mathcal{P}$ in a black-box way, even in the presence of another primitive $\mathcal{Z}$. This is formalized by saying that $\mathcal{Q}$ is BBU for $\mathcal{P}$ if for *any* auxiliary primitive $\mathcal{Z}$, whenever there exists a black-box construction of $\mathcal{P}$ from $(\mathcal{Q}, \mathcal{Z})$, then there must already also exist a black-box construction of $\mathcal{P}$ from $\mathcal{Z}$ alone. We also formalize various other notions of black-box uselessness, and consider in particular the setting of *efficient* black-box constructions when the number of queries to $\mathcal{Q}$ is below a threshold.

Impagliazzo and Rudich (STOC'89) initiated the study of black-box separations by separating key agreement from one-way functions. We prove a number of initial results in this direction, which indicate that one-way functions are perhaps also *black-box useless* for key agreement. In particular, we show that OWFs are black-box useless in any construction of key agreement in either of the following settings: (1) the key agreement has perfect correctness and one of the parties calls the OWF a constant number of times; (2) the key agreement consists of a single round of interaction (as in Merkle-type protocols). We conjecture that OWFs are indeed black-box useless for general key agreement protocols.

We also show that certain techniques for proving black-box separations can be lifted to the uselessness regime. In particular, we show that known lower bounds for assumptions behind black-box constructions of indistinguishability obfuscation (IO) can be extended to derive black-box uselessness of a variety of primitives for obtaining (approximately correct) IO. These results follow the so-called "compiling out" technique, which we prove to imply black-box uselessness.

Eventually, we study the complementary landscape of black-box uselessness, namely black-box *helpfulness*. Formally, we call primitive $\mathcal{Q}$ black-box helpful (BBH) for $\mathcal{P}$, if there exists an auxiliary primitive $\mathcal{Z}$ such that there exists a black-box construction of $\mathcal{P}$ from $(\mathcal{Q}, \mathcal{Z})$, but there exists no black-box construction of $\mathcal{P}$ from $\mathcal{Z}$ alone. We put forth the conjecture that one-way functions are black-box helpful for building collision-resistant hash functions. We define two natural relaxations of this conjecture, and prove that both of these conjectures are implied by a natural conjecture regarding random permutations equipped with a collision finder oracle, as defined by Simon (Eurocrypt'98). This conjecture may also be of interest in other contexts, such as hardness amplification.

*CNRS and IRIF, Paris-Diderot University, France

†Department of Computer Science, University of York, UK

‡Department of Computer Science, University of Virginia, USA. Supported by NSF grants CNS-1936799 and CCF-1910681.

Contents

1 Introduction

1.1 Background

Black-box reductions are a central tool in cryptography. They have helped shape a rich landscape of relations between different cryptographic notions, allowing us to develop a better understanding of their powers and limitations.

Roughly speaking, in a (fully) black-box reduction both the design and analysis of a protocol treat the underlying primitives and adversaries in a black-box way, obliviously of their internals. More precisely, we say there is a fully black-box construction of a primitive $\mathcal{P}$ from a primitive $\mathcal{Q}$ if there is an efficient construction P^{Q} that for every implementation Q of primitive $\mathcal{Q}$ implements primitive $\mathcal{P}$, and further, there is an efficient security reduction $\mathsf{S}^{\mathsf{Q},A}$ which for every adversary A^{P} that breaks P, breaks Q. This notion originates in the seminal work of Impagliazzo and Rudich [IR89], and it was later refined by Reingold, Trevisan, and Vadhan [RTV04] as well as Baecher, Brzuska, and Fischlin [BBF13] who proposed a taxonomy of notions of reducibility between cryptographic primitives.

Impagliazzo and Rudich showed how to attack any key-agreement (KA) protocol in the random-oracle (RO) model with only a polynomial number queries to the random oracle.[1] This result is sufficient to rule out fully black-box reductions, since, roughly speaking, the construction is assumed to work for any OWF oracle f, and in particular for a RO and moreover, the security reduction works for any adversary A, and in particular for one the does not necessarily run in polynomial time but makes a polynomial number of queries to f.

Following this work, a successful and long line of research studying separations between different cryptographic primitives followed (e.g., see [Sim98, GGKT05, HHRS07, BM09, BKSY11, AS15, GMM17a] and references therein). In this work we will revisit these works and ask if and to what extent their results hold in the presence of other primitives.

1.2 Black-Box Uselessness

Cryptographic constructions typically rely on multiple, incomparable building blocks. This raises the question if, and to what extent, a black-box separation result proves that some primitive is *useless* for building another even with other primitives. Take, for example, the case of key-agreement (KA) and one-way functions (OWFs). Although OWFs on their own are insufficient to build KA, this leaves the possibility that together with some other primitive $\mathcal{Z}$ they do imply KA in a black-box way, even if $\mathcal{Z}$ also does not black-box imply KA.[2] More generally, suppose we have separated primitive $\mathcal{P}$ from primitives $\mathcal{Q}_1$ and $\mathcal{Q}_2$ with respect to black-box reductions. That is, neither $\mathcal{Q}_1$ nor $\mathcal{Q}_2$ can be used to build $\mathcal{P}$. Does it then necessarily follow that $\mathcal{P}$ is also black-box separated from $\mathcal{Q}_1$ and $\mathcal{Q}_2$ put together? More generally, one may ask:

Which black-box impossibility results compose?

In general, not all black-box impossibility results compose. Indeed, consider a primitive $\mathcal{P}$ that is set to be the "union" of primitives $\mathcal{Q}_1$ and $\mathcal{Q}_2$, where $\mathcal{Q}_1$ and $\mathcal{Q}_2$ are mutually separated (i.e., neither can be based on the other). Then although $\mathcal{P}$ cannot be based on $\mathcal{Q}_1$ or $\mathcal{Q}_2$, it can be

[1] More precisely, their attack made $\mathcal{O}((qm)^3)$ RO queries, where q is the number of queries of the protocol to the RO and m is the number of messages exchanged.

[2] Note that constructions of PKE from OWFs plus indistinguishability obfuscation are *non*-black-box [SW14].

trivially based on their union.[3] This situation is somewhat unsatisfying: due to a joint effort of the cryptographic community in the past three decades, we have at our disposal a large number of black-box separations between pairs of primitives, yet we know essentially nothing about whether this situation changes if we are willing to use several primitives in a black-box construction – and of course, black-box separating subsets of primitives from a target primitive would be tedious. This leaves out the possibility that such separations could be obtained more systematically.

In this work, we seek to devise a more efficient strategy, by identifying conditions under which a primitive $\mathcal{P}$ is black-box separated from primitive $\mathcal{Q}$ *in a composable way*. That is, primitive $\mathcal{Q}$ in conjunction with *any* other primitive $\mathcal{Z}$ cannot be used to build $\mathcal{P}$, unless of course $\mathcal{P}$ can be built using $\mathcal{Z}$ alone. Our starting point is the following notion of *black-box uselessness*:[4]

Definition 1.1 (Black-box uselessness, informal)**.** A primitive $\mathcal{Q}$ is black-box useless (BBU) for primitive $\mathcal{P}$ if for any auxiliary primitive $\mathcal{Z}$, whenever there is a black-box construction of $\mathcal{P}$ from $(\mathcal{Q}, \mathcal{Z})$ there also exists a black-box construction of $\mathcal{P}$ from the auxiliary primitive $\mathcal{Z}$ alone.

A *composition theorem* for black-box uselessness immediately follows: if $\mathcal{Q}_1$ is BBU for $\mathcal{P}$ and $\mathcal{Q}_2$ is BBU for $\mathcal{P}$ then $\mathcal{Q}_1$ and $\mathcal{Q}_2$ put together are BBU for $\mathcal{P}$. Indeed, for any $\mathcal{Z}$, if $(\mathcal{Q}_1, \mathcal{Q}_2, \mathcal{Z}) = (\mathcal{Q}_1, (\mathcal{Q}_2, \mathcal{Z}))$ black-box implies $\mathcal{P}$, then $(\mathcal{Q}_2, \mathcal{Z})$ must black-box imply $\mathcal{P}$ (by the black-box uselessness of $\mathcal{Q}_1$). This in turn implies, by the black-box uselessness of $\mathcal{Q}_2$, that $\mathcal{Z}$ alone black-box imply $\mathcal{P}$.

Remark 1.2. A black-box uselessness result of $\mathcal{Q}$ for $\mathcal{P}$ implies in particular that $\mathcal{Q}$ does not black-box imply $\mathcal{P}$, as long as $\mathcal{P}$ is not a (trivial) primitive that exists unconditionally. Indeed, by the black-box uselessness of $\mathcal{Q}$, if $\mathcal{Q}$ black-box implies $\mathcal{P}$, then taking $\mathcal{Z}$ as the "empty primitive" we get that $\mathcal{P}$ exists unconditionally in the plain model. For instance, although one-way functions are black-box useless for the one-time pad (since the latter exists unconditionally in the presence of any auxiliary oracle), one-way functions black-box imply the one-time pad (for essentially the same reason).

1.3 One-Way Functions and Key Agreement

Perhaps one of the most fundamental questions regarding black-box uselessness is to understand whether or not one-way functions are black-box useless for key agreement. We start with an observation on a natural approach for building key-agreement from one-way functions together with other primitives.

Remark 1.3. When looking for candidate primitives that when combined with one-way functions imply key agreement, the notion of indistinguishability obfuscation (iO) [BGI+01, SW14] might be the first that comes to mind. As mentioned, the construction of PKE from IO and OWFs [SW14]

[3] This formal counterexample can be converted into more "natural" ones. Take identity-based encryption (IBE) with *compact* user keys, whose lengths are independent of the length of user identities. One can use a standard IBE and a collision-resistance hash function (CRHF) to build a compact IBE (by simply hashing the identities). Yet, compact IBE cannot be based on CRHFs in a black-box way (since PKE cannot be based on ROs). Furthermore, compact IBE cannot be based on standard IBE, since compact IBE implies CRHF (the keys must be collision-free for otherwise the compact IBE can be broken by finding a collision among keys) and standard IBE does not imply CRHFs [MM16].

[4] The terminology "a primitive X is useless for black-box constructions of a primitive Y" has been used sometimes in the literature, e.g. in [MW20], to mean that Y does not black-box reduce to X. Our notion of black-box uselessness should not be confused with this terminology, which only refers to conventional black-box separations.

is not black-box. Furthermore, we observe that this is unavoidable: Impagliazzo and Rudich [IR89] actually show, along the way, that there is no black-box construction of key agreement from one-way functions *and iO*. This is because the result of [IR89] shows that relative to a random oracle *and a* PSPACE *oracle* there is no key agreement, yet one-way functions exist. However, relative to these oracles, there is also a perfectly secure (deterministic) IO scheme: on input a circuit of a given size, use the PSPACE oracle to find the lexicographically first functionally equivalent circuit of the same size.

In this work, we provide a partial answer to the question of whether or not one-way functions are black-box useless for key agreement. We show that one-way functions are black-box useless in any construction of key agreement satisfying certain restrictions. To describe our result, it is instructive to start with the separation of perfectly correct KA from OWFs by Brakerski, Katz, Segev, and Yerukhimovich [BKSY11].

BKSY11 in a nutshell. Given a perfectly correct, two-party KA protocol in the RO model, where Alice and Bob place at most q queries to the RO, consider an attacker Eve that proceeds as follows. Given a transcript T of the protocol, Eve samples coins r'_A and a random oracle RO' for Alice that are consistent with T. Eve then runs Alice on r'_A and RO' and records all oracle queries and the resulting key. It then places all these queries to the real RO to obtain an assignment (a list of query-answer pairs) L. Eve repeats this process, appending to L and storing keys, while ensuring that the sampling of RO' is consistent with L computed so far. Now, if in a sampled run of Alice, there are no queries of Alice in common with those of Bob outside L, by perfect correctness, the correct key will be computed. Otherwise, a query of Bob will be discovered. Since Bob has at most q queries, if Eve executes this procedure $2q+1$ times, in at least q of the runs no intersection queries will be discovered, and in these runs the correct key will be computed. Thus taking a majority over the set of keys computed, Eve obtains the key with probability one.

Upgrading to BBU. In order to convert this proof into a black-box uselessness result, we make a case distinction based on whether or not one can "lift" the sampling procedure to a Z-relativized world for any oracle Z. Given a construction $\mathsf{KA}^{\mathsf{F},\mathsf{Z}}$ of KA from a OWF F and Z, if the sampling procedure can be successfully carried out in the presence of Z, then we could efficiently break any perfectly correct KA protocol in the presence of Z only (since the attacker computes the correct key with probability one).

Now suppose at some iteration Eve is no longer able to find coins and a random oracle consistent with the transcript while making polynomially many queries to Z. We claim that in this case we can construct a *weak* one-way function. Indeed, consider the function that runs the above attack procedure up to but excluding the stage where Eve fails. Thus, this function starts by running the protocol, then uses the sampler for the first iteration (if sampling was not already impossible at this stage) to obtain an assignment, and continues with another round of sampling, until it arrives at the stage where sampling is no longer possible. The transcript of the protocol at this stage together with the list of assignments so far constitutes the challenge for which there is no inverter. From a weak one-way function, a full-fledged one-way function follows by the result of [Yao82], as this construction is black-box and hence relativizes with respect to Z.

We may now use the one-way function obtained from Z in place of the original one-way function F to remove reliance on the F all together. Thus, we obtain a KA protocol which relies only on Z,

as required. We emphasize that this proof only shows the uselessness of OWFs for (perfect) KA and not that of random oracles, since we only obtain a one-way function using Z.

Note that since we recursively rely on the existence of an inverter, the query complexity (to Z) of the samplers can potentially blow up. Indeed, suppose for some Z, any sampler needs to place $\mathcal{O}(n^2)$ queries to Z to invert a function that places n queries to Z. After the first iteration, we arrive at the construction of a function that places $n + \mathcal{O}(n^2) = \mathcal{O}(n^2)$ queries to Z. Thus, it may well be that a successful inverter at this step needs to place $\mathcal{O}(n^4)$ queries to Z. This in particular implies that the recursive argument above can be applied for only a *constant* number of steps. Since we only need to apply the recursive sampling for *either* Alice *or* Bob, we obtain a BBU result as long as either Alice or Bob makes a constant number of queries to the RO (but polynomially many calls to Z). We formalize this proof in Section 4, where we point out the other subtleties that arise.

Currently, we are not able to extend the proof to arbitrary protocols where both Alice and Bob make a polynomial number of RO queries. Despite this, we can show that OWFs are black-box useless for building *constant-round, imperfect* key agreements when both parties make a constant number of queries to the OWF (and an arbitrary number of queries to the auxiliary oracle), and that OWFs are black-box useless for *one-round* key agreement (without restriction on the queries made by the parties). We defer the details to Section 4.2.

1.4 The Compilation Technique

A number of black-box separation results rely on what we here refer to as the *efficient compiling-out* (or simply compilation) paradigm [GGKT05, CKP15, GMM17a, GMM17c, BLP17, AKW18, GHMM18]. At a high-level, here given a construction $\mathsf{G}_1^{\mathsf{P}}$ one compiles out the primitive P via an (oracle-free) simulator Sim which simulates P for the construction in a consistent way and without affecting security. The result is a new construction G_2 that no longer uses P. This proof technique is closely related to black-box uselessness, and as we will see can often be turned into a black-box uselessness result with minor modifications. This in turn highlights the advantage of separation results that are achieved using this technique.

