Limits on the Power of Indistinguishability Obfuscation and Functional Encryption

Gilad Asharov*   Gil Segev*

Abstract

Recent breakthroughs in cryptography have positioned indistinguishability obfuscation as a "central hub" for almost all known cryptographic tasks, and as an extremely powerful building block for new cryptographic tasks resolving long-standing and foundational open problems. However, constructions based on indistinguishability obfuscation almost always rely on non-black-box techniques, and thus the extent to which it can be used as a building block has been completely unexplored so far.

We present a framework for proving meaningful negative results on the power of indistinguishability obfuscation. By considering indistinguishability obfuscation for *oracle-aided* circuits, we capture the common techniques that have been used so far in constructions based on indistinguishability obfuscation. These include, in particular, *non-black-box* techniques such as the punctured programming approach of Sahai and Waters (STOC '14) and its variants, as well as sub-exponential security assumptions.

Within our framework we prove the first negative results on the power of indistinguishability obfuscation and of the tightly related notion of functional encryption. Our results are as follows:

- There is no fully black-box construction of a collision-resistant function family from an indistinguishability obfuscator for oracle-aided circuits.
- There is no fully black-box construction of a key-agreement protocol with perfect completeness from a private-key functional encryption scheme for oracle-aided circuits.

Specifically, we prove that any such potential constructions must suffer from an exponential security loss, and thus our results cannot be circumvented using sub-exponential security assumptions. Our framework captures constructions that may rely on a wide variety of primitives in a non-black-box manner (e.g., obfuscating or generating a functional key for a function that uses the evaluation circuit of a puncturable pseudorandom function), and we only assume that the underlying indistinguishability obfuscator or functional encryption scheme themselves are used in a black-box manner.

*School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem 91904, Israel. Email: {asharov,segev}@cs.huji.ac.il. Supported by the European Union's Seventh Framework Programme (FP7) via a Marie Curie Career Integration Grant, by the Israel Science Foundation (Grant No. 483/13), and by the Israeli Centers of Research Excellence (I-CORE) Program (Center No. 4/11).

Contents

1 Introduction 1

1.1 Our Contributions . . . 2

1.2 Related Work . . . 4

1.3 Overview of Our Approach . . . 5

1.4 Paper Organization . . . 6

2 Preliminaries 6

2.1 Indistinguishability Obfuscation for Oracle-Aided Circuits . . . 7

2.2 Private-Key Functional Encryption for Oracle-Aided Circuits . . . 7

2.3 Oracle-Aided Collision-Resistant Function Families . . . 8

2.4 Oracle-Aided Key-Agreement Protocols . . . 9

3 Limits on the Power of Indistinguishability Obfuscation 9

3.1 The Class of Reductions . . . 10

3.2 Proof Overview and the Oracle Γ . . . 12

3.3 Breaking Any Collision-Resistant Function Family Relative to Γ . . . 15

3.4 $i\mathcal{O}$ is an Indistinguishability Obfuscator Relative to Γ . . . 15

3.4.1 From Distinguishing to Hitting . . . 18

3.4.2 Avoiding Hits . . . 19

3.4.3 From Hitting to Compressing . . . 24

3.5 f is a One-Way Permutation Relative to Γ . . . 28

3.5.1 Avoiding Hits . . . 29

3.5.2 From Inverting to Compressing . . . 32

3.5.3 Extension to Trapdoor Permutations . . . 35

4 Limits on the Power of Private-Key Functional Encryption 36

4.1 The Class of Reductions . . . 36

4.2 Proof Overview and the Oracle Ψ . . . 37

4.3 f is a One-Way Permutation Relative to Ψ . . . 40

4.4 Π is a Functional Encryption Scheme Relative to Ψ . . . 40

4.4.1 Simulating the Decryption Oracle . . . 41

4.4.2 From Distinguishing to Hitting . . . 44

4.4.3 Concluding the Proof . . . 46

4.5 Breaking Any Perfectly-Complete Bit-Agreement Protocol Relative to Ψ . . . 46

4.5.1 Warm-up: Breaking Perfectly-Complete Bit Agreement Relative to f . . . 47

4.5.2 Breaking Perfectly-Complete Bit Agreement Relative to Ψ . . . 48

4.6 Extending the Result to Indistinguishability Obfuscation . . . 53

Acknowledgments 56

References 56

1 Introduction

The study of program obfuscation within the cryptography community was initiated by Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang [BGI$^+$01, BGI$^+$12] with the goal of understanding the extent to which programs can be made "unintelligible" while preserving their functionality. Barak et al. introduced two notions of obfuscation, virtual black-box obfuscation and indistinguishability obfuscation, which offer two such extents: Virtual black-box obfuscation asks that an obfuscated program reveals no more information than a black box implementing the program, and indistinguishability obfuscation asks, somewhat more modestly, that obfuscations of any two functionally-equivalent programs be computationally indistinguishable.