In order to show how this can be done, we briefly discuss this in the context of the work of Canetti, Kalai, Paneth [CKP15], who showed that obfuscation cannot be based on random oracles.[5]

CKP and black-box uselessness of random oracles for indistinguishability obfuscation. Consider an obfuscator $\mathsf{Obf}^{\mathsf{RO}}$ that makes use of a random oracle RO. On input a circuit without random oracle gates[6] the obfuscation algorithm outputs a circuit C^{RO}, which may also call the random oracle RO. CKP compile out the random oracle RO from any such construction as follows. First they convert $\mathsf{Obf}^{\mathsf{RO}}$ to an obfuscator in the plain model by simulating the RO for the obfuscator via lazy sampling. Next to ensure that the oracle expected by an obfuscated circuit C^{RO} is consistent with the oracle simulated for obfuscation, CKP execute C on multiple randomly chosen inputs at the obfuscation phase while simulating the random oracle consistently, record all queries made and corresponding simulated answers, and return them together with the obfuscated oracle circuit. Leaking this list cannot hurt security as an adversary in the RO model can also compute such a list given an obfuscated circuit. On the other hand, having this list allows the evaluation of C^{RO}

[5] The work of [CKP15] dealt with *virtual black-box* obfuscation, but it turns out [BBF16, MMN+16a, MMN+16b] that their proof could also be applied to the case of indistinguishability obfuscation.

[6] This restriction only makes the results of CKP stronger.

to be consistent with the obfuscation phase with high probability on a *random* input. (This is why the obtained primitive is only an *approximate-correct* IO – see Definition 2.8.)

As can be seen, this proof Z-relativizes in the sense that the simulation of the random oracle RO and the execution of the obfuscated circuit can be both done in the presence of any other oracle Z. By compiling out the random oracle RO in the presence of Z, we obtain a construction that relies on Z. That is, we obtain that random oracles are black-box uselessness for obfuscation.

A number of other impossibility results follow the compilation paradigm and here we observe that they can be lifted to the black-box uselessness regime. In particular, we consider the results from [GGKT05, CKP15, GMM17a, GMM17c] and show how they can be lifted to black-box uselessness. Indeed, as long as compilation is performed for a constant "number of rounds" and each step can be done efficiently we obtain black-box uselessness.[7] As a result we get that approximate IO cannot be obtained from a black-box *combination* of random oracles, predicate encryption, fully homomorphic encryption and witness encryption.[8]

Remark 1.4. We note that some previous works (e.g., [CKP15]) have described the result of Impagliazzo and Rudich as "compiling out" the random oracle from any key-agreement protocol. This process, however, differs from the compiling-out technique that we study in this paper in two aspects. First, compilation is inefficient and uses the sampling algorithm. Second, the process is carried out adaptively for multiple rounds. The inefficiency of the sampler translates to obtaining BBU for one-way functions only; adaptivity restricts our final result to protocols where one party makes at most a constant number of RO queries. On a similar note, the recent work of Maji and Wang [MW20] uses the term "black-box useless" as an alternative to proving (traditional) black-box separations. So, despite similarity in the terms, our notion of uselessness is quite different.

1.5 The Case of Collision Resistance

A classic result of Simon [Sim98] separates collision-resistance hash functions (CRHFs) from one-way functions (and indeed one-way permutations). This is done by giving a pair of oracles $(\pi, \mathsf{Coll}^{\pi})$ relative to which one-way permutations (and hence one-way functions) exist, but CRHFs don't. Here π implements a random permutation, and Coll^{π} is an oracle that takes as input a circuit with π-gates and returns a random collision for it (by first computing the circuit on a random point and then picking a random preimage).

Are one-way functions/permutations also black-box useless for collision resistance? One way to answer this question affirmatively would be to extend Simon's result along the lines of what was done for the separation results above. However, in this case we have an oracle separation result, and it is not clear how to "relativize" the proof. Indeed, we conjecture the opposite: OWFs are black-box *helpful* for building CRHFs. To this end, we would need to show that for *any* one-way function F there is a primitive Z, which although on its own is insufficient for building CRHFs, together with F can be used to build CRHFs. We present two possible approaches for proving this conjecture.

One approach follows the recent result of Holmgren and Lombardi [HL18], who showed how to obtain CRHFs from exponentially secure *one-way product functions* (OWPFs) in a black-box way. Roughly speaking, a OWP is a tuple of one-way functions $(\mathsf{F}_1, \ldots, \mathsf{F}_k)$ where any polynomial-time

[7] Lemma 3.25 in [GMM17b] shows a similar phenomenon in a context where we completely compile out a sequence of idealized oracles. Here we deal with a setting that an auxiliary oracle remains at the end.

[8] We note that this does not apply in the so-called monolithic model; see discussion at the end of Section 5.

adversary can invert $(\mathsf{F}_1(x_1), \ldots, \mathsf{F}_k(x_k))$ for random x_i with probability at most $\mathrm{negl}(n)/2^n$ (where n is the security parameter). A good candidate for primitive Z is thus a random permutation π together with Simon's oracle Coll^π for it. To get a positive helpfulness result we need to show that for any one-way function F the pair of functions $(\mathsf{F}, (\pi, \mathsf{Coll}^\pi))$ is product one-way. We intuitively expect this result to hold since π is fully independent of F and essentially all an adversary can do is to invert the F and (π, Coll^π) independently. Formalizing this observation requires handling additional technicalities; we refer the reader to Section 6 for details. We did not manage to prove this conjecture, and leave it as an interesting open problem which might be of independent interest.[9]

A second approach follows the work of Bauer, Farshim, Mazaheri [BFM18], who defined a new idealized model of computation, known as the backdoored random oracle (BRO) model, whereby an adversary can obtain arbitrary leakage of the function table of the RO. Under a communication complexity conjecture related to the set-intersection problem, BFM show that two independent BROs can be used to build a CRHF by simply xoring their outputs. The leakage oracle defined by BFM is sufficiently powerful to allow implementing Simon's collision-finding oracle. As a result, although a single BRO as an idealized primitive is black-box separated from CRHFs, conjecturally it is not black-box useless for building CRHFs.

Open problems. The central open problem left by our work is that of black-box uselessness of OWFs for arbitrary key agreement protocols. Given our BBU results for special classes of KA protocols, this conjecture may well be within reach. On the other hand, a straightforward generalization seems to require a refined sampler technique with low *adaptivity*. At the moment, however, it seems challenging to reduce the adaptivities of known samplers [IR89, BM09, BKSY11, MMV11]. Besides OWFs, whether or not CRHFs, or for that matter random oracles, are black-box useless for key agreement remains open. More generally, black-box separation results can be revisited from the point of view of uselessness. In particular, it would be interesting to consider extensions of the recent *monolithic* models to the BBU setting, as these capture certain well-known non-black-box techniques in cryptography.

1.6 Organization

We start by defining notions of reducibility and a number of cryptographic primitives in Section 2. Then in Section 3 we formally define various notions of black-box uselessness. In Section 4 we study the helpfulness of one-way functions for building key agreement. In Section 5 we show that a number of separations utilizing the complication technique lift to black-box uselessness results. We conclude the paper in Section 6 with a number of conjectures on the helpfulness of one-way functions for building collision-resistance hash functions.

2 Preliminaries

Notation. PPT stands for probabilistic polynomial time. An oracle-aided PPT machine/circuit A^{O} is a PPT machine/circuit with oracle access/gates, such that for any oracle O machine/circuit A^{O} runs in probabilistic polynomial time, where we count each call to the oracle as one step. For any $n \in \mathbb{N}$, we let $[n]$ denote the set $\{1, \ldots, n\}$. Given an interactive protocol between two parties

[9] It might be helpful to consider weaker versions of this problem. For example, given an ε-secure one-way permutation and a random oracle, can an attacker invert both simultaneously with probability better than $\mathrm{negl}(n)/2^n$?

Alice and Bob with access to an oracle O, we let $\langle \mathsf{Alice}^{\mathsf{O}}, \mathsf{Bob}^{\mathsf{O}} \rangle$ denote the distribution of triples (T, y_A, y_B) over the randomness of the parties and the oracle, where T denotes the transcript of the protocol, and y_A (resp., y_B) denotes the output of Alice (resp., Bob). We also use the same notation when O comes from a *distribution* over the oracles, in which case the probability distribution of the outputs are also over the randomness of the oracle.

2.1 Black-Box Reductions

The definitions and notions in this section mostly follow those in [RTV04]. Throughout, we use calligraphic letters such as $\mathcal{P}$ or $\mathcal{KA}$ for a cryptographic primitive, sans-serif letters (for example P or KA) for specific implementations, S for the security reduction/proof and P for a "generic" implementation. We denote an auxiliary oracle by Z and an adversary by A.

Definition 2.1 (Cryptographic primitive)**.** A cryptographic primitive $\mathcal{P}$ is a pair $(F_{\mathcal{P}}, R_{\mathcal{P}})$, where $F_{\mathcal{P}}$ is the set of functions *implementing* $\mathcal{P}$ and $R_{\mathcal{P}}$ is a relation. For each $\mathsf{P} \in F_{\mathcal{P}}$, the relation $(\mathsf{P}, A) \in R_{\mathcal{P}}$ means that the adversary A *breaks* the implementation P (according to $\mathcal{P}$). It is required that at least one function $\mathsf{P} \in \mathcal{P}$ is computable by a PPT algorithm.

Definition 2.2 (Fully black-box reduction)**.** A *fully black-box* reduction of a primitive $\mathcal{P}$ to another primitive $\mathcal{Q}$ is a pair (P, S) of oracle-aided PPT machine such that for any implementation $\mathsf{Q} \in \mathcal{Q}$ the following two conditions hold.

- **Implementation reduction**: P^{Q} implements $\mathcal{P}$, that is, $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$.
- **Security reduction**: For any function (adversary) A that $\mathcal{P}$-breaks $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$, i.e., $(\mathsf{P}^{\mathsf{Q}}, A) \in R_{\mathcal{P}}$), it holds that $\mathsf{S}^{\mathsf{Q},A}$ $\mathcal{Q}$-breaks Q, i.e., $(\mathsf{Q}, \mathsf{S}^{\mathsf{Q},A}) \in R_{\mathcal{Q}}$.

When clear from context, we will refer to fully black-box reductions simply as black-box reductions, to the implementation reduction as the *construction*, and to the security reduction as the *security proof*.

Definition 2.3 (Semi and weakly black-box reductions)**.** We say there is a *semi-black-box* reduction of a primitive $\mathcal{P}$ to primitive $\mathcal{Q}$ if there is an oracle-aided PPT P such that for any implementation $\mathsf{Q} \in \mathcal{Q}$,

- **Implementation reduction**: $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$.
- **Security reduction**: If there exists an oracle-aided PPT A such that A^{Q} $\mathcal{P}$-breaks P^{Q}, then there exists an oracle-aided PPT S such that S^{Q} $\mathcal{Q}$-breaks Q.

If the order of the quantifiers for the implementation reduction is switched in the sense that for any implementation $\mathsf{Q} \in \mathcal{Q}$ there is an oracle-aided PPT P such that the above two conditions hold, then we say there is a $\forall\exists$-*semi-black-box* reduction of $\mathcal{P}$ to $\mathcal{Q}$.

Weakly black-box reductions and its $\forall\exists$ variant thereof are defined analogously with the following difference. In weakly-black-box reductions (in comparison with semi-black-box ones) the adversary A cannot call the oracle Q and needs to be a regular PPT. Namely, for any PPT A such that A $\mathcal{P}$-breaks P^{Q}, then there exists an oracle-aided PPT S such that S^{Q} $\mathcal{Q}$-breaks Q.

Definition 2.4 (Existence relative to an oracle)**.** A primitive $\mathcal{P}$ is said to *exist* relative to an oracle O whenever (1) there is an oracle-aided PPT P such that $\mathsf{P}^{\mathsf{O}} \in F_{\mathcal{P}}$; and (2) no oracle-aided PPT machine A^{O} can $\mathcal{P}$-break P^{O}.

Non-uniform variants of security reductions were formalized in [CLMP13]. The following definition extends this to non-uniform implementation reductions.

Definition 2.5. A fully black-box reduction (P, S) of a primitive $\mathcal{P}$ from primitive $\mathcal{Q}$ is said to have a *non-uniform implementation* if P additionally takes as input a polynomial-sized non-uniform advice that can also depend on its oracle Q. The reduction is said to have a *non-uniform security reduction* if S additionally takes as input a polynomial-sized non-uniform advice that can also depend on its oracles Q and A.

2.2 Specific Cryptographic Primitives

Definition 2.6 (One-way functions)**.** A *one-way function* F relative to an oracle O is an oracle-aided PPT machine such that for any PPT adversary A (modeled as an oracle-aided PPT machine) there is a negligible function $\mathrm{negl}(\lambda)$ such that

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n}[\mathsf{F}^{\mathsf{O}}(A^{\mathsf{O}}(1^\lambda, \mathsf{F}^{\mathsf{O}}(x))) = \mathsf{F}^{\mathsf{O}}(x)] = \mathrm{negl}(\lambda) \ .$$

If the above is only required to hold for infinitely many values of $\lambda \in \mathbb{N}$, then F is an *infinitely-often* one-way function (io-$\mathcal{F}$).

Definition 2.7 (Key agreement)**.** An oracle-aided ε-key agreement with respect to an oracle O is an interactive protocol between two oracle-aided PPT machines Alice and Bob that satisfies the following ε-*correctness* and *security* properties.

- (ε-correctness) For any $\lambda \in \mathbb{N}$,

$$\Pr[(T, K_A, K_B) \xleftarrow{\$} \langle \mathsf{Alice}^{\mathsf{O}}(1^\lambda), \mathsf{Bob}^{\mathsf{O}}(1^\lambda) \rangle \ : \ K_A = K_B] \geq \varepsilon(\lambda) \ .$$

- (Security) For any PPT adversary Eve, any polynomial p, and any sufficiently large λ,

$$\Pr[(T, K_A, K_B) \xleftarrow{\$} \langle \mathsf{Alice}^{\mathsf{O}}(1^\lambda), \mathsf{Bob}^{\mathsf{O}}(1^\lambda) \rangle, K_E \xleftarrow{\$} \mathsf{Eve}^{\mathsf{O}}(1^\lambda, T) \ : \ K_E = K_B] \leq \frac{\varepsilon(\lambda)}{p(\lambda)} \ .$$

If security is only required to hold for infinitely many values of $\lambda \in \mathbb{N}$ in the sense that for every polynomial p and all adversaries, there exists an infinite set of values for λ for which the adversary's winning probability is below $\varepsilon(\lambda)/p(\lambda)$, then the construction is called an *infinitely-often* key agreement. If the number of queries to O is bounded by a constant for either Alice or Bob, then we say that the key agreement is *unbalanced* with respect to O. We say that the key agreement is *perfectly correct* if $\varepsilon(\lambda) \equiv 1$.

Definition 2.8 ((Approximate) indistinguishability obfuscation – (a)IO)**.** An IO scheme IO consists of two PPT machines $(\mathsf{Obf}, \mathsf{Eval})$ as follows.

- $\mathsf{Obf}(1^\lambda, C)$ takes as inputs a security parameter and a Boolean circuit C, and outputs D, which is usually also a circuit.
- $\mathsf{Eval}(D, x)$ takes as input D and an input x, and outputs a bit $b \in \{0, 1\}$.

We define the following correctness and security requirements.

- **Approximate correctness**: There is a function ε such that $\varepsilon(\lambda) \geq 1/\operatorname{poly}(\lambda)$ for sufficiently large λ and for every circuit C of input size n it holds that

$$\Pr[x \xleftarrow{\$} \{0,1\}^n; D \xleftarrow{\$} \mathsf{Obf}(1^\lambda, C) \ : \ D(x) = C(x)] \geq 1 - o_\lambda(1) \ .$$

- **Security**: For every sequences of circuit pairs (C_1, C_2) of the same size $|C_1| = |C_2|$, the ensembles

$$\mathsf{Obf}(1^\lambda, C_1) \qquad \text{and} \qquad \mathsf{Obf}(1^\lambda, C_2)$$

are computationally indistinguishable for $\operatorname{poly}(\lambda)$-sized distinguishers.