Barak et al. showed that general-purpose virtual black-box obfuscation is impossible to achieve, and left open the problem of whether or not indistinguishability obfuscation exists[1]. This has rooted the cryptography community with a somewhat pessimistic view, as it was not at all clear whether indistinguishability obfuscation (if even exists) is nearly as useful as virtual black-box obfuscation. In a recent breakthrough result, Garg, Gentry, Halevi, Raykova, Sahai, and Waters [GGH$^+$13] proposed the first candidate construction of a general-purpose indistinguishability obfuscator, and showed how to apply indistinguishability obfuscation for constructing the first general-purpose functional-encryption scheme.

The power of indistinguishability obfuscation. The initial breakthrough by Garg et al. [GGH$^+$13] has motivated Sahai and Waters [SW14] to carry out a systematic study of the applicability of indistinguishability obfuscation. Despite the somewhat modest flavor of obfuscation that is offered by indistinguishability obfuscation, Sahai and Waters presented strikingly-surprising constructions, positioning indistinguishability obfuscation as a "central hub" in cryptography. Specifically, relying on indistinguishability obfuscation Sahai and Waters resolved the long-standing foundational open problem of constructing a deniable encryption scheme [CDN$^+$97], as well as constructed a variety of core cryptographic objects such as public-key encryption, short "hash-and-sign" signatures, CCA-secure public-key encryption, non-interactive zero-knowledge proofs, injective trapdoor functions, and oblivious transfer.

Following the work of Sahai and Waters, extensive research has been recently devoted to the applicability of indistinguishability obfuscation, positioning it as a central hub for almost all known cryptographic tasks, and as an extremely powerful building block for new cryptographic tasks resolving long-standing and foundational open problems. Applications of indistinguishability obfuscations range from fundamental building blocks such as one-way functions [KMN$^+$14] and trapdoor permutations [BPW15], to more complex tasks such as full-domain hash without random oracles [HSW14], multiparty key exchange [BZ14], efficient traitor tracing [BZ14], multi-input functional encryption [GGG$^+$14, AJ15], functional encryption for randomized functionalities [GJK$^+$15], adaptively-secure multiparty computation [GGH$^+$14a, CGP15, DKR15, GP15], communication-efficient secure computation [HW15], adaptively-secure functional encryption [Wat15], polynomially-many hardcore bits for any one-way function [BST14], ZAPs and non-interactive witness-indistinguishable proofs [BP15], constant-round zero-knowledge proofs [CLP14], fully-homomorphic encryption [CLT$^+$15], and even cryptographic hardness for the complexity class PPAD [BPR15].

Despite the extensive recent research on indistinguishability obfuscation, the following fundamental question has remained completely open:

Is there a natural task that cannot be solved using indistinguishability obfuscation?

[1] Nevertheless, there are examples of function families that can be obfuscated in a virtual black-box manner under strong assumptions (e.g., [Can97, Wee05]).

The power of functional encryption. Functional encryption [SW08, BSW11] allows tremendous flexibility when accessing encrypted data: it supports restricted decryption keys that allow users to learn specific functions of the encrypted data and nothing else. The notion of functional encryption is tightly related to that of indistinguishability obfuscation. On one hand, Garg et al. [GGH$^+$13] showed that indistinguishability obfuscation implies general-purpose public-key functional encryption. On the other hand, Ananth and Jain [AJ15] and Bitansky and Vaikuntanathan [BV15] showed that general-purpose public-key functional encryption with sub-exponential security (and some additional requirements) implies general-purpose indistinguishability obfuscation.

Recent research has explored the private-key variant of functional encryption [AAB$^+$13, ABS$^+$15, AJ15, BS15, BKS15, KSY15], showing that it is a very useful building block that can essentially be used instead of indistinguishability obfuscation in various cases based on the notion of function privacy [BS15]. Although private-key functional encryption seems significantly weaker than its public-key variant, constructions of private-key functional encryption schemes are currently known based only on public-key functional encryption [GGH$^+$13, GGH$^+$14c, Wat15]. At the same time, however, we currently do not know how to construct any public-key primitive based on private-key functional encryption, and this raises the following fundamental question[2]:

Does general-purpose private-key functional encryption imply public-key cryptography?