We retrieve the standard definition of IO when correctness holds with probability 1 for any λ. When clear from context we drop 1^λ as an input to Obf.

In this definition we could choose to define approximate correctness in other ways, for example by requiring approximate correctness to hold with probability $1/2 + 1/\operatorname{poly}(\lambda)$ or $1 - 1/\operatorname{poly}(\lambda)$. Our main results on black-box uselessness of various primitives for aIO (Theorems 5.6 and 5.7) also hold for these alternative definitions.

3 Defining Black-Box Uselessness

To formally define black-box uselessness, we first formalize joint primitives, to simplify statements about black-box constructions from several primitives:

Definition 3.1 (Joint primitive)**.** Given two primitives $\mathcal{P} = (F_\mathcal{P}, R_\mathcal{P})$ and $\mathcal{Q} = (F_\mathcal{Q}, R_\mathcal{Q})$ the joint primitive $(\mathcal{P}, \mathcal{Q}) = (F_{(\mathcal{P},\mathcal{Q})}, R_{(\mathcal{P},\mathcal{Q})})$ is defined by setting $F_{(\mathcal{P},\mathcal{Q})} := F_\mathcal{P} \times F_\mathcal{Q}$ and for each $\mathsf{G} = (\mathsf{P}, \mathsf{Q}) \in F_{(\mathcal{P},\mathcal{Q})}$ and any $A = (A_\mathcal{P}, A_\mathcal{Q})$ defining $(\mathsf{G}, A) \in R_{(\mathcal{P},\mathcal{Q})}$ iff $(\mathsf{P}, A_\mathcal{P}) \in R_\mathcal{P}$ or $(\mathsf{Q}, A_\mathcal{Q}) \in R_\mathcal{Q}$.

We are now ready to formally define what it means for a cryptographic primitive $\mathcal{Q}$ to be black-box useless for a primitive $\mathcal{P}$.

3.1 Definition

The following definition is more general than complete black-box uselessness of a primitive $\mathcal{P}$ for obtaining another primitive $\mathcal{Q}$. In particular, the definition states a *set* of primitives $\mathscr{Z}$ such that $\mathcal{P}$ is useless for obtaining $\mathcal{Q}$ in the presence of *any* of the primitives $\mathcal{R} \in \mathscr{Z}$.

Definition 3.2 (Black-box uselessness)**.** Let $\mathscr{Z}$ be a set of primitives. A cryptographic primitive $\mathcal{Q}$ is fully (resp., semi, $\forall\exists$-semi) *black-box useless* for constructing a primitive $\mathcal{P}$ in the presence of auxiliary primitives $\mathscr{Z}$ if for every auxiliary primitive $\mathcal{Z} \in \mathscr{Z}$ whenever there exists a fully (resp., semi, $\forall\exists$-semi) black-box reduction of $\mathcal{P}$ to the joint primitive $(\mathcal{Q}, \mathcal{Z})$ there also exists a fully (resp., semi, $\forall\exists$-semi) black-box reduction of $\mathcal{P}$ to $\mathcal{Z}$ alone. In the special case where $\mathscr{Z}$ contains *all* primitives, then we simply say that $\mathcal{Q}$ is fully (resp., semi, $\forall\exists$-semi) *black-box useless* for constructing $\mathcal{P}$.

Remark 3.3 (Other special cases)**.** Here we point out two other important special cases of Definition 3.2 that could be obtained by setting $\mathscr{Z}$ differently. For $\mathscr{Z} = \varnothing$, black-box uselessness is identical to traditional black-box separation (showing that $\mathcal{Q}$ cannot be black-box reduced to $\mathcal{P}$). In addition, when $\mathscr{Z}$ contains a specific primitive $\mathcal{Z}$, then the definition captures the notion that $\mathcal{P}$ is useless for building $\mathcal{Q}$ when we already assume the existence of $\mathcal{Z}$ as a black box.

Remark 3.4 (Other variants of Definition 3.2)**.** By default, we consider the three cases where the source construction and the target construction in Definition 3.2 both use the same flavor of black-box reduction (and accordingly use the terms fully, semi, or $\forall\exists$-semi to describe the corresponding notion of black-box uselessness). However, we can (and will) also consider more general notions of black-box uselessness whereby the source construction and the target construction use different notions of black-box reduction. For example, we will write that $\mathcal{Q}$ is [semi $\to \forall\exists$-semi] black-box useless for $\mathcal{P}$ if for every auxiliary primitive $\mathcal{Z} \in \mathscr{Z}$, whenever there exists a semi-black-box reduction of $\mathcal{P}$ to the joint primitive $(\mathcal{Q}, \mathcal{Z})$ there is a $\forall\exists$-semi-black-box reduction of $\mathcal{P}$ to $\mathcal{Z}$ alone.

3.2 Composition

Given the definition of black-box uselessness, the following composition theorem follows easily. Here, we use the term black-box uselessness to refer to any fixed flavor of black-box uselessness.

Theorem 3.5. *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be three cryptographic primitives. If $\mathcal{Q}$ is black-box useless for $\mathcal{P}$ and $\mathcal{R}$ is black-box useless for $\mathcal{P}$ (for the same flavor), then the joint primitive $(\mathcal{Q}, \mathcal{R})$ is black-box useless for $\mathcal{P}$ (for the same flavor).*

Proof. Let $\mathcal{Z}$ be an arbitrary auxiliary primitive. If there is a black-box reduction of $\mathcal{P}$ to the joint primitive $(\mathcal{Z}, (\mathcal{Q}, \mathcal{R})) = ((\mathcal{Z}, \mathcal{Q}), \mathcal{R})$, then by the black-box uselessness of $\mathcal{R}$ for $\mathcal{P}$, there is a black-box reduction of $\mathcal{P}$ to $(\mathcal{Z}, \mathcal{Q})$ (viewing $(\mathcal{Z}, \mathcal{Q})$ as an auxiliary primitive). In turn, using the black-box uselessness of $\mathcal{Q}$ for $\mathcal{P}$, we obtain a black-box construction of $\mathcal{P}$ from $\mathcal{Z}$ alone. Hence, $(\mathcal{Q}, \mathcal{R})$ is black-box useless for $\mathcal{P}$. □

We note that a similar composition theorem can be easily established even when $\mathcal{Q}$ and $\mathcal{R}$ do not satisfy the same flavor of black-box uselessness for $\mathcal{P}$, in the following sense: if a primitive $\mathcal{Q}$ is $[X \to Y]$ BBU for $\mathcal{P}$ and a primitive $\mathcal{R}$ is $[X' \to Y']$ BBU for $\mathcal{P}$, where X, Y, X', Y' are flavors of black-box reduction, then $(\mathcal{Q}, \mathcal{R})$ is $[X' \to Y]$ BBU for $\mathcal{P}$ as long as X a stronger flavor than Y' (e.g., X is "fully" and Y' is "semi").

3.3 Restricted Black-Box Uselessness

In many settings, it can be useful to consider a more general notion of black-box uselessness, which restricts the type of primitive (e.g., only infinitely-often variants) or the type of construction for which black-box uselessness is shown to hold. For readability, we will not define cumbersome formal notations for such variants, but instead will simply state the restriction explicitly when needed.

This generalization is especially useful to study the *efficiency* of black-box reductions. Indeed, black-box separations in cryptography are not limited to only showing their nonexistence. A more concrete treatment would make statements of the form "in any black-box construction of $\mathcal{P}$ from $\mathcal{Q}$, any implementation of $\mathcal{P}$ must call an implementation of $\mathcal{Q}$ at least q times," or for interactive primitives would state that "any black-box construction of $\mathcal{P}$ from $\mathcal{Q}$ must have at least r rounds." This approach to bounding the efficiency of generic cryptographic construction was initiated in the seminal work of Gennaro, Gertner, Katz, and Trevisan [GGKT05] and has subsequently proven very fruitful.

4 On the Black-Box Uselessness of OWFs for Key Agreement

In this section, we prove black-box uselessness of OWFs for key agreement for several special forms of key-agreement protocols. We leave the proof of black-box uselessness of OWFs for general key-agreement protocols as an intriguing open question.

4.1 Black-Box Uselessness of OWFs for Perfectly Correct Key Agreement

In this section, we prove the following result.

Theorem 4.1 (Black-box uselessness of OWFs for perfect unbalanced KA)**.** *Infinitely-often one-way functions are [semi $\to \forall\exists$-semi] black-box useless for infinitely-often perfect key agreement in any construction that is unbalanced with respect to the io-OWF.*

Before proving Theorem 4.1, let us breakdown its content. The "dream result" here would be to show that one-way functions are black-box useless for any key agreement. Unfortunately, we do not know how to prove this result. Theorem 4.1 provides a meaningful step in this direction, but it suffers from three limitations:

1. It only applies to *infinitely-often* one-way functions (though it shows black-box uselessness for infinitely-often key agreement, which is a weaker primitive).
2. It only applies to constructions where one of the parties makes a *constant* number of queries to the io-OWF oracle, which we call unbalanced key agreement. Note that the key agreement can still make an arbitrary number of queries to the auxiliary primitive.
3. It only applies to *perfectly correct* key agreement.

The first limitation stems from the fact that our proof of Theorem 4.1 relies on a case distinction based on the existence of one-way functions: if they exist, we get a construction of key agreement, else we get an attack on the candidate construction. However, this attack requires applying a one-way function inverter to several functions at once. But since a one-way function inverter is only guaranteed to succeed on *infinitely many* security parameters, which need not be equal across the different functions that we need to invert (and in fact could be exponentially far apart), this approach fails. To get around this, we rely on an inverter for an infinitely-often OWF, which gives an inverter which is guaranteed to work for *all* sufficiently large security parameters and we can use to simultaneously invert several functions. This, however, comes at the cost of obtaining black-box uselessness result for infinitely-often OWFs (for building infinitely-often key agreements). Such technicalities are relatively common in cryptography and stem from the asymptotic nature of primitives.

Remark 4.2. In general, statements of the form "$\mathcal{A}$ and $\mathcal{B}$ black-box imply $\mathcal{C}$" and statements of the form "io-$\mathcal{A}$ and io-$\mathcal{B}$ black-box imply io-$\mathcal{C}$", where io-$\mathcal{X}$ denotes an infinitely-often flavor of a primitive $\mathcal{X}$, can be incomparable for the trivial reason that io-$\mathcal{A}$ and io-$\mathcal{B}$ can never be simultaneously secure on the same security parameters. However, this situation does not arise in the setting of black-box uselessness, since the statement "io-$\mathcal{A}$ is BBU for io-$\mathcal{C}$" refers to the inexistence of black-box construction of io-$\mathcal{C}$ from io-$\mathcal{A}$ together with any other primitive $\mathcal{Z}$ – and not only "infinitely-often" types of primitives. In general, it is easy to show that the statement "$\mathcal{A}$ is BBU for $\mathcal{C}$" is stronger than (i.e., implies) the statement "io-$\mathcal{A}$ is BBU for io-$\mathcal{C}$" for all notions of black-box uselessness.

The second limitation stems from the fact that the proof requires to iteratively define efficient functions F_i, where each F_i builds upon an (efficient) OWF inverter applied to F_{i-1}. The total number of functions can be picked to be the minimum of the number of queries to the OWF made by either of the two parties. However, this argument crucially relies on the fact that the number of functions F_i is constant. To see this, imagine that we have at our disposal a OWF inverter that would always make a number of queries that is quadratic in the number of queries made by the function in the forward direction. Such an inverter would be efficient (i.e., it inverts any poly-time function in poly time), yet one cannot obtain a poly-time function by iteratively defining a function F_i which invokes $\mathsf{Inv}_{\mathsf{F}_{i-1}}$ unless i is constant, since the complexity of F_i grows as $\mathsf{runtime}(\mathsf{F}_1)^{2^i}$.

Eventually, our result in this section focuses on perfectly correct key agreement. We discuss the case of imperfect key agreement in Section 4.2.

4.1.1 A Helpful Logical Lemma

Below, we state a simple lemma which allows for more direct proofs of [semi $\to \forall\exists$-semi] black-box uselessness.

Lemma 4.3. *Let $\mathcal{P}$ and $\mathcal{Q}$ be two primitives. Then whenever the following statement is established, it implies in particular that $\mathcal{Q}$ is [semi $\to \forall\exists$-semi] black-box useless for $\mathcal{P}$:*

"Fix any primitive $\mathcal{Z}$ and any $\mathsf{Z} \in F_{\mathcal{Z}}$. Assume that there exists an oracle-aided PPT P_1 such that for any $\mathsf{Q} \in F_{\mathcal{Q}}$, $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}} \in F_{\mathcal{P}}$. Further assume that whenever (Q,Z) is a secure implementation of $(\mathcal{Q},\mathcal{Z})$, then $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}}$ is a secure implementation of $\mathcal{P}$. Then there exists an efficient implementation $\mathsf{P}_2^{\mathsf{Z}}$ of $\mathcal{P}$ relative to Z, and furthermore, whenever Z is a secure implementation of $\mathcal{Z}$, $\mathsf{P}_2^{\mathsf{Z}}$ is a secure implementation of $\mathcal{P}$."

Proof. If $\mathcal{Q}$ is semi-black-box useless for $\mathcal{P}$, by Definition 3.2 for any $\mathcal{Z}$ the following holds.

Assume that there is a semi-black-box reduction from $\mathcal{P}$ to $(\mathcal{Q},\mathcal{Z})$. This means that (1) there exists an oracle-aided PPT P_1 such that for any $(\mathsf{Q},\mathsf{Z}) \in F_{\mathcal{Q}} \times F_{\mathcal{Z}}$, $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}} \in F_{\mathcal{P}}$ (so in particular, if we fix any primitive $\mathcal{Z}$ and any $\mathsf{Z} \in F_{\mathcal{Z}}$, there exists an oracle-aided PPT P_1 such that for any $\mathsf{Q} \in F_{\mathcal{Q}}$, $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}} \in F_{\mathcal{P}}$). And (2) for any PPT oracle-aided A such that $A^{\mathsf{Q},\mathsf{Z}}$ $\mathcal{P}$-breaks $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}}$, there exists a PPT oracle-aided S such that $\mathsf{S}^{\mathsf{Q},\mathsf{Z}}$ $(\mathcal{Q},\mathcal{Z})$-breaks (Q,Z). Then for any $\mathsf{Z} \in F_{\mathcal{Z}}$, there exists an efficient implementation P_2 of $\mathcal{P}$ relative to Z, such that for any PPT oracle-aided A such that A^{Z} $\mathcal{P}$-breaks $\mathsf{P}_2^{\mathsf{Z}}$, there exists a PPT oracle-aided S such that S^{Z} $\mathcal{Q}$-breaks Z.