The challenge of dealing with non-black-box constructions. The above-stated open questions clearly call for a study of the limitations of using indistinguishability obfuscation and functional encryption as building blocks in cryptographic constructions[3]. In general, one cannot argue about limitations of cryptographic constructions without placing any restrictions (as long as we believe that the cryptographic primitives under consideration exist – since a construction may simply ignore its building blocks). In an effort to capture what is meant by a "natural" construction of one primitive from another, Impagliazzo and Rudich [IR89] introduced the framework of black-box constructions, which has been extensively and successfully used over the years for studying the limitations of cryptographic constructions.[4]

The main challenge in the setting of indistinguishability obfuscation and functional encryption is that constructions that are based on either one of these two primitives almost always have a non-black-box ingredient. Specifically, such constructions obfuscate or generate a functional key for a function that uses the *description* of a cryptographic primitive (e.g., the evaluation circuit of a puncturable pseudorandom function or of a pseudorandom generator). Thus, any study of the limitations of indistinguishability obfuscation and functional encryption must face the challenge of dealing with such non-black-box ingredients.

1.1 Our Contributions

In this work we prove the first limitations on the power of indistinguishability obfuscation and functional encryption as building blocks in cryptographic constructions. Our results are obtained within a subtle framework capturing constructions that may rely on a wide variety of primitives in a non-black-box manner (e.g., obfuscating or generating a functional key for a function that uses the

[2] We note that for other forms of encryption, such as homomorphic encryption or re-randomizable encryption, it is known that any private-key scheme implies a public-key one [Rot11, CLT$^+$15].

[3] Whereas various efforts have already been devoted to studying the *existence* of indistinguishability obfuscation (see, for example, [BCC$^+$14] and the references therein), our goal is to explore the *limitations* of indistinguishability obfuscation and functional encryption.

[4] Informally, a black-box construction is a construction that "ignores the internal structure" of its underlying building blocks.

evaluation circuit of a puncturable pseudorandom function), and we only assume that the underlying indistinguishability obfuscator or functional encryption scheme themselves are used in a black-box manner. Specifically, we observe that by considering indistinguishability obfuscation and functional encryption for *oracle-aided* circuits, we can capture the common techniques that have been used so far in constructions based on indistinguishability obfuscation. These include, in particular, *non-black-box* techniques such as the punctured programming approach of Sahai and Waters [SW14] and its variants, as well as sub-exponential security assumptions.

Within our framework we begin by exploring the limitations of indistinguishability obfuscation and prove the following theorem:

Theorem (informal) 1.1. *There is no fully black-box construction of a collision-resistant function family from a general-purpose indistinguishability obfuscator and a one-way permutation (and even a trapdoor permutation).*

Specifically, we prove that any such potential construction must suffer from an exponential security loss, and thus sub-exponential security assumptions (e.g., [AJ15, BPR15, CLT$^+$15, BPW15, BV15]) are not useful for circumventing our negative result.

This shows that not only there exists a cryptographic task that cannot be solved using indistinguishability obfuscation using the common techniques, but in fact identifies collision-resistant hashing (one of the most fundamental primitives) as a specific such task. In turn, this implies a similar result for any primitive that is known to imply collision-resistant hashing in a fully black-box manner, such as homomorphic encryption, homomorphic commitments, two-message private information retrieval, and more (see [IKO05]).[5]

We then explore the limitations of private-key functional encryption and prove the following theorem:

Theorem (informal) 1.2. *There is no fully black-box construction of a key-agreement protocol with perfect completeness from a general-purpose private-key functional encryption scheme and a one-way permutation.*

As with Theorem 1.1, we prove that any such potential construction must suffer from an exponential security loss. Moreover, Theorem 1.2 holds even when the underlying functional encryption scheme is "compact" [AJ15, BV15] in the sense that the efficiency of its encryption algorithm depends only on the security parameter and on length of the plaintext (in particular, it is independent of the complexity of the function family supported by the scheme). This result positions private-key functional encryption as a private-key primitive unless additional non-black-box techniques are used.

Finally, we show that Theorem 1.2 can be extended for separating indistinguishability obfuscation for oracle-aided circuits from private-key functional encryption for oracle-aided circuits (this does not follow immediately from Theorem 1.2 since indistinguishability obfuscation and one-way permutations are not known to imply key agreement in a black-box manner). We prove the following theorem:

Theorem (informal) 1.3. *There is no fully black-box construction of an indistinguishability obfuscator for oracle-aided circuits from a private-key functional encryption scheme for oracle-aided circuits.*

[5] A construction of a (fully-)homomorphic encryption scheme based on indistinguishability obfuscation with sub-exponential security was recently given by Canetti et al. [CLT$^+$15]. Their construction assumes, in addition, re-randomizable encryption – which is currently known to exists based only on assumptions that already imply collision-resistant hashing.

As with Theorems 1.1 and 1.2, we prove that any such potential construction must suffer from an exponential security loss. An impossibility result of a somewhat similar flavor was proved by Goldwasser and Rothblum [GR14] who showed that in the *programmable* random-oracle model there exists a class of oracle-aided circuits for which there is no indistinguishability obfuscator[6]. Our result is obtained by considering a more structured oracle relative to which we prove that there exists a private-key functional encryption scheme for polynomial-size oracle-aided circuits, but there is no indistinguishability obfuscator for polynomial-size oracle-aided circuits. Our result can be viewed as strengthening that of Goldwasser and Rothblum, both by avoiding any flavor of *oracle programmability*, and by considering a seemingly stronger building block (specifically, a private-key functional encryption scheme for oracle-aided circuits in our case vs. a random function in their case).

Finally, we note that unlike Theorems 1.1 and 1.2 that show separation results in the *standard model*, Theorem 1.3 does not necessarily imply a separation result in the standard model (we refer the reader to [GR14] for a discussion of the possible implications of results of this flavor). Specifically, Theorem 1.3 does not necessarily imply a separation in the standard model, since it may be that there exists an indistinguishability obfuscator for circuits, but there does not exist such an obfuscator for *oracle-aided* circuits. Nevertheless, this provides substantial evidence that *private-key* functional encryption is somewhat unlikely to imply indistinguishability obfuscation using the common techniques.

1.2 Related Work

Our approach in this paper is inspired by a combination of various approaches and ideas that were developed in previous work. Our framework for capturing certain non-black-box techniques in constructions that are based on indistinguishability obfuscation and functional encryption is inspired by the work of Brakerski, Katz, Yerukhimovich and Segev [BKS$^+$11]. Their work addressed the question of whether non-black-box techniques that are originated from zero-knowledge proofs can be used for circumventing known impossibility results for black-box constructions. Although our work shares the same theme of capturing non-black-box techniques in a black-box manner, we capture a different class of non-black-box techniques, which raises many additional difficulties when compared to their work.

Our impossibility result for collision-resistant hashing generalizes the approach of Haitner et al. [HHR$^+$15], who in turn generalized the ideas of Simon [Sim98], Gennaro et al. [GGK$^+$05], and Wee [Wee07]. Specifically, we rely on Simon's collision-finding oracle, but since we also use additional (inefficient) oracles for implementing indistinguishability obfuscation (and these oracles can interact with Simon's oracle), we again deal with many additional difficulties when compared to previous work.

Our impossibility result for key-agreement protocols is inspired by ideas that were developed in the early work of Impagliazzo and Rudich [IR89], in its improvement by Barak and Mahmoody-Ghidary [BM09], and in the work of Brakerski, Katz, Yerukhimovich and Segev [BKS$^+$11] who focused on the case of perfectly-complete protocols. We refer the reader to the work of Reingold, Trevisan and Vadhan [RTV04] and to various recent impossibility results (see, for example, [BM07, Wee07, HHS08, BM09, DLM$^+$11, MP12, CLM$^+$13, DMM14, MMP14] and the references therein) for an overview of black-box reductions and the known impossibility results.

[6] A different impossibility result for obfuscation considered the significantly stronger notion of virtual black-box obfuscation – see [LPS04, CTP15] who extended the work of Barak et al. [BGI$^+$12] to the random-oracle model.

1.3 Overview of Our Approach

In this section we provide a high-level overview of our approach. First, we describe our framework that enables us to prove meaningful impossibility results for constructions that are based on indistinguishability obfuscation and functional encryption. Then, within our framework, we describe the main ideas underlying our impossibility results.