From here, the statement of Lemma 4.3 follows by observing that the statement "for any PPT oracle-aided A such that $A^{\mathsf{Q},\mathsf{Z}}$ $\mathcal{P}$-breaks $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}}$, there exists a PPT oracle-aided S such that $S^{\mathsf{Q},\mathsf{Z}}$ $\mathcal{Q}$-breaks (Q,Z)" is equivalent to "if there exists no PPT oracle-aided S such that $S^{\mathsf{Q},\mathsf{Z}}$ $\mathcal{Q}$-breaks (Q,Z) (i.e., if (Q,Z) is a secure implementation of $(\mathcal{Q},\mathcal{Z})$), then there exists no PPT oracle-aided A such that $A^{\mathsf{Q},\mathsf{Z}}$ $\mathcal{P}$-breaks $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}}$ (i.e., $\mathsf{P}_1^{\mathsf{Q},\mathsf{Z}}$ is a secure implementation of $\mathcal{P}$)". The same equivalence holds for the second part of the definition of $\forall\exists$-semi-black-box uselessness. □

4.1.2 Proof of Theorem 4.1

Let $\mathsf{io}\text{-}\mathcal{F}$ be the io-OWF primitive. To prove Theorem 4.1, we will prove the following:

Lemma 4.4. *Fix any primitive $\mathcal{Z}$ and any $\mathsf{Z} \in F_{\mathcal{Z}}$. Assume that there exists an oracle-aided PPT KA_1 such that for any implementation ioF of an infinitely-often one-way function, $\mathsf{KA}_1^{\mathsf{ioF},\mathsf{Z}}$ implements an infinitely-often perfect key agreement unbalanced with respect to ioF, relative to*

$(\mathsf{ioF}, \mathsf{Z})$. *Assume furthermore that if* $(\mathsf{ioF}, \mathsf{Z})$ *is a secure implementation of* $(\mathsf{io}\text{-}\mathcal{F}, \mathcal{Z})$*, then* $\mathsf{KA}_1^{\mathsf{ioF},\mathsf{Z}}$ *is a secure implementation of infinitely-often key agreement, unbalanced with respect to* ioF*. Then there exists an efficient implementation* KA_2 *of (infinitely-often) key agreement relative to* Z*, and furthermore, if* Z *is a secure implementation of* $\mathcal{Z}$*, then* $\mathsf{KA}_2^{\mathsf{Z}}$ *is a secure implementation of infinitely-often key agreement.*

The proof of Theorem 4.1 follows directly from the above lemma by applying Lemma 4.3. To prove Lemma 4.4, we rely on the following lemma.

Lemma 4.5. *Let* RO *be a random oracle. For any auxiliary oracle* Z*, if there exists no infinitely-often one-way function relative to* Z*, then there exists no construction* $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$ *of a perfect infinitely-often key agreement which is unbalanced with respect to* RO*.*

Given Lemma 4.5, the proof of Lemma 4.4 follows from a disjunction argument: fix any auxiliary primitive $\mathcal{Z}$ and any $\mathsf{Z} \in F_{\mathcal{Z}}$. Two complementary cases can occur:

- Either there exists an efficient implementation of an infinitely-often one-way function $\mathsf{ioF}^{\mathsf{Z}}$ relative to Z. By the assumption of Lemma 4.4, there exists an efficient implementation KA_1 of key agreement relative to $(\mathsf{ioF}', \mathsf{Z})$ for any $\mathsf{ioF}' \in F_{\mathsf{io}\text{-}\mathcal{F}}$. Define the following efficient construction $\mathsf{KA}_2^{\mathsf{Z}}$: $\mathsf{KA}_2^{\mathsf{Z}} := \mathsf{KA}_1^{\mathsf{ioF}^{\mathsf{Z}},\mathsf{Z}}$. By our assumption, this is therefore an efficient implementation of i.o.-key agreement relative to Z, which is also secure if $(\mathsf{ioF}, \mathsf{Z})$ is secure.
- Or there exists no efficient implementation of an infinitely-often one-way function $\mathsf{ioF}^{\mathsf{Z}}$ relative to Z. By Lemma 4.5, for a random oracle RO, there must therefore exist an efficient attack on $\mathsf{KA}_1^{\mathsf{RO},\mathsf{Z}}$. By Theorem 5.2 of [IR89], for measure 1 of the choices of the random oracle RO, RO is a one-way function and therefore in particular an io-OWF. (Note that this theorem also holds relative to an arbitrary oracle.) Therefore, KA_1 is not a secure implementation of key agreement with respect to any $(\mathsf{ioF}', \mathsf{Z})$ with $\mathsf{ioF}' \in F_{\mathsf{io}\text{-}\mathcal{F}}$, and by the assumptions of Lemma 4.5, $(\mathsf{RO}, \mathsf{Z})$ is not a secure implementation of $(\mathsf{io}\text{-}\mathcal{F}, \mathcal{Z})$. Since RO is a secure implementation of io-OWF, this implies that Z is not a secure implementation of $\mathcal{Z}$. Therefore, we can define KA_2 to be the trivial protocol in which Alice samples the output key and sends it to Bob. This is an efficient implementation of key agreement; it need not be secure since Z is not a secure implementation of $\mathcal{Z}$.

4.1.3 Proof of Lemma 4.5

It remains to prove Lemma 4.5. Consider a candidate construction $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$ of key agreement such that one of Alice and Bob makes a constant number of queries to RO. Let $\lambda \in \mathbb{N}$ be the security parameter, and consider a run $(T, K_A, K_B) \xleftarrow{\$} \langle \mathsf{Alice}^{\mathsf{RO},\mathsf{Z}}(1^\lambda), \mathsf{Bob}^{\mathsf{RO},\mathsf{Z}}(1^\lambda) \rangle$ of the construction $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$. We will describe an efficient attacker $\mathsf{Eve}^{\mathsf{RO},\mathsf{Z}}$ that breaks $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$ for infinitely many λ. The attack closely follows the (inefficient) strategy of [BKSY11] but relies on $\mathsf{Inv}^{\mathsf{Z}}$ to make the attack efficient. Without loss of generality, assume that Bob makes at most a constant number of queries q_B in any execution of the protocol. Furthermore, let $r_A(\lambda)$ and $r_B(\lambda)$ be (polynomial) bounds on the length of the random tapes of Alice and Bob respectively, and let $q(\lambda) = q_A(\lambda) + q_B$ be a (polynomial) bound on the total number of queries to RO made by both parties in any execution.

Lazy oracle sampling. Let $q \in \mathbb{N}$. For any string r of length q, and any list L of (query, answer) pairs, we let $\mathsf{SimRO}[L]_q(\cdot; r)$ be a stateful *lazy sampler* for a random oracle consistent with L. Namely, $\mathsf{SimRO}[L]_q(\cdot; r)$ works as follows. It maintains a counter i that is initialized to 1 and a list L' of (query, answer) pairs that is initially empty. Each time it receives an input x, $\mathsf{SimRO}[L]_q(x; r)$ first checks whether or not the query belongs to $L \cup L'$, and outputs the corresponding answers if this holds. If the query does not belong to L or L', algorithm $\mathsf{SimRO}[L]_q(x; r)$ defines q_i to be the answer to the query, adds (query, q_i) to L', and sets $i \leftarrow i + 1$. Note that for any interactive protocol Π^{RO} where the parties make less than q queries in total, and any list L of (query, answer) pairs consistent with RO, the distribution of the views of all parties obtained by sampling a random oracle RO and running Π^{RO} is identical to the distribution of the views of all parties obtained by sampling a q-bit string r and running Π while emulating RO using $\mathsf{SimRO}[L]_q(\cdot; r)$.

An inefficient attack. We first describe an inefficient attack on the candidate construction $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$, taken almost verbatim from [BKSY11]. The attacker $\mathsf{Eve}^{\mathsf{RO},\mathsf{Z}}$, given a transcript T of an execution of $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$, maintains a set Q_E of query/answer pairs for Z, and a multi-set of candidate keys $\mathcal{K}$, both initialized to $\varnothing$. Eve runs $2q_B + 1$ iterations of the following procedure.

- *Simulation phase:* Eve finds a view of $\mathsf{Alice}^{\mathsf{RO}',\mathsf{Z}}$ with respect to some (possibly different) oracle RO', consistent with the transcript T and all query/answer pairs in Q_E. This view contains a random tape r_A, the set of queries Q_A made by $\mathsf{Alice}^{\mathsf{RO}',\mathsf{Z}}$ (which is consistent with Q_{Eve}, but not necessarily with RO), and the key K_A computed by Alice. Eve adds K_A to $\mathcal{K}$.
- *Update phase:* $\mathsf{Eve}^{\mathsf{RO},\mathsf{Z}}$ makes all queries in Q_A to the true random oracle RO, and adds the results to Q_E.

After running $2q_B + 1$ iterations of the above attack, Eve has a multi-set $\mathcal{K}$ of $2q_B + 1$ possible keys; Eve outputs the majority value in $\mathcal{K}$. Observe that during each round of the attack, two events can happen:

1. Either one of the new queries (not already contained in Q_E) made by Alice in the simulated run was made by Bob in the real execution of the protocol. In this case, Eve discovers (and adds to Q_E) a new query of Bob.
2. Or none of the new queries of Alice was made by Bob in the real protocol, in which case there exists an oracle RO' which is consistent with the view of Bob in the real protocol, and the view of Alice in the simulated run. By perfect correctness, this means that the key K_A computed by Alice in this run is necessarily the correct key output by Bob.

Now, since Bob makes at most q_B distinct queries, the first of the two events can happen at most q_B times, hence the second event necessarily happens at least $q_B + 1$ times, which guarantees that the majority value in the multi-set $\mathcal{K}$ is indeed the correct key with probability 1.

The above attack requires $\mathcal{O}(q_A q_B)$ queries to RO. However, it requires finding a view for $\mathsf{Alice}^{\mathsf{RO}',\mathsf{Z}}$ that is consistent with a given transcript, where RO' is a simulated random oracle, but Z is the "true" auxiliary oracle. In general, this might require exponentially many queries to Z and hence $\mathsf{Eve}^{\mathsf{RO},\mathsf{Z}}$ is not necessarily an efficient oracle-aided algorithm. In the following, we show how to make the attack efficient given an inverter for io-OWFs.

Lazy protocol emulation. Let L be a list of (query, answer) pairs to RO. Given the construction $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$, let $\mathsf{SimC}^{\mathsf{Z}}_L$ be an oracle-aided PPT algorithm that emulates a run of Alice and Bob in $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$ that is consistent with L (but not necessarily with the rest of RO). That is, given a random string $r_A||r_B||q$ of length $r_A(\lambda) + r_B(\lambda) + q(\lambda)$, $\mathsf{SimC}^{\mathsf{Z}}_L(1^\lambda; r_A||r_B||q)$ runs Alice and Bob on input 1^λ and respective random tapes r_A and r_B, while using $\mathsf{SimRO}[L]_q(\cdot; q)$ to lazily emulate the random oracle RO. After completion of the protocol, SimC outputs the transcript T of the interaction, the lists (Q_A, Q_B) of all queries to RO made by Alice and Bob during the emulation of the protocol (together with their answers), and the outputs (K_A, K_B) of both parties. Observe that SimC corresponds to a valid interaction between $\mathsf{Alice}^{\mathsf{RO}',\mathsf{Z}}(1^\lambda; r_A)$ and $\mathsf{Bob}^{\mathsf{RO}',\mathsf{Z}}(1^\lambda; r_B)$ with respect to a random oracle RO' sampled uniformly at random, conditioned on being consistent with L.

The inverter. Since there is no infinitely-often OWF relative to Z, there exists an efficient inverter for any efficient oracle-aided function F^{Z}:

Lemma 4.6. *For any oracle-aided PPT function F^{Z} and any polynomial p, there exists an oracle-aided PPT inverter $\mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F},p}$ such that for all large enough $n \in \mathbb{N}$, it holds that*

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n}[\mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F},p}(\mathsf{F}^{\mathsf{Z}}(x), 1^n) \in (\mathsf{F}^{-1})^{\mathsf{Z}}(\mathsf{F}^{\mathsf{Z}}(x))] \geq 1 - \frac{1}{p(n)} \ .$$

Proof. This follows directly from the fact that the inexistence of io-OWFs relative to Z implies the inexistence of *weak* io-OWFs relative to Z (a weak OWF is a OWF where security is relaxed by saying that there exists a polynomial p such that no efficient adversary can invert with probability better than $1 - 1/p(n)$). The latter follows from standard hardness amplification methods as was initially proven by Yao [Yao82]. □

The sequence of functions. Let $p : \lambda \mapsto 1/(6q_B + 3)$ be a constant polynomial. We iteratively define a sequence of $2(q_B + 1)$ oracle functions $(\mathsf{F}^{\mathsf{Z}}_0, \mathsf{G}^{\mathsf{Z}}_0), \cdots, (\mathsf{F}^{\mathsf{Z}}_{q_B}, \mathsf{G}^{\mathsf{Z}}_{q_B})$ as follows.

- $\mathsf{F}^{\mathsf{Z}}_0$ gets as input a string $(r_A||r_B||q_0)$ of length $r_A(\lambda) + r_B(\lambda) + q(\lambda)$, computes

  $$(T, Q_A, Q_B, K_A, K_B) \leftarrow \mathsf{SimC}^{\mathsf{Z}}_{\varnothing}(1^\lambda; r_A||r_B||q) \ ,$$

  and outputs T. The function $\mathsf{G}^{\mathsf{Z}}_0$ is defined similarly, but it outputs (T, Q_A, Q_B, K_A) instead (without K_B).

- $\mathsf{F}^{\mathsf{Z}}_1$ gets as input a string $(r_A||r_B||q_0||q_1)$ of length $r_A(\lambda) + r_B(\lambda) + 2q(\lambda)$. First, it computes $(T, Q_A, Q_B, K_A) \leftarrow \mathsf{G}^{\mathsf{Z}}_0(r_A||r_B||q_0)$. Second, it sets $n \leftarrow r_A(\lambda) + r_B(\lambda) + q(\lambda)$ and runs $(r'_A||r'_B||q'_0) \leftarrow \mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F}_0,p}(T, 1^n)$. Third, it computes $(T', Q'_A, Q'_B, K'_A) \leftarrow \mathsf{G}^{\mathsf{Z}}_0(1^\lambda; r'_A||r'_B||q'_0)$. Eventually, it uses $\mathsf{SimRO}[Q_A \cup Q_B](\cdot; q_1)$ to lazily sample the answers to all queries contained in Q'_A, and stores the results in a set Q_E of pairs query/answer. $\mathsf{F}^{\mathsf{Z}}_1$ outputs (T, Q_E). We also define $\mathsf{G}^{\mathsf{Z}}_1$ to be the function defined as $\mathsf{F}^{\mathsf{Z}}_1$ except that it additionally outputs (Q_A, Q_B, K'_A).

- $\mathsf{F}^{\mathsf{Z}}_i$ gets as input a string $(r_A||r_B||q_0||\cdots||q_i)$ of length $r_A(\lambda) + r_B(\lambda) + (i+1) \cdot q(\lambda)$. First, it computes $(T, Q_E, Q_A, Q_B, K_A) \leftarrow \mathsf{G}^{\mathsf{Z}}_{i-1}(r_A||r_B||q_0||\cdots||q_{i-1})$. Second, it sets $n \leftarrow r_A(\lambda) + r_B(\lambda) + i \cdot q(\lambda)$ and runs $(r'_A||r'_B||q'_0||\cdots||q'_{i-1}) \leftarrow \mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F}_{i-1},p}((T, Q_E), 1^n)$. Third, it computes $(T', Q'_A, Q'_B, K'_A) \leftarrow \mathsf{G}^{\mathsf{Z}}_1(1^\lambda; r'_A||r'_B||q'_0)$. Eventually, it uses $\mathsf{SimRO}[Q_A \cup Q_B](\cdot; q_i)$ to lazily

sample the answers to all queries contained in Q'_A, and adds the results to Q_E. $\mathsf{F}_i^{\mathsf{Z}}$ outputs (T, Q_E). We also define $\mathsf{G}_i^{\mathsf{Z}}$ to be the function defined as $\mathsf{F}_i^{\mathsf{Z}}$ except that it additionally outputs (Q_A, Q_B, K'_A).