Capturing non-black-box constructions via oracle-aided circuits. The fact that constructions that are based on indistinguishability obfuscation or functional encryption are almost always *non-black-box* makes it extremely challenging to prove any impossibility results. For example, a typical such construction would apply the obfuscator or the key-generation algorithm to a function that uses the evaluation circuit of a pseudorandom generator or a pseudorandom function, and this requires *specific implementations* of its underlying building blocks.

However, most of the non-black-box techniques that are used on such constructions have essentially the same flavor: The obfuscator or the key-generation algorithm are applied to functions that can be constructed in a fully black-box manner from a low-level primitive, such as a one-way function or a trapdoor permutation. In particular, the vast majority of constructions rely on the obfuscator or the functional encryption scheme themselves in a black-box manner. Thus, when considering the stronger primitives of indistinguishability obfuscation and functional encryption for *oracle-aided* circuits (see Sections 2.1 and 2.2), these non-black-box techniques in fact directly translate into black-box ones. These include, in particular, non-black-box techniques such as the punctured programming approach of Sahai and Waters [SW14] and its variants (as well as sub-exponential security assumptions – which are already captured by most frameworks for black-box impossibility results).

Example: The Sahai-Waters punctured programming approach. Consider, for example, the construction of a public-key encryption scheme from a one-way function and a general-purpose indistinguishability obfuscator by Sahai and Waters [SW14]. Their construction relies on the underlying one-way function in a non-black-box manner. However, relative to an oracle that allows the existence of a one-way function f and indistinguishability obfuscation $i\mathcal{O}$ for *oracle-aided* circuits C^f, it is in fact a fully black-box construction. Specifically, Sahai and Waters use the underlying indistinguishability obfuscator for obfuscating a circuit that invokes a puncturable pseudorandom function and a pseudorandom generator as sub-routines. Given that puncturable pseudorandom functions and pseudorandom generators can be based on any one-way function in a fully black-box manner, from our perspective such a circuit is a polynomial-size oracle-aided circuit C^f – which can be obfuscated using $i\mathcal{O}$ (we refer to reader to Section 4.6 for more details).

This reasoning naturally extends to various variants of the punctured programming approach by Sahai and Waters [SW14]. However, it does not extend to constructions that may rely on the obfuscator or the functional encryption scheme themselves in a non-black-box manner (e.g., [BP15])[7], or constructions that may rely on zero-knowledge techniques and require using NP reductions[8].

Our separation results. Within our framework we obtain our results via two oracle separations on which we now elaborate. For ruling out a fully black-box construction of a primitive X from a primitive Y, it suffices to construct an oracle relative to which there exists an implementation

[7] With the exception of obfuscating a function that may invoke an indistinguishability obfuscator in a black-box manner. This is captured by our approach – see Section 3.1.

[8] Such techniques are captured by the work of Brakerski et al. [BKS+11], and we leave it as an intriguing open problem to see whether the two approaches for capturing non-black-box techniques can be unified.

of Y but any implementation of X can be efficiently broken by an oracle-aided algorithm (see, for example, [IR89, RTV04]).

We prove Theorem 1.1 by presenting an oracle Γ relative to which the following three properties hold: (1) there exists an exponentially-secure one-way permutation (and even a trapdoor permutation family), (2) there exists an exponentially-secure general-purpose obfuscator for oracle-aided circuits, and (3) there does not exist a collision-resistant function family. Our oracle Γ is quite structured and consists of three main ingredients. The first ingredient is a random permutation f that will serve as a one-way permutation. The second ingredient is a pair $(\mathcal{O}, \mathsf{Eval}^{f,\mathcal{O}})$ of functions, where $\mathcal{O}$ is a random permutation that will serve as an obfuscator (obfuscating a circuit C is done by computing a "handle" $\mathcal{O}(C, r)$ for a randomly-chosen string r), and Eval is a function that enables evaluations of obfuscated circuits (Eval has access to both f and $\mathcal{O}$): Given a handle $\mathcal{O}(C, r)$ and an input x, it "finds" C and returns $C(x)$. The third and final ingredient is the "collision-finding" oracle of Simon [Sim98] and Haitner et al. [HHR+15] that takes as input any oracle-aided circuit (which may access f, $\mathcal{O}$, and Eval), and returns a random collision for the circuit. It is straightforward to prove that relative to Γ there are no collision-resistant function families, and the vast majority of our effort is in showing that relative to Γ there exist a one-way permutation and an indistinguishability obfuscator. We refer the reader to Section 3 for more information.