For readability, we also provide a pseudocode for the function $\mathsf{F}_i^{\mathsf{Z}}$ for $i \geq 1$ below.

function $\mathsf{F}_i^{\mathsf{Z}}(r_A||r_B||q_0||\cdots||q_i)$ ▷ $(r_A||r_B||q_1||\cdots||q_i)$ is of length $r_A(\lambda) + r_B(\lambda) + (i+1)\cdot q(\lambda)$
  $(T, Q_E, Q_A, Q_B, K_A) \leftarrow \mathsf{G}_{i-1}^{\mathsf{Z}}(r_A||r_B||q_1||\cdots||q_{i-1})$
  $n \leftarrow r_A(\lambda) + r_B(\lambda) + i \cdot q(\lambda)$
  $(r'_A||r'_B||q'_0||\cdots||q'_{i-1}) \leftarrow \mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F}_{i-1},p}((T, Q_E), 1^n)$ ▷ $p : \lambda \mapsto 1/(6q_B + 3)$
  $(T', Q'_A, Q'_B, K'_A) \leftarrow \mathsf{G}_1^{\mathsf{Z}}(1^\lambda; r'_A||r'_B||q'_0)$
  for $(x, y) \in Q'_A$ **do**
    $Q_E \leftarrow Q_E \cup (x, \mathsf{SimRO}[Q_A \cup Q_B](x; q_i))$
  end for
  return (T, Q_E) ▷ $\mathsf{G}_i^{\mathsf{Z}}$ is similar but additionally outputs (Q_A, Q_B, K'_A)
end function

Making the [BKSY11] attack efficient with Inv. To overcome the inefficiency of the attack of [BKSY11], we leverage the fact that, by assumption, there exists no infinitely-often one-way function relative to Z. As before, the attacker $\mathsf{newEve}^{\mathsf{RO},\mathsf{Z}}$, given a transcript T of an execution of $\mathsf{KA}^{\mathsf{RO},\mathsf{Z}}$, maintains a set Q_E of query/answer pairs for Z, and a multi-set of candidate keys $\mathcal{K}$, both initialized to $\varnothing$. Let $p : \lambda \mapsto (6q_B + 3)$ be a constant polynomial. newEve runs $2q_B + 1$ iterations of the following attack.

- *Simulation phase:* During the i-th round of attack, newEve does the following:
  1. *Finding a view of* Alice *consistent with* T *and* Q_E*:* newEve sets $n \leftarrow r_A(\lambda)+r_B(\lambda)+i\cdot q(\lambda)$ and computes $(r'_A||r'_B||q'_0||\cdots||q'_{i-1}) \leftarrow \mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F}_{i-1},p}((T, Q_E); 1^n)$.
  2. *Simulating the run of Alice with the view above:* newEve computes $(T', Q'_A, Q'_B, K'_A) \leftarrow \mathsf{G}_1^{\mathsf{Z}}(1^\lambda; r'_A||r'_B||q'_0)$.
  3. *Storing the key:* newEve adds K'_A to $\mathcal{K}$.
- *Update phase:* $\mathsf{Eve}^{\mathsf{RO},\mathsf{Z}}$ makes all queries in Q'_A to the true random oracle RO, and adds the results to Q_E.

After running $2q_B + 1$ iterations of the above attack, Eve has a multi-set $\mathcal{K}$ of $2q_B + 1$ possible keys; Eve outputs the majority value in $\mathcal{K}$. We now analyze the success probability of the attack.

Claim 4.7. newEve *outputs the correct key with probability at least* $2/3$.

First, observe that by definition of $\mathsf{F}_0^{\mathsf{Z}}$, the transcripts T in a real execution of the protocol (which newEve gets as input) are distributed identically to $\mathsf{F}_0^{\mathsf{Z}}(r_A||r_B||q_0)$ for uniformly random (r_A, r_B, q_0), where r_A (resp., r_B) is the real random tape of $\mathsf{Alice}^{\mathsf{RO},\mathsf{Z}}$ (resp., $\mathsf{Bob}^{\mathsf{RO},\mathsf{Z}}$) and q_0 is the ordered string of all answers of RO to distinct queries from Alice and Bob. Therefore, by the definition of $\mathsf{Inv}^{\mathsf{Z}}_{\mathsf{F}_0,p}$, the tuple $(r'_A||r'_B||q'_0)$ computed in the first iteration of the attack is consistent with the real transcript T with probability at least $1 - 1/p = 1 - 1/(6q_B + 3)$.

Consider now the set Q_E of queries obtained by newEve after the update phase of the first iteration. (T, Q_E) is distributed exactly as $\mathsf{F}_1^{\mathsf{Z}}(r_A||r_B||q_0||q_1)$ for uniformly random (r_A, r_B, q_0, q_1).

This is because Q_E in $\mathsf{F}_1^{\mathsf{Z}}$ is computed by lazily sampling the answers of a random oracle, conditioned on being consistent with all queries made by Alice and Bob in the execution of $\mathsf{F}_0^{\mathsf{Z}}(r_A||r_B||q_0)$. The real run of the protocol corresponds to an execution of $\mathsf{F}_0^{\mathsf{Z}}$ on a random input $(r_A||r_B||q_0)$, and making the queries in Q'_A to RO is identical to lazily sampling RO while being consistent with q_0 (i.e., the query/answer pairs obtained by Alice and Bob in the real execution). Therefore, the tuple $(r'_A||r'_B||q'_0||q'_1)$ which newEve computes in the second iteration of the attack is consistent with the real transcript T with probability at least $1 - 1/p = 1 - 1/(6q_B + 3)$.

More generally, the distribution of (T, Q_E) obtained by newEve during the i-th iteration of the attack after receiving the transcript T of a real execution of the protocol is distributed exactly as $\mathsf{F}_i^{\mathsf{Z}}(r_A||r_B||q_0||\cdots||q_i)$ for uniformly random $(r_A, r_B, q_0, \cdots, q_i)$, hence the tuple $(r'_A||r'_B||q'_0||\cdots||q'_i)$ which newEve computes in the $(i+1)$-th iteration of the attack is consistent with the real transcript T with probability at least $p = 1/(6q_B + 3)$.

Putting everything together, after finishing the attack, by a straightforward union bound, all simulated views computed by newEve during the attack are consistent with T with probability at least $1 - (2q_B + 1) \cdot 1/(6q_B + 3) = 2/3$. When this happens, by the same argument as for the inefficient attack, the majority key in $\mathcal{K}$ is necessarily the correct key, and the claim follows.

Claim 4.8. *The number of queries made by* newEve *to* RO *and* Z *is bounded by a polynomial.*

As in the inefficient attack, newEve makes at most $\mathcal{O}(q_A q_B) = \mathcal{O}(q_A)$ queries to RO. The polynomial bound on the number of queries to Z follows from the efficiency of Inv: $\mathsf{Inv}_{\mathsf{F}_{0},p}^{\mathsf{Z}}$ is efficient by definition, hence $\mathsf{F}_1^{\mathsf{Z}}$ (which invokes $\mathsf{Inv}_{\mathsf{F}_{0},p}^{\mathsf{Z}}$ internally) is an efficient function, from which we get that $\mathsf{Inv}_{\mathsf{F}_{1},p}^{\mathsf{Z}}$ is also efficient, and so on, and the claim follows (note that this crucially relies on our assumption that the protocol is unbalanced, and therefore q_B is constant).

4.2 Black-Box Uselessness of OWFs for Imperfect KA

In this section, we discuss how our result of the previous section can be extended to the case of imperfect key agreement. More precisely, we provide a sketch of how to modify our previous proof to show the following:

Theorem 4.9. *Infinitely-often one-way functions are [semi* $\to \forall\exists$*-semi] black-box useless for infinitely-often perfect key agreement in any construction where:*

- *Both parties make a constant number of queries to the io-OWF (but any polynomial number of queries to the auxiliary oracle);*
- *The key agreement protocol has a constant number of rounds.*

Proof sketch. The natural approach to extend our result is to replace the attack of [BKSY11] by an attack that applies to any imperfect key agreement protocol in the random-oracle model, such as those of Impagliazzo and Rudich [IR89] or Barak and Mahmoody [BM09, BM17], making the same case distinction based on the existence of io-OWFs to make the attack efficient. The structures of these attacks are very similar, though more involved than the attack of [BKSY11]: they proceed in a sequence of steps, where each step has a simulation phase, in which the attacker samples views consistent with (a portion of) the transcript and a set of queries, and an update phase, where the attacker makes some queries based on the simulated run.

However, two important technicalities arise when modifying our previous proof with the attacks of [IR89, BM09].

First, in the simpler attack of [BKSY11], perfect correctness guarantees that finding *any* consistent view is sufficient; in the attacks of [IR89, BM09], on the other hand, the attacker is required to *sample* views from a distribution close to the uniform distribution over views conditioned on a transcript and a set of queries. This can still be achieved assuming only the inexistence of io-OWFs: the inexistence of io-OWFs further entails the inexistence of *distributional* io-OWFs [IL89]. Namely, we must rely on the following lemma, a proof of which can be found in [BHT14].

Lemma 4.10. *Assume that there exists no infinitely-often one-way function relative to* Z*. Then for any efficient oracle-aided function* F^{Z} *and any polynomial* p*, there exists an inverter* $\mathsf{Inv}_{\mathsf{F}}^{\mathsf{Z}}$ *such that for all large enough* $n \in \mathbb{N}$*,*

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n, y \leftarrow \mathsf{F}^{\mathsf{Z}}(x)} \left[\mathsf{SD}\left((\mathsf{F}^{-1})^{\mathsf{Z}}(y), \mathsf{Inv}_{\mathsf{F}}^{\mathsf{Z}}(y, 1^n) \right) > \frac{1}{p(n)} \right] \leq \frac{1}{p(n)} \ ,$$

where SD *denotes the statistical distance between the two distributions.*

Second, the attacks of [IR89, BM09] proceed in a number of steps that grow with the number of queries of *both* parties to the random oracle *and* the round complexity of the protocol. More precisely, the attack requires executing several simulation and updates phases (of the order of $\tilde{\mathcal{O}}(q_A q_B)$, where q_A, q_B bound the respective number of queries of Alice and Bob to the random oracle) for each round of the protocol, where the simulation phase for round i inverse samples views consistent with the set of queries made by the attacker so far and the transcript of the protocol *up to round* i. This means that for the attack to be efficient, the key agreement must be constant round, and both parties must make a constant number of queries to the random oracle.

From here, a proof of Theorem 4.9 follows by fixing a (constant) bound B on the total number of steps of the (inefficient) attacker, and using the inverse sampler guaranteed by Lemma 4.10 for statistical distance $1/(10B)$ to make it efficient, similarly to our previous proof. By a union bound over all steps, the B steps of the inverse sampling will simultaneously guarantee that with probability at least $1/10$, at any round i, no intersection query between Alice and Bob made prior to round i was missed by the attacker. This allows us to conclude that the overall success probability of the efficient attacker (which uses the inverse sampler) is at most one-tenth of the success probability of the inefficient attacker described in [IR89, BM09].

With these technicalities in mind, this proof shows that io-OWFs are [semi $\rightarrow \forall\exists$-semi] black-box useless for constant-query constant-round constructions of imperfect key agreement. □

We note that a general technique was described in [MMV11] which allows to reduce the number of rounds (i.e., the adaptivity) of the attacker against a large variety of protocols (referred to as *puzzles* in [MMV11] and capturing in particular key-agreement protocols). It might appear at first sight that this approach could be used to remove the restriction of protocols being constant round (though still requiring both parties to make a constant number of queries to the OWF). However, this technique requires the attacker with lower adaptivity to simulate in "its head" the original attacker which cannot be done efficiently if the protocol was not constant-round. We leave as the main open problem of our work the question of showing that (infinitely-often) OWFs are black-box useless for more general forms of key agreement.

4.3 Black-Box Uselessness of OWFs for Merkle-Type Key Agreement

In this section, we prove that OWF are black-box useless for building key-agreement protocols where both parties place arbitrary many queries to both the OWF oracle, but are restricted in that the parties place oracles queries, exchange message, and then compute keys and terminate without placing any further queries. We call such protocols Merkle-type [Mer78], since Merkle's celebrated protocol (that achieves only polynomial security) is of this form.

Definition 4.11 (Merkle-type protocols)**.** We call a key agreement in the RO model (or based on OWFs) with an auxiliary oracle Z a *Merkle-type protocol* if Alice and Bob ask all of their queries to the RO before exchanging messages.

We emphasize that the definition above only restricts parties' access to the oracle that implements the OWF (e.g., a random oracle) and queries to Z are not restricted.

Theorem 4.12 (OWFs are black-box useless for Merkle-type key agreement)**.** *One-way functions are [semi* $\to \forall\exists$*-semi] black-box useless for any Merkle-type protocol.*

We prove a slightly more general result that allows Bob to wait for Alice's message T_A before asking its own queries and sending out a message T_B back to Alice. These protocols in particular encompass public-key encryption schemes for which the decryption algorithm does not call F. Thus, we obtain a BBU result for such PKE schemes. More precisely, we prove our result for any protocol that takes the following structure.

- $\mathsf{Alice}_1^{\mathsf{F},\mathsf{Z}}(1^\lambda)$: Alice places queries to the OWF oracle F and the auxiliary oracle Z, computes state st_A, sends a message T_A to Bob.
- $\mathsf{Bob}^{\mathsf{F},\mathsf{Z}}(1^\lambda, T_A)$: Bob receives a message T_A from Alice, places queries to F and the auxiliary oracle Z. Bob sends a message T_B to Alice, outputs a key K_B, and terminates.
- $\mathsf{Alice}_2^{\mathsf{Z}}(1^\lambda, T_B, st_A)$: Alice receives a message T_B from Bob, does not query its OWF oracle F any more, but may place queries to Z. It outputs a key K_A and terminates.

In the rest of this section we prove Theorem 4.12. We start with a high-level overview.

Intuition. We follow the same proof strategy for perfect KA (Theorem 4.1). Namely, we will make a distinction between two cases:

Case (1) : There is a construction of OWFs relative to Z: In this case, we just use this construction to realize the OWF primitive and hence would realize our key agreement based on Z only.

Case (2) : There are no constructions of OWFs relative to Z, which means that for any efficient algorithm relative to Z, there is an inversion algorithm. Similarly to the argument in Section 4.2, we will use the result of [IL89] and obtain something stronger that we will crucially rely upon: for any Z-aided PPT machine M, there is an efficient algorithm Inv that, given $\mathsf{M}(x) = y$ for a uniformly random $x \xleftarrow{\$} \{0,1\}^\lambda$, *almost uniformly samples* from $\mathsf{M}^{-1}(y)$, the set of preimages of y. Looking ahead, since we will use the existence of such inverters *only once*, we will no longer need to state our result with respect to the infinitely often variants of the primitives involved.

In what follows, we assume that we are in Case (2) above, i.e., that no OWFs exist relative to Z. This is because Case (1) finishes the proof directly as explained. Our goal would be to devise a polynomial-time attack on the security of the key agreement when we instantiate the one-way function using a random oracle.

In the following, we first describe a poly-query attack on the key agreement in the RO model when there are no Z oracles. Then, we show how to make this attack polynomial time relative to Z, assuming that we are in Case (2).

In this case, we first show how Eve can *efficiently* compute a list L that contains the heavy queries of Bob with respect to the random oracle (efficiency of this step will be crucial later on as we will add the oracle Z back and implement the attack in polynomial time). This is done by running Bob multiple times using the *actual* random oracle RO and on the input T_A. Using standard probabilistic arguments, one can show that conditioned on the set L, (the first stage of) Alice and Bob will have a small chance of having any intersection queries outside L. Eve will then wait for the actual Bob generating T_B, and then it "inverts" the process of the real Bob by sampling coins and a random oracle for it conditioned on T_A, T_B, and L. It then simply outputs the key that this sampled Bob outputs. It is at this stage that we will use an inverter to invert Bob's process given (T_A, T_B, L).

The formal proof. Consider an algorithm $\mathsf{Eve}^{\mathsf{RO}}_{\varepsilon}(T_A, T_B)$ that operates to break the security of Merkle-type protocols as follows.