We prove Theorems 1.2 and 1.3 by presenting an oracle Ψ relative to which the following three properties hold: (1) there exists an exponentially-secure one-way permutation, (2) there exists an exponentially-secure private-key functional encryption scheme for oracle-aided circuits, and (3) there does not exist a perfectly-complete bit-agreement protocol. Our oracle Ψ is again quite structured and consists of two main ingredients. The first ingredient is again a random permutation f that will serve as a one-way permutation. The second ingredient is a triplet $(\mathcal{K}, \mathcal{E}, \mathcal{D}^{f,\mathcal{K},\mathcal{E}})$ of functions that we use for implementing a functional encryption scheme as follows. A functional key for an oracle-aided circuit C is a handle $\mathcal{K}(\mathsf{msk}, C)$, and an encryption of a message m is a handle $\mathcal{E}(\mathsf{msk}, m, r)$, where msk is the master-secret key, $\mathcal{K}$ and $\mathcal{E}$ are randomly-chosen functions, and r is a randomly-chosen string for each encryption. The decryption algorithm is implemented using $\mathcal{D}^{f,\mathcal{K},\mathcal{E}}$ which on input a functional key $\mathsf{sk}_C = \mathcal{K}(\mathsf{msk}, C)$ and an encryption $\mathsf{ct} = \mathcal{E}(\mathsf{msk}, m, r)$ "finds" C and m, and then returns $C^f(m)$ if and only if sk_C and ct were indeed computed using the same master secret key msk. Here, as opposed to the proof of Theorem 1.1, each of the above-specified three properties is quite challenging to prove, and we refer the reader to Section 4 for more information.

1.4 Paper Organization

The remainder of this paper is organized as follows. In Section 2 we introduce the cryptographic primitives under consideration in this paper. In Sections 3 and 4 we present our results on the limitations of indistinguishability obfuscation and functional encryption, respectively. In each of these sections, prior to providing our proof, we first define the class of constructions to which the proof applies, and then provide an overview of both the main ideas underling the proof and of the structure of the proof.

2 Preliminaries

In this section we present the notation and basic definitions that are used in this work. For a distribution X we denote by $x \leftarrow X$ the process of sampling a value x from the distribution X. Similarly, for a set $\mathcal{X}$ we denote by $x \leftarrow \mathcal{X}$ the process of sampling a value x from the uniform distribution over $\mathcal{X}$. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \ldots, n\}$. A function negl :

$\mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every constant $c > 0$ there exists an integer N_c such that $\mathsf{negl}(n) < n^{-c}$ for all $n > N_c$. Throughout the paper, we denote by n the security parameter.

2.1 Indistinguishability Obfuscation for Oracle-Aided Circuits

We consider the standard notion of indistinguishability obfuscation [BGI+01, BGI+12, GGH+13] when naturally generalized to the setting of oracle-aided computations. We first define the notion of functional equivalence relative to a specific function (provided as an oracle), and then we define the notion of an indistinguishability obfuscation for a class of oracle-aided circuits. In what follows, when considering a class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of oracle-aided circuits, we assume that each $\mathcal{C}_n$ consists of circuits of size at most n.

Definition 2.1. *Let C_1 and C_2 be two oracle-aided circuits, and let f be a function. We say that C_1 and C_2 are* functionally equivalent relative to f, *denoted $C_1^f \equiv C_2^f$, if for any input x it holds that $C_1^f(x) = C_2^f(x)$.*

Definition 2.2. *A probabilistic polynomial-time algorithm $i\mathcal{O}$ is an* indistinguishability obfuscator *relative to an oracle Γ for a class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of oracle-aided circuits if the following conditions are satisfied:*

- **Functionality.** *For all $n \in \mathbb{N}$ and for all $C \in \mathcal{C}_n$ it holds that*

$$\Pr\left[C^\Gamma \equiv \widehat{C}^\Gamma \ : \ \widehat{C} \leftarrow i\mathcal{O}^\Gamma(1^n, C)\right] = 1.$$

- **Indistinguishability.** *For any probabilistic polynomial-time distinguisher $D = (D_1, D_2)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that*

$$\mathsf{Adv}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},D,\mathcal{C}}(n) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},D,\mathcal{C}}(n) = 1\right] - \frac{1}{2}\right| \le \mathsf{negl}(n)$$

for all sufficiently large $n \in \mathbb{N}$, where the random variable $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},D,\mathcal{C}}(n)$ is defined via the following experiment:

1. *$b \leftarrow \{0,1\}$.*
2. *$(C_0, C_1, \mathsf{state}) \leftarrow D_1^\Gamma(1^n)$ where $C_0, C_1 \in \mathcal{C}_n$ and $C_0^\Gamma \equiv C_1^\Gamma$.*
3. *$\widehat{C} \leftarrow i\mathcal{O}^\Gamma(1^n, C_b)$.*
4. *$b' \leftarrow D_2^\Gamma(\mathsf{state}, \widehat{C})$.*
5. *If $b' = b$ then output 1, and otherwise output 0.*

Throughout this paper we sometimes find it convenient to denote by $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},D,\mathcal{C}}(n; b, r^*)$ the above experiment when using specific values b and r^*, where b is the bit chosen in the first step of the experiment and r^* is the randomness used by the algorithm $i\mathcal{O}$ to obfuscating the challenge circuit in Step 3 of the experiment.

2.2 Private-Key Functional Encryption for Oracle-Aided Circuits

A private-key functional encryption scheme over a message space $\mathcal{M} = \{\mathcal{M}_n\}_{n\in\mathbb{N}}$ and a function space $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ is a quadruple $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ of probabilistic polynomial-time oracle-aided algorithms. The setup algorithm Setup takes as input the unary representation 1^n of the security parameter $n \in \mathbb{N}$ and outputs a master secret key msk. The key-generation algorithm KG takes as input a master secret key msk and a circuit $C \in \mathcal{C}_n$, and outputs a functional key sk_C. The encryption algorithm Enc takes as input a master secret key msk and a message $m \in \mathcal{M}_n$, and outputs a ciphertext ct. In terms of correctness we require that for all sufficiently large $n \in \mathbb{N}$, for every circuit $C \in \mathcal{C}_n$ and message $m \in \mathcal{M}_n$ it holds that $\mathsf{Dec}(\mathsf{KG}(\mathsf{msk}, C), \mathsf{Enc}(\mathsf{msk}, m)) = C(m)$ with all but a negligible probability over the internal randomness of the algorithms Setup, KG, and Enc.

In terms of security, we rely on the standard private-key indistinguishability-based notion of adaptive security (see, for example, [AAB$^+$13, BS15]) when naturally generalized to the setting of oracle-aided computations. For simplicity, we consider a *single*-challenge security notion, and we note that it is in fact equivalent to a *multiple*-challenge one (via a standard hybrid argument – as we allow adversaries to access an encryption oracle).

Definition 2.3 (Valid adversary)**.** *A probabilistic polynomial-time algorithm $\mathcal{A}$ is a* valid adversary relative to an oracle Ψ *if for all private-key functional encryption schemes* $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$*, for all $n \in \mathbb{N}$ and $b \in \{0,1\}$, and for all oracle-aided circuits C with which $\mathcal{A}$ queries the key-generation oracle* KG *it holds that $C^{\Psi}(m_0) = C^{\Psi}(m_1)$, where m_0 and m_1 are the challenge messages produced by $\mathcal{A}$.*

Definition 2.4 (Adaptive security)**.** *A private-key functional encryption scheme* $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *over a message space $\mathcal{M} = \{\mathcal{M}_n\}_{n\in\mathbb{N}}$ and an oracle-aided function space $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ is* adaptively secure relative to an oracle Ψ *if for any probabilistic polynomial-time valid adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\mathsf{Adv}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n) \overset{\mathsf{def}}{=} \left| \Pr\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n) = 1\right] - \frac{1}{2} \right| \leq \mathsf{negl}(n)$$

for all sufficiently large $n \in \mathbb{N}$, where the random variable $\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n)$ is defined via the following experiment:

1. $\mathsf{msk} \leftarrow \mathsf{Setup}^{\Psi}(1^n)$*,* $b \leftarrow \{0,1\}$.
2. $(m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_1^{\Psi,\mathsf{KG}^{\Psi}(\mathsf{msk},\cdot),\mathsf{Enc}^{\Psi}(\mathsf{msk},\cdot)}(1^n)$.
3. $c^* \leftarrow \mathsf{Enc}^{\Psi}(\mathsf{msk}, m_b; r^*)$*, where* $r^* \leftarrow \{0,1\}^*$.
4. $b' \leftarrow \mathcal{A}_2^{\Psi,\mathsf{KG}^{\Psi}(\mathsf{msk},\cdot),\mathsf{Enc}^{\Psi}(\mathsf{msk},\cdot)}(\mathsf{state}, c^*)$.
5. *If $b' = b$ then output 1, and otherwise output 0.*

Throughout the paper we sometimes find it convenient to denote by $\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$ the above experiment when using specific values msk, b, and r^*.