1. *Learn Bob's queries:* Run $\mathsf{Bob}^{\mathsf{RO}}(T_A; r'_B)$ on independent random coins r'_B for $1/(n \cdot \varepsilon)$ times. Record the at most n^2/ε queries that Bob makes to its random oracle in a list L consisting of (query,answer) pairs.

2. *Sample Bob:* Invert Bob's view conditioned on (T_A, T_B, L). This requires waiting for Bob's message, and also involves probably a shadow sample for Alice's view that we will not use.

3. *Compute key:* Output the key that sampled Bob in the previous step produces.

We first analyze the above attack in the RO model, and then we show how to mimic this attack in a Z-relativized world, assuming that no OWFs relative to Z exist.

Let $T := (T_A, T_B)$ be the transcript of the protocol. We claim that the distributions

$$(\mathsf{Bob} \mid T, L) \qquad \text{and} \qquad (\mathsf{Bob} \mid T, L, \text{View}[\mathsf{Alice}_1])$$

are $\mathcal{O}(\varepsilon)$ statistically close. For fixed $(\text{View}[\mathsf{Alice}_1], L, T)$, we define the event $\text{Bad}(\text{View}[\mathsf{Bob}])$ over a candidate view of Bob (which may be real or fake) to hold iff the queries in $\text{View}[\mathsf{Bob}]$ intersect with those of Alice outside L. Now, for sake of the proof, imagine the following imaginary Bob algorithm that we refer to the "fake Bob" algorithm. This algorithm will only stick to L and ignores the execution of the algorithm on Alice's side. It also tosses coins for any new random oracle query that is not answered in L.

A key observation is that the distribution of $(\mathsf{Bob} \mid T_A, L, \text{View}[\mathsf{Alice}_1], \neg\text{Bad}(\mathsf{Bob}))$ is identical to the distribution of $(\mathsf{Fake\text{-}Bob} \mid T_A, L, \neg\text{Bad}(\mathsf{Fake\text{-}Bob}))$. That is, Bob and $\mathsf{Fake\text{-}Bob}$ are the same processes until Bad happens for them. Thus Pr[Bad] is the same for both of them and conditioned on Bad not happening, they implement the same process.

Fact 4.13. *Let $X_1, \ldots, X_k$ be k i.i.d. Bernoulli random variables. Then*[10]

$$\Pr[X_1 = X_2 = \cdots = X_{k-1} = 0 \wedge X_k = 1] = (1-p)^k \cdot p \leq \frac{1}{ek} \ .$$

By Fact 4.13, we already know that the probability of Bad happening for the real Bob is at most ε. This follows by bounding the probability of Bad in every query of Alice, and applying the union bound.

We conclude that the distribution of the real Bob *conditioned on Alice's view* is ε-close to the distribution of the fake Bob, which in turn is *independent of Alice's view* once we fix T, L. This means that the views of Alice and Bob are $\mathcal{O}(\varepsilon)$-close to being a product distribution, which means that conditioning or not conditioning on Alice will change Bob's distribution only by $\mathcal{O}(\varepsilon)$ in statistical distance (on average). As a result, Bob's view that Eve samples in the last step of the attack is $\mathcal{O}(\varepsilon)$-close to the distribution of Bob, even conditioned on the real view of Alice (even though Eve does not have access to that view and hence does *not* condition its sample on it). This finishes the proof of why Eve works in the random-oracle model.

Finishing the proof using inverse samplers. All we need to do now is to observe that the above proof can be used to derive an attack on any Merkle-type key agreement in the RO model in the presence of Z, if we are in Case (2). The reason is as follows. First, the initial step of Eve can easily be implemented relative to Z as well (because we can still run Bob efficiently in the RO model and relative to Z multiple times). The challenge is that we cannot necessarily invert the process and sample a view for Bob conditioned on the given (T_A, T_B, L). This, however, can be done in polynomial time if no one-way (and hence no *distributional* one-way) functions exist relative to Z. More formally, we define an efficient process M which runs the protocol by running Alice and Bob, followed by Eve's learning of the heavy queries. The actual input to this process is only the randomness for Alice and Bob as well as sufficient coins for simulating the random oracle. M then outputs (T_A, T_B, L). Since we are in Case (2), we can indeed invert this efficient process and sample a *uniform* preimage, by running an efficient oracle-aided algorithm using Z. As a result, we can indeed use an inverse-sampler Inv for M and efficiently implement the last step of Eve, even in the presence of Z.

5 Black-Box Uselessness via Compiling Out

In this section, we show that a number of black-box separation results proved in the literature [GGKT05, CKP15, GMM17a, GMM17c, BLP17, AKW18, GHMM18] can be lifted to the black-box uselessness setting.

At a very high level, what unifies these works is that they all follow the same blueprint. They all "compile out" a primitive $\mathcal{P}$ from any construction of $\mathcal{Q}$ from $\mathcal{P}$ and obtain a closely related construction of $\mathcal{Q}$ in the plain model, thereby showing that $\mathcal{P}$ was not needed for $\mathcal{Q}$ to begin with. Our key observations in showing how these results can be lifted to the black-box uselessness setting are as follows:

1. The compiling-out process relativizes; namely it holds even if an auxiliary oracle Z is present in the process.

[10] The displayed inequality follows by differentiating the bound with respect to p to obtain the maximum at $p = 1/k$ and then using the fact that $(1-1/k)^k \leq 1/e$ for $k > 0$.

2. The new construction, obtained through the compiling out process applied to a fully black-box construction, is *oblivious* to the auxiliary oracle Z. This observation is needed if we want to obtain a (normal) black-box uselessness result that does not suffer from only being a $\forall\exists$ variant. Note that a primitive $\mathcal{Z}$ might have many implementations, and we want the obtained construction (after the compilation process) to be the same for all such implementations.

For concrete examples, we show how to apply the above blueprint to results from [GGKT05, CKP15, GMM17a, GMM17c, GHMM18]. In [BLP17, AKW18] it was shown how to compile out random oracles or graded encoding schemes from variants of functional encryption schemes, and we believe our techniques would allow extending such results also to black-box uselessness from primitives implied by random oracle or bilinear maps. However, for concrete statements, we only focus on results from [GGKT05, CKP15, GMM17a, GMM17c, GHMM18].

5.1 Black-Box Uselessness in (Query) Efficient Constructions

We will use the following result proved in [GGKT05]. A useful observation is that, although the work of [GGKT05] proved this result for the empty auxiliary oracle $\mathsf{Z} = \bot$, but the same exact proof works for any fixed auxiliary oracle Z as well. Namely, this result relativizes.

Theorem 5.1 (Gennaro et al. [GGKT05])**.** *Let $t(n) \in [1, n]$ be a function of n. Let $\Pi_{t,n}$ be a (family of) length-preserving oracle sampled as follows. A sampled $\pi_{t,n} \leftarrow \Pi_{t,n}$ is a permutation over $\{0,1\}^n$ in which the first $t(n)$ bits of the input are mapped to the first $t(n)$ bits of the output using a random permutation, and the remaining $n - t(n)$ bits are left unchanged. For simplicity, we call this oracle (distribution) Π_t. Then, if $t(n) = \omega(\log n)$, then there are negligible $\varepsilon(n)$ and super-polynomial $s(n)$ functions such that, for* any *auxiliary oracle Z, with probability $1-\varepsilon(n)$ over the sampling of $\pi_t \leftarrow \Pi_t$, it holds that π_t is one-way against all $s(n)$-sized circuits that are allowed to have π_t gates as well as Z gates in their computation.*

Theorem 5.2 (Black-box uselessness of OWPs for query efficient PRGs)**.** *Suppose $\mathcal{Q}$ is the primitive of OWPs and $\mathcal{P}$ is the primitive of PRGs with stretch $\ell = \ell(\lambda)$ (meaning that an implementation P maps $\{0,1\}^\lambda \to \{0,1\}^{\lambda+\ell}$ for all λ). Suppose we limit fully black-box reduction of $\mathcal{P}$ to $\mathcal{Q}$ as follows. The implementation reduction P, on inputs of length λ, calls Q at most $o(\ell/\log\lambda)$ times on inputs of length $n = \mathrm{poly}(\lambda)$. Then $\mathcal{Q}$ is useless for $\mathcal{P}$ (for such class of black-box reductions).*

Proof sketch. Here we sketch the key ideas of the proof and refer the reader to the full version for the full proofs.

The proof is an adaptation of the proof of [GGKT05] to the context of black-box uselessness. For simplicity, we use ℓ to denote $\ell(\lambda)$. At a high level, [GGKT05] showed how to compile out the random permutation from any $o(\ell/\log\lambda)$-query black-box construction of PRGs ℓ bits of stretch. Here we observe that: (1) the proof of [GGKT05] relativizes and holds in the presence of any auxiliary oracle Z, and (2) that the produced black-box reduction after the compiling out process does *not* depend on the specific oracle Z.

Suppose, for the sake of contradiction, that $(\mathsf{PRG}, \mathsf{S})$ is a fully black-box construction of PRGs with stretch $\ell(\lambda)$ from one-way permutations and an auxiliary primitive $\mathcal{Z}$ with only $q = q(\lambda) = o(\ell/\log\lambda)$ number of queries to the OWP oracle $\pi\colon \{0,1\}^n \mapsto \{0,1\}^n$ for $n = \mathrm{poly}(\lambda)$.

We first prove the following claim.

Claim 5.3. *We can choose $t(\lambda) = \omega(\log\lambda)$ such that $q(\lambda) \cdot t(n(\lambda)) < \ell(\lambda)$.*

Proof. Let $t(\cdot)$ be such that $t(n(\lambda)) = \ell(\lambda)/(2q(\lambda))$. Then the following two items hold.

1. $q \cdot t(n(\lambda)) = \ell/2 < \ell$ holds by the definition of $t(\cdot)$.
2. $t(\lambda) = \omega(\log \lambda)$ holds, because (1) $\log(n(\lambda)) = \Theta(\log \lambda)$, and (2) $q(\lambda) = o(\ell(\lambda)/\log \lambda)$ which means $t(n(\lambda)) = \ell(\lambda)/(2q(\lambda)) = \omega(\log \lambda)$. □

We now continue with the proof of Theorem 5.2. We will describe a new construction that only uses $\mathcal{Z}$. We first describe its implementation reduction, and then we show that the security reduction (for this implementation reduction) exists.

The implementation reduction. Consider a new PRG implementation PRG' solely using $\mathcal{Z}$. PRG' that does not call the OWP oracle at all, but rather it takes an input of length $\lambda + \ell - 1$ and uses it as follows to produce $\lambda + \ell$ output bits (which still constitutes one bit of stretch). Suppose $x' = (x, r)$ is an input to PRG' where $|x| = \lambda$ and $|r| = \ell(\lambda) - 1$. $\mathsf{PRG}'^{\mathsf{Z}}(x')$ will emulate $\mathsf{PRG}^{(\cdot),(\cdot)}(x)$, in which PRG needs oracle access to implementations of OWP as follows. It forwards any oracle call by to the implementation of $\mathcal{Z}$ to its own implementing oracle Z, but whenever there is a query to the OWP oracle π, it uses (the remaining unused part of) r to simulate this query according to the oracle π_t described in Theorem 5.1. First note that by Claim 5.3, the extra randomness r is long enough for emulating all the oracle answers to the oracle π_t. So, all we need to do is to analyze the pseudo-randomness of the input. Then, one can amplify the one-bit stretch of the provided PRG construction to arbitrarily long stretch as well.

The security reduction. Suppose A^{Z} is a PPT adversary that PRG-breaks $\mathsf{PRG}'^{\mathsf{Z}}$. We will show that there is an oracle-aided PPT Sim such that $\mathsf{Sim}^{\mathsf{Z},A}$ $\mathcal{Z}$-breaks Z. From A's perspective it is the same if PRG' uses r to emulate an $\pi_t \leftarrow \Pi_t$ oracle or to use an actual $\pi_t \leftarrow \Pi_t$ oracle. So, the same A will PRG-break $\mathsf{PRG}^{\pi_t,\mathsf{Z}}$ over inputs sampled as $x \leftarrow \{0,1\}^\lambda$ and OWP oracles $\pi_t \leftarrow \Pi_t$.

By the weakly black-box security property of the implementation PRG, there should exist an oracle-aided PPT S such that $S^{\pi_t,\mathsf{Z},A}$ either OWP-breaks π_t or $\mathcal{Z}$-breaks Z. The former can only happen with a negligible probability by Theorem 5.7 (and for sufficiently large λ, n). Therefore, the only possibility is that $S^{\pi_t,\mathsf{Z},A}$ $\mathcal{Z}$-breaks Z. Therefore, if we let S' be a different reduction that emulates π_t in its head (according to the definition of Π_t), it will still break Z with non-negligible $\rho(n) \cdot (\rho(n) - \mathrm{negl}(n))$ probability.

□

An alternative approach for in the case of OWFs. In the proof of Theorem 5.2 we used the single-bit stretching PRG that is realized from the primitive $\mathcal{Z}$ to get a PRG with stretch ℓ. There is a less direct approach to obtain the final result *if* we did *not* want to get rid of one-way *permutations*, but rather we wanted to do this for one-way functions. In that case, we could use a (fully) black-box construction of one-way functions from single-bit stretching PRGs (that is being realized from the primitive $\mathcal{Z}$). We would then plug in this construction and use it to realize the OWF primitive that is used in the original construction of PRGs from OWFs and $\mathcal{Z}$. This approach does not work when we want to prove the black-box uselessness of OWPs, since there are no black-box constructions of OWPs from OWFs [Rud88, KSS00]. However, as we shall see below, this idea is useful for proving the black-box uselessness of OWFs for public-key encryption when the message space is large and the number of queries to OWFs is small (see Theorem 5.5).

The other results stated in the following theorem can be obtained by adapting the proofs of [GGKT05] as in the proof of Theorem 5.2 by observing that the proofs of [GGKT05] relativize to any oracle Z, and that the obtained construction is oblivious to Z. Then, one can compile out the OWP and obtain a construction of a primitive $\mathcal{R}$ (in particular OWFs) relative to Z alone. Then, in all these cases, the obtained construction of $\mathcal{R}$ would be enough to obtain the final goal primitive in a fully black-box way. By combining these two steps, we get that $\mathcal{A}$ can be obtained from Z in a black-box way.[11]

Theorem 5.4 (Black-box uselessness in efficient constructions)**.** *In all cases below, $\mathcal{Q}$ is the one-way permutation primitive and is fully black-box useless for $\mathcal{P}$, when the construction's implementation calls $\mathcal{Q}$ at most k times.*

- **Universal one-way hash functions***: $\mathcal{P}$ is a universal one-way hash function that uses randomness of length $r = r(\lambda)$ and shrinks its inputs from length $\lambda + \ell(\lambda)$ to the length λ. The limitation on the construction is that the implementation reduction asks $k = o(\ell/\log \lambda)$ queries to the OWP oracle.*
- **Digital signatures***: $\mathcal{P}$ is a signature scheme with key length λ and message space $\ell = \ell(\lambda)$. The limitation on the construction is that the implementation reduction asks $k = o(\ell/\log \lambda)$ queries to the OWP oracle.*
- **Private-key encryption***: $\mathcal{P}$ is a private-key encryption scheme with key length λ and message space of $\ell = \ell(\lambda)$ bits. The limitation on the construction is that the implementation reduction asks $k = o((\ell - \lambda)/\log \lambda)$ queries to the OWP oracle.*

Theorem 5.5 (Black-box uselessness of OWFs for efficient PKE with long messages)**.** *OWFs are fully black-box useless for constructing public-key encryption if messages are of length ℓ and the implementation reduction calls OWF at most $o(\ell/\log \lambda)$ times, where λ is the security parameter.*

The proof of the above theorem follows the blueprint of Theorem 5.2, but here we sketch the steps. Here, after compiling out the OWP, as shown in [GGKT05], one obtains a *private*-key encryption scheme whose keys are shorter than the messages it encrypts. However, [GGKT05] shows how to obtain one-way functions from this primitive. This proof relativizes with respect to any auxiliary oracle and with a construction that is oblivious to this auxiliary oracle. Furthermore, one-way functions can be used to obtain private-key encryption in a black-box way. So, if we choose to work with one-way *functions* (instead of permutations) to start with, we get that OWFs are black-box useless for efficient PKEs with large messages.

5.2 Black-Box Uselessness of OWFs for Approximate IO

In this section, we state black-box uselessness results that can be obtained by lifting the results of [CKP15, GMM17a, GMM17c]. These results deal with assumptions behind (and separations for) IO. In fact, to state the results with regard to black-box uselessness, we need to work with approximately correct IO (see Definition 2.8). We prove the following result formally, and explain how Theorem 5.7 can be enhanced to black-box uselessness similarly. At a very high level, the blueprint is the same as that in Section 5.1: we observe that the compilation technique used

[11] One exception to this is Theorem 5.5 for which we use the obtained OWF to instantiate the primitive that we want to actually avoid using and show its black-box uselessness.

in [CKP15, GMM17a, GMM17c] hold relative to any auxiliary oracle Z, and that the compiling-out process is oblivious to Z.

Theorem 5.6 (Black-box uselessness of OWFs for aIO)**.** *OWFs are fully black-box useless for approximate IO.*

Proof. The proof follows the blueprint of the proof of [CKP15]. Suppose there is a fully black-box reduction $(\mathsf{IO}, \mathsf{S})$ from of aIO to the composition of OWFs and another primitive $\mathcal{Z}$. Let O be a random oracle and Z be any secure implementation of $\mathcal{Z}$. Since $\mathsf{F} \leftarrow \mathsf{O}$ will be one-way with measure one [IR89], even in the presence of Z, it would imply that relative to $\mathsf{F} \leftarrow \mathsf{O}$ and any secure implementation Z of $\mathcal{Z}$ the construction $(\mathsf{IO}, \mathsf{S})$ would give a secure aIO. Here we show how to compile out the random oracle from the construction and get a new one in the presence of Z (whose algorithm does work for any implementation Z of $\mathcal{Z}$ uniformly) as follows.

Let $\mathsf{IO} = (\mathsf{Obf}, \mathsf{Eval})$ be an obfuscation scheme. We design schemes $\mathsf{IO}_k = (\mathsf{Obf}_k, \mathsf{Eval}_k)$ which do not call the random oracle as follows. $\mathsf{Obf}_k^{\mathsf{Z}}(C)$, given a circuit C, operates as follows.

- Run $\mathsf{Obf}^{(\cdot),\mathsf{Z}}(C)$ while simulating the random oracle F (i.e., the first oracle) using fresh randomness, and keep the simulated answers in a list L. Obtain an obfuscation D.
- Run D over k random inputs $x_1, \ldots, x_n$ using $\mathsf{Eval}^{(\cdot),\mathsf{Z}}(D, x_i)$ while *simulating* all the answers to the oracle F using fresh randomness and keeping them consistent with the previous simulations stored in the list L. Let L be the final list of such oracle query-answer simulations.
- Output (D, L) as the obfuscation.

To evaluate the "circuit" $D' = (D, L)$ on a (random) input x, $\mathsf{Eval}_k^{\mathsf{Z}}(D', x)$ operates as follows. Run $\mathsf{Eval}^{(\cdot),\mathsf{Z}}(D, x)$, and if during the execution there was a need to simulate an oracle answer to F: first look up L. If the answer was not there, toss and use fresh random coins (and keep them for consistent future answers).

The same analysis as that of [CKP15] can be used to show that when $k = \omega(n)$, where n is the query complexity of the obfuscation algorithm, with probability $1 - \mathcal{O}(n/k) = 1 - o(1)$ we get a *perfect* simulation of the random oracle that is *consistent* across the obfuscation and evaluation steps on a random point x. This is because, for any fixed query to F that is used during the obfuscation step, the probability of *not* asking this query in the first k iterations on $x_1, \ldots, x_k$ and then suddenly asking it during the final actual evaluation is at most $1/k$. Now by a union bound the probability of this event happening to *some* query to F asked during the obfuscation is at most $\mathcal{O}(n/k)$. This means that the correctness of the obfuscated circuit on a random input can only be lost on at most an $o(1)$ fraction of the inputs. This means that the scheme is still an approximate IO scheme. The new scheme is also as secure since the extra information revealed in the new obfuscation algorithm is efficiently simulatable in the random-oracle model by running the given code D on k random inputs. As is clear from this argument, all the steps hold relative to the available oracle Z. □

Theorem 5.7 (Black-box uselessness for aIO)**.** *Let $\mathcal{P}$ be any of the following primitives: witness encryption, predicate encryption, fully homomorphic encryption, or Boolean functional encryption. Then $\mathcal{P}$ is fully black-box useless for approximately correct IO.*

Proof sketch. Here we sketch why the proofs of [GMM17a, GMM17c] extend to the setting of black-box uselessness just like how the proof of Theorem 5.6 could be obtained from the similar proof of [CKP15].

As it was explained in the proof of Theorem 5.6, to get rid of the random oracle (which gives us OWFs), the compiler of [CKP15] changes the obfuscation mechanism by adding an extra string L attached to the obfuscated circuit D with several properties:

- L was efficiently computed by the obfuscation algorithm.
- Revealing L does not hurt security of aIO, because one could obtain it in the ideal world with the oracle that implements the one-way function (i.e.., the random oracle) by running the obfuscated code D on random inputs quite a few times.
- Having L is enough to allow the evaluator to do an (approximately) correct simulation of the (in this case random) oracle that is *consistent* with the simulation of the oracle done during the obfuscation phase (that generated D).

Our key observation was that the proof achieving above relativizes to any fixed oracle Z. When it comes to more complicated primitives such as witness encryption and predicate encryption, as needed by Theorem 5.7, the proofs of [GMM17a, GMM17c] follow the same paradigm described above for the random oracle, with the only difference that coming up with a useful information L with all three properties above is much more challenging. However, when it comes to whether the proof still holds relative to a fixed oracle Z, there is no difference. □

Monolithic uselessness. The black-box separations proved in [GMM17a, GMM17c] for IO and aIO from the primitives listed in Theorem 5.7 hold in a *stronger* model of separations known as the *monolithic* framework, in which the input circuits to these primitives are allowed to have oracle gates to the *primitive itself*. For example, in a functional encryption, one has to issue keys SK_C for a circuit C such that SK_C allows one to compute $C(x)$ from any ciphertext that encrypts x. A monolithic extension of functional encryption allows C to have oracle gates to the subroutines of the functional encryption oracle itself. Just like how [GMM17a, GMM17c] proved their results for the *monolithic* extensions of these primitives, our black-box uselessness of Theorem 5.7 also holds for these monolithic extensions. This suggests developing a theory of *monolithic* uselessness, in which one can allow cross plantations of oracle calls in circuits that are given as input to *both* primitive $\mathcal{P}$ (that we want to prove to be useless) and the auxiliary oracle Z, when these primitives have the properties that allows them to be monolithically extended.[12] We leave the exploration of such lines of work, as an extension of black-box uselessness, for future work.

6 Towards Black-Box Helpfulness of OWFs for Collision-Resistant Hash Functions

In this section, we explore a complementary aspect of black-box uselessness, namely that of black-box *helpfulness*. A primitive $\mathcal{Q}$ is BB helpful for another primitive $\mathcal{P}$ if $\mathcal{Q}$ is not black-box useless

[12] We refer to the full version of the work of Garg et al. [GMM17b] for in-depth discussions of what primitives have this property in general, however a simple rule of thumb is that monolithic extension is possible when inputs are circuits and are run (in the completeness and security definitions) just as black-box. In that case, one can talk about planting oracle gates inside those circuits and still obtain a well-defined primitive.

for $\mathcal{P}$ – meaning, if there exists an auxiliary primitive $\mathcal{Z}$ such that $(\mathcal{Q}, \mathcal{Z})$ black-box implies $\mathcal{P}$, yet $\mathcal{Z}$ does not black-box imply $\mathcal{P}$. Of course, any primitive $\mathcal{Q}$ that black-box implies a primitive $\mathcal{P}$ is BB helpful for $\mathcal{P}$. The more interesting setting is the "gray zone" between black-box uselessness and black-box separations, namely, the black-box helpfulness of $\mathcal{Q}$ for $\mathcal{P}$ when $\mathcal{P}$ is black-box separated from $\mathcal{Q}$. We believe the study of black-box helpfulness can unravel a richer landscape compared to the picture conveyed by the (in)existence of black-box reductions. This can be done by identifying primitives which even though are insufficient to construct another in a black-box way, will provably be helpful in future black-box construction from a combination of primitives. For the sake of readability, we maintain a relatively informal discussion in this section.

We start with the trivial observation that this gray zone is not empty. Consider the artificial (joint) primitive *collision-resistant hash function (CRHF) and trapdoor permutation (TDP)*. An implementation of this primitive is any pair (H, F) where H is a CRHF and F a TDP. By [IR89], there is no black-box construction of this primitive from CRHFs (since [IR89] separates random oracles from KA, which in particular implies that there is no fully-BB construction of TDPs from CRHFs). By a result of Fischlin [Fis02], there is no black-box construction of this primitive from a TDP either (since Fischlin shows, in particular, that there is no fully-BB construction of CRHFs from TDPs); yet, both TDPs and CRHFs are trivially black-box helpful for this primitive. More generally, for every pair of primitives $(\mathcal{P}_1, \mathcal{P}_2)$ where each is black-box separated from the other, $\mathcal{P}_1$ and $\mathcal{P}_2$ are black-box separated from their union, yet each is black-box helpful for $(\mathcal{P}_1, \mathcal{P}_2)$. However, such examples are somewhat artificial.

6.1 A Conjecture on the Black-Box Helpfulness of OWFs for CRHFs

The reader might have observed that while we show that several standard techniques in black-box separations (the "compiling out" technique, and techniques based on sampling views consistent with a transcript) can be fully or partially adapted to the black-box uselessness setting, our analyses leaves open several other techniques, the most important of which is perhaps Simon's oracle separation of collision-resistant hash functions from one-way functions [Sim98]. This leaves an important and intriguing open question:

Are one-way functions (fully) black-box useless for collision-resistant hash functions?

At first sight, it is not clear how one would extend Simon's strategy to prove that one-way functions are BBU for CRHFs. In fact, we believe that this is inherent: we conjecture that the answer to the above question is *no*.

Conjecture 6.1. *One-way functions are fully black-box helpful for collision-resistant hash functions.*

If Conjecture 6.1 holds, then it provides a natural (and important) example of a primitive which does not black-box imply another primitive, yet is black-box helpful for it. Perhaps more importantly, it also provides an indication of why OWFs could play a role in future black-box construction of CRHFs from seemingly weaker assumptions.

While we believe that Conjecture 6.1 holds, proving it seems quite challenging. Towards getting a better understanding of its plausibility, we introduce two natural relaxations of the conjecture, and show that each of these natural relaxations would follow from a plausible conjecture regarding Simon's oracle, which might be of independent interest.

6.2 A First Relaxation of Conjecture 6.1: Distributional Black-Box Helpfulness

In this first relaxation, we extend the notion of black-box reductions (which work for all implementations of a primitive) to a notion which we call *distributional* black-box reductions, which are reductions which, informally, work for *almost all* implementations of a primitive. To make this formal, we must first define the notion of distributional primitives:

Definition 6.2. A distributional primitive $D(\mathcal{Q})$ is a pair $(\mathcal{Q}, \mathcal{D}_{\mathcal{Q}})$ where $\mathcal{Q}$ is a primitive, and $\mathcal{D}_{\mathcal{Q}}$ is a distribution over $F_{\mathcal{Q}}$.

The notion of distributional primitives allows to formalize black box reductions which work for almost all implementations of primitives, by saying that the reduction should work for a *measure 1* of implementations of $\mathcal{Q}$ with respect to the distribution $\mathcal{D}_{\mathcal{Q}}$.

Definition 6.3. A (distributional) primitive $D(\mathcal{Q})$ *distributionally* fully black-box implies a primitive $\mathcal{P}$ if there is a pair (P, S) of oracle-aided PPT machines such that for a measure one of implementation $\mathsf{Q} \in F_{\mathcal{Q}}$ (with respect to the distribution $\mathcal{D}_{\mathcal{Q}}$),

- **Implementation reduction**: P^{Q} implements $\mathcal{P}$, that is, $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$.
- **Security reduction**: For any function (adversary) A that $\mathcal{P}$-breaks $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$, it holds that $\mathsf{S}^{\mathsf{Q},A}$ $\mathcal{Q}$-breaks Q.

More generally, when considering a pair $(D(\mathcal{Q}), \mathcal{R})$ where $D(\mathcal{Q})$ is a distributional primitive and $\mathcal{R}$ is a (standard) primitive, we say that $(\mathcal{Q}, D(\mathcal{R}))$ distributionally fully black-box implies a primitive $\mathcal{P}$ if the above definition holds for all implementations of $\mathcal{Q}$, and for a measure one of implementations of $\mathcal{R}$ w.r.t. $\mathcal{D}_{\mathcal{R}}$. Other flavors of almost black-box reductions can be defined similarly, e.g. almost semi or weakly black-box reductions. Equipped with the above definition, we can define *distributional black-box helpfulness*:

Definition 6.4. A primitive $\mathcal{Q}$ is *distributionally (fully) black-box helpful* for a primitive $\mathcal{P}$ if there exists an auxiliary distributional primitive $D(\mathcal{Z})$ such that $(\mathcal{Q}, D(\mathcal{Z}))$ distributionally (fully) black-box implies $\mathcal{P}$, yet $\mathcal{Z}$ alone does not distributionally (fully) black-box imply $\mathcal{P}$.

We now state our relaxed conjecture:

Conjecture 6.5. *One-way functions are distributionally fully black-box helpful for collision-resistant hash functions.*

We now provide some support for Conjecture 6.5, by relating it to another plausible conjecture regarding Simon's oracle; we believe this independent conjecture to be interesting in its own right. It is well-known that for any attacker A, with measure one over the choice of a random permutation RP, A cannot invert RP with probability better than $\mathrm{poly}(\lambda)/2^{\lambda}$ [IR89]. By Simon's result, this still holds even in the presence of a *collision-finder* Coll that samples a random collision for any oracle circuit with RP-gates and (recursively) Coll-gates [Sim98]. These two results immediately extend to a setting where an auxiliary oracle Z that is "independent" of $\mathsf{RP}, \mathsf{Coll}$ is provided. Due to the random nature of RP, it seems plausible that a stronger statement holds, which states that the hardness of inverting *simultaneously* some one-way function and a random permutation should be very hard:

Conjecture 6.6 (Amplification, weak version)**.** *Suppose that* F *is an oracle implementing an* ε*-secure one-way function; that is, no oracle-aided PPT machine can invert* $\mathsf{F}(x)$ *on a random* $x \in \{0,1\}^\lambda$ *with probability better than* $\varepsilon(\lambda)$*. Then for any oracle-aided PPT machine* A *and for a measure one of random permutation* RP*,*

$$\Pr\left[(x_1,x_2) = A^{\mathsf{F},\mathsf{RP}}(\mathsf{P}(x_1),\mathsf{RP}(x_2))\right] \leq \varepsilon(\lambda)\cdot\frac{\mathrm{poly}(\lambda)}{2^\lambda}\ ,$$

where the probability is taken over the choice of $(x_1,x_2) \xleftarrow{\$} (\{0,1\}^\lambda)^2$*, and coins of* A*.*

The above conjecture is known in the case where F is itself a random oracle, but not when it is an arbitrary one-way function. We believe that Conjecture 6.6 is a natural conjecture regarding random permutations, and its study might be of interest beyond the setting of black-box helpfulness. In our context, though, we need an even stronger version, which we still deem plausible, where the random permutation comes with a collision finder Coll:

Conjecture 6.7 (Amplification, strong version)**.** *Suppose that* F *is an oracle implementing an* ε*-secure one-way function. Then for any oracle-aided PPT machine* A *and for a measure one of random permutation* RP *and collision-finders* $\mathsf{Coll}^{\mathsf{RP}}$*,*

$$\Pr\left[(x_1,x_2) = A^{\mathsf{F},\mathsf{RP},\mathsf{Coll}^{\mathsf{RP}}}(\mathsf{P}(x_1),\mathsf{RP}(x_2))\right] \leq \varepsilon(\lambda)\cdot\frac{\mathrm{poly}(\lambda)}{2^\lambda}\ ,$$

where the probability is taken over the choice of (x_1,x_2) *from* $(\{0,1\}^\lambda)^2$*, and the coins of the adversary.*

We now sketch why proving the above conjecture would establish that OWFs are distributionally black-box helpful for collision-resistant hash functions:

Theorem 6.8. *If Conjecture 6.7 holds, then Conjecture 6.5 holds.*

Proof Sketch. Assume that Conjecture 6.7 holds. Let $(\mathsf{RP},\mathsf{Coll}^{\mathsf{RP}})$ be as in Simon. Then, for any one-way function $\mathsf{F} \in F_{\mathsf{OWF}}$, with measure 1 over the choice of $(\mathsf{RP},\mathsf{Coll}^{\mathsf{RP}})$, there exists a pair $(\mathsf{F}_1,\mathsf{F}_2)$ of one-way functions such that for any oracle-aided PPT machine A,

$$\Pr\left[(x_1',x_2') \xleftarrow{\$} A^{\mathsf{F},\mathsf{RP},\mathsf{Coll}^{\mathsf{RP}}}(\mathsf{F}_1(x_1),\mathsf{F}_2(x_2)) : \mathsf{F}_1(x_1') = \mathsf{F}_1(x_1) \wedge \mathsf{F}_2(x_2') = \mathsf{F}_2(x_2)\right] \leq \frac{\mathrm{negl}(\lambda)}{2^\lambda}\ .$$

Indeed, one can simply set F_1 to be F and F_2 to be RP (it follows directly from Simon's analysis [Sim98] that F_2 remains (exponentially) one-way even in the presence of $\mathsf{Coll}^{\mathsf{RP}}$ and any other independent oracle). From there, the inequality follows directly from Conjecture 6.7.

Now, suppose that a pair $(\mathsf{F}_1,\mathsf{F}_2)$ as above exists. By a result of Holmgren and Lombardi [HL18], this pair can be used to construct collision-resistant hash-functions in a black-box way.[13] This implies that there exists an auxiliary primitive $\mathcal{Q}$, namely, all pairs (one-way permutation, collision finder for the OWF) such that for *any* OWF F, there is a measure 1 of implementations Z of $\mathcal{Z}$ such that (F,Z) can be used to construct a CRHF. This, in turn, implies that one-way functions are distributionally black-box helpful for collision-resistant hash functions. □

[13] More precisely, the result of Holmgren and Lombardi holds whenever $\mathsf{F}_1 = \mathsf{F}_2$ and F_1 is injective, but their work also provides a black-box reduction from an arbitrary $(\mathsf{F}_1,\mathsf{F}_2)$ to an injective $(\mathsf{F}_1,\mathsf{F}_2)$ with $\mathsf{F}_1 = \mathsf{F}_2$, albeit with some extra loss. We omit the exact security strength that is needed from F_1 from this high-level sketch proof.

Note that the measure 1 of implementations $\mathsf{Z} \in F_{\mathcal{Z}}$ needs not be the same for every F, which is the reason why we cannot extract from this proof a universal primitive for which all implementations would work.

The notion of distributional black-box reductions is not standard. In the next section, we observe that our conjecture about Simon's oracle suffices to prove another relaxation of Conjecture 6.1, where we consider *class reductions* instead of black-box reductions. Unlike distributional black-box reductions, class reductions are a natural notion of reduction which has been used in several recent works. For class reductions, we show that we *can* extract a universal primitive, by restricting our attention solely to *efficient* one-way functions (i.e. computable by a PPT Turing machine). We elaborate in the next section.

6.3 A Second Relaxation of Conjecture 6.1: Class Helpfulness

There are several examples in cryptography and complexity theory of non-black-box reductions which are very similar to black-box reductions, except that they are only guaranteed to succeed when the oracle implementing the primitive belongs to a class of *efficient* implementations. Such reductions have been recently formalized in [Sha20] as *class reductions*. We observe that if we restrict our attention to class reductions instead of black-box reductions, then we can actually extract a universal primitive in our above argument.

Definition 6.9 ((Full) class reduction [Sha20])**.** A (full) *class* reduction of a primitive $\mathcal{P}$ to another primitive $\mathcal{Q}$ with respect to a class $C \subseteq F_{\mathcal{Q}}$ is a pair (P, S) of oracle-aided PPT machine such that for any implementation $\mathsf{Q} \in C$ the following two conditions hold.

- **Implementation reduction**: P^{Q} implements $\mathcal{P}$, that is, $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$.
- **Security reduction**: For any function (adversary) A that $\mathcal{P}$-breaks $\mathsf{P}^{\mathsf{Q}} \in F_{\mathcal{P}}$, i.e., $(\mathsf{P}^{\mathsf{Q}}, A) \in R_{\mathcal{P}}$), it holds that $\mathsf{S}^{\mathsf{Q},A}$ $\mathcal{Q}$-breaks Q, i.e., $(\mathsf{Q}, \mathsf{S}^{\mathsf{Q},A}) \in R_{\mathcal{Q}}$.

Unless specified otherwise, we consider C to be the class of all implementations of $\mathcal{Q}$ by PPT Turing machines (i.e., the class of efficient implementations of $\mathcal{Q}$).

Examples of class reductions include several works on worst-case to average-case reduction within **NP** [Gut06, Ats06, GSTS07, GTS07, Hir18]. In recent work [Sha20], Shaltiel proves that class reductions cannot be used to improve over Yao's XOR lemma. In all these works, the class considered is that of all efficient implementations of the primitive.

Definition 6.10 (Class helpfulness)**.** A cryptographic primitive $\mathcal{Q}$ is fully *class-helpful* with respect to a class $C \subseteq F_{\mathcal{Q}}$ for constructing a primitive $\mathcal{P}$ if there exists an auxiliary primitive $\mathcal{Z}$ such that there exists a (full) class reduction from $\mathcal{P}$ to $(\mathcal{Q}, \mathcal{Z})$, yet there is no fully black-box reduction from $\mathcal{P}$ to $\mathcal{Z}$.

Equipped with the above definitions, we can now formulate our second relaxation of Conjecture 6.1.

Conjecture 6.11. *One-way functions are fully class-helpful for collision-resistant hash functions, for the class of all efficient one-way functions.*

Theorem 6.12. *If Conjecture 6.7 holds, then Conjecture 6.11 holds.*

Proof sketch. The beginning of the proof proceeds as in the proof of Theorem 6.8. Recall that this proof gives us a distribution over implementations of a primitive $\mathcal{Z}$ such that for *any* OWF F, there is a measure 1 of implementations Z of $\mathcal{Z}$ such that (F, Z) can be used to construct a CRHF. Now, we restrict our attention to the class C of efficient implementations of one-way functions. Since this is a countable set, we can apply the Borel–Cantelli lemma to show that there must therefore exist a *single* implementation Z of $\mathcal{Z}$ that simultaneously works for all $\mathsf{F} \in C$ (this is the same argument as used by Impagliazzo and Rudich to extract a one-way function secure against all efficient adversaries from a distribution over random oracles [IR89]). Therefore, there exists an implementation Z of $\mathcal{Z}$ such that for any $\mathsf{F} \in C$, (F, Z) can be used to construct a CRHF. Define the primitive $\mathcal{Z}'$ to be the primitive $\mathcal{Z}$ restricted to this single implementation Z (i.e., $F_{\mathcal{Z}'} := \{\mathsf{Z}\}$). Then there is a class reduction from $\mathcal{P}$ to $(\mathcal{F}, \mathcal{Z}')$, for the class $C \subseteq F_{\mathcal{F}}$ of efficient one-way functions; yet, by Simon's separation, $\mathcal{Z}'$ does not black-box imply collision-resistant hash functions. This concludes the proof. □

6.4 A Black-Box Helpful Idealized Primitive for CRHFs

We conclude this section by observing that, based on a communication complexity conjecture, the recent work of [BFM18] describes combiners that can turn any pair of *backdoored* random oracles into a collision-resistant hash function. Their backdoored random oracle model allows for fully adaptive leakage on the random oracle, and can in particular can implement Simon's oracle. Therefore, although a single backdoored random oracle is black-box separated from CRHFs, two independent instances of such oracles suffice to build a CRHF (under a communication complexity conjecture). This provides a relatively natural example of an idealized primitive which does not black-box imply CRHFs, yet is BB helpful for CRHFs.

References

[AKW18] Shashank Agrawal, Venkata Koppula, and Brent Waters. Impossibility of simulation secure functional encryption even with random oracles. In *Theory of Cryptography Conference*, pages 659–688. Springer, 2018. 4, 21, 22

[AS15] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 191–209, Berkeley, CA, USA, October 17–20, 2015. IEEE Computer Society Press. 1

[Ats06] Albert Atserias. Distinguishing sat from polynomial-size circuits, through black-box queries. In *21st Annual IEEE Conference on Computational Complexity (CCC'06)*, pages 8–pp. IEEE, 2006. 30

[BBF13] Paul Baecher, Christina Brzuska, and Marc Fischlin. Notions of black-box reductions, revisited. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 296–315, Bengalore, India, December 1–5, 2013. Springer, Heidelberg, Germany. 1

[BBF16] Zvika Brakerski, Christina Brzuska, and Nils Fleischhacker. On statistically secure obfuscation with approximate correctness. In Matthew Robshaw and Jonathan Katz,

editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 551–578, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. 4

[BFM18] Balthazar Bauer, Pooya Farshim, and Sogol Mazaheri. Combiners for backdoored random oracles. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 272–302, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 6, 31

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany. 2

[BHT14] Itay Berman, Iftach Haitner, and Aris Tentes. Coin flipping of *any* constant bias implies one-way functions. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 398–407, New York, NY, USA, May 31 – June 3, 2014. ACM Press. 18

[BKSY11] Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 559–578, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. 1, 3, 6, 13, 14, 16, 17, 18

[BLP17] Nir Bitansky, Huijia Lin, and Omer Paneth. On removing graded encodings from functional encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–29. Springer, 2017. 4, 21, 22

[BM09] Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle puzzles are optimal - an $O(n^2)$-query attack on any key exchange from a random oracle. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 374–390, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Heidelberg, Germany. 1, 6, 17, 18

[BM17] Boaz Barak and Mohammad Mahmoody. Merkle's key agreement protocol is optimal: An $O(n^2)$ attack on any key agreement from random oracles. *Journal of Cryptology*, 30(3):699–734, July 2017. 17

[CKP15] Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On obfuscation with random oracles. 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 456–467, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany. 4, 5, 21, 22, 24, 25, 26

[CLMP13] Kai-Min Chung, Huijia Lin, Mohammad Mahmoody, and Rafael Pass. On the power of nonuniformity in proofs of security. In Robert D. Kleinberg, editor, *ITCS 2013:*

4th Innovations in Theoretical Computer Science, pages 389–400, Berkeley, CA, USA, January 9–12, 2013. Association for Computing Machinery. 8

[Fis02] Marc Fischlin. On the impossibility of constructing non-interactive statistically-secret protocols from any trapdoor one-way function. In Bart Preneel, editor, *Topics in Cryptology – CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 79–95, San Jose, CA, USA, February 18–22, 2002. Springer, Heidelberg, Germany. 27

[GGKT05] Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM journal on Computing*, 35(1):217–246, 2005. 1, 4, 5, 10, 21, 22, 24

[GHMM18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ameer Mohammed. Limits on the power of garbling techniques for public-key encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 335–364, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 4, 21, 22

[GMM17a] Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. Lower bounds on obfuscation from all-or-nothing encryption primitives. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 661–695, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 1, 4, 5, 21, 22, 24, 25, 26

[GMM17b] Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. Lower bounds on obfuscation from all-or-nothing encryption primitives. In *Draft of the full version*, 2017. http://www.cs.virginia.edu/~mohammad/files/papers/IO-all-or-nothing.pdf. 5, 26

[GMM17c] Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. When does functional encryption imply obfuscation? In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 82–115, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany. 4, 5, 21, 22, 24, 25, 26

[GSTS07] Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. If np languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007. 30

[GTS07] Dan Gutfreund and Amnon Ta-Shma. Worst-case to average-case reductions revisited. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 569–583. Springer, 2007. 30

[Gut06] Dan Gutfreund. Worst-case vs. algorithmic average-case complexity in the polynomial-time hierarchy. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 386–397. Springer, 2006. 30

[HHRS07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *48th Annual Symposium on Foundations of Computer Science*, pages 669–679, Providence, RI, USA, October 20–23, 2007. IEEE Computer Society Press. 1

[Hir18] Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In Mikkel Thorup, editor, *59th Annual Symposium on Foundations of Computer Science*, pages 247–258, Paris, France, October 7–9, 2018. IEEE Computer Society Press. 30

[HL18] Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In Mikkel Thorup, editor, *59th Annual Symposium on Foundations of Computer Science*, pages 850–858, Paris, France, October 7–9, 2018. IEEE Computer Society Press. 5, 29

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science*, pages 230–235, Research Triangle Park, NC, USA, October 30 – November 1, 1989. IEEE Computer Society Press. 18, 19

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *21st Annual ACM Symposium on Theory of Computing*, pages 44–61, Seattle, WA, USA, May 15–17, 1989. ACM Press. 1, 3, 6, 13, 17, 18, 25, 27, 28, 31

[KSS00] Jeff Kahn, Michael Saks, and Cliff Smyth. A dual version of reimer's inequality and a proof of rudich's conjecture. In *Proceedings 15th Annual IEEE Conference on Computational Complexity*, pages 98–103. IEEE, 2000. 23

[Mer78] Ralph C Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978. 19

[MM16] Mohammad Mahmoody and Ameer Mohammed. On the power of hierarchical identity-based encryption. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 243–272, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany. 2

[MMN+16a] Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. Lower bounds on assumptions behind indistinguishability obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 49–66, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany. 4

[MMN+16b] Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. A note on black-box separations for indistinguishability obfuscation. Cryptology ePrint Archive, Report 2016/316, 2016. http://eprint.iacr.org/2016/316. 4

[MMV11] Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Time-lock puzzles in the random oracle model. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 39–50, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Heidelberg, Germany. 6, 18

[MW20] Hemanta K. Maji and Mingyuan Wang. Black-box use of one-way functions is useless for optimal fair coin-tossing. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 593–617, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany. 2, 5

[RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany. 1, 7

[Rud88] Steven Rudich. *Limits on the provable consequences of one-way functions*. University of California at Berkeley, 1988. 23

[Sha20] Ronen Shaltiel. Is it possible to improve yao's xor lemma using reductions that exploit the efficiency of their oracle? In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020. 30

[Sim98] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345, Espoo, Finland, May 31 – June 4, 1998. Springer, Heidelberg, Germany. 1, 5, 27, 28, 29

[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. 1, 2

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press. 3, 15