2.3 Oracle-Aided Collision-Resistant Function Families

We rely on the standard notion of a collision-resistant function family when generalized to the setting of oracle-aided computations. For our purposes in this paper it suffices to consider functions that compress their input by a single bit.

Definition 2.5. *A pair* $(\mathsf{Gen}, \mathsf{Eval})$ *of polynomial-time oracle-aided algorithms is a collision-resistant function family relative to an oracle Γ if it satisfies the following properties:*

- *The index-generation algorithm* Gen *is a probabilistic algorithm that on input 1^n and oracle access to Γ outputs a function index $\sigma \in \{0,1\}^{m(n)}$.*
- *The evaluation algorithm* Eval *is a deterministic algorithm that takes as input a function index $\sigma \in \{0,1\}^{m(n)}$ and a string $x \in \{0,1\}^n$, has oracle access to Γ, and outputs a string* $y = \mathsf{Eval}^{\Gamma}(\sigma, x) \in \{0,1\}^{n-1}$.

- *For any probabilistic polynomial-time oracle-aided algorithm $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{c} x \neq x' \text{ and} \\ \mathsf{Eval}^{\Gamma}(\sigma, x) = \mathsf{Eval}^{\Gamma}(\sigma, x') \end{array} \middle| \begin{array}{c} \sigma \leftarrow \mathsf{Gen}^{\Gamma}(1^n) \\ (x, x') \leftarrow \mathcal{A}^{\Gamma}(1^n, \sigma) \end{array}\right] \leq \mathsf{negl}(n)$$

for all sufficiently large $n \in \mathbb{N}$.

2.4 Oracle-Aided Key-Agreement Protocols

We rely on the standard notion of a key-agreement protocol when generalized to the setting of oracle-aided computations. For our purposes in this paper it suffices to consider key-agreement protocols in which the parties agree on a single bit, and we refer to such protocols as bit-agreement protocols.

A bit-agreement protocol consists of a pair $(\mathcal{A}, \mathcal{B})$ of probabilistic polynomial-time oracle-aided algorithms. We denote by $(k_{\mathcal{A}}, k_{\mathcal{B}}, T) \leftarrow \langle \mathcal{A}^{\Psi}(1^n; r_{\mathcal{A}}), \mathcal{B}^{\Psi}(1^n; r_{\mathcal{B}}) \rangle$ the random process of executing the protocol relative to an oracle Ψ, where $r_{\mathcal{A}}$ and $r_{\mathcal{B}}$ are the random tapes of $\mathcal{A}$ and $\mathcal{B}$, respectively, $k_{\mathcal{A}}$ and $k_{\mathcal{B}}$ are the output bits of $\mathcal{A}$ and $\mathcal{B}$, respectively, and T is the transcript of the protocol (i.e., the messages exchanged by the parties; this does not include the oracle queries/answers of the parties during the execution). In this paper we consider bit-agreement protocol that are perfectly complete (i.e., the parties always output the same bit).

Definition 2.6. *A pair $\Pi = (\mathcal{A}, \mathcal{B})$ of probabilistic polynomial-time oracle-aided algorithms is a perfectly-complete bit-agreement protocol relative to an oracle Ψ if the following two conditions hold:*

- **Perfect completeness.** *For any $n \in \mathbb{N}$ it holds that*

$$\Pr_{r_{\mathcal{A}}, r_{\mathcal{B}}}\left[k_{\mathcal{A}} = k_{\mathcal{B}} \,\middle|\, (k_{\mathcal{A}}, k_{\mathcal{B}}, T) \leftarrow \langle \mathcal{A}^{\Psi}(1^n; r_{\mathcal{A}}), \mathcal{B}^{\Psi}(1^n; r_{\mathcal{B}}) \rangle\right] = 1.$$

- **Security.** *For any probabilistic polynomial-time oracle-aided algorithm E, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\mathsf{Adv}^{\mathsf{KA}}_{\Psi,\Pi,E}(n) \stackrel{\mathsf{def}}{=} \left| \Pr\left[\mathsf{Exp}^{\mathsf{KA}}_{\Psi,\Pi,E}(n) = 1\right] - \frac{1}{2} \right| \leq \mathsf{negl}(n)$$

for all sufficiently large $n \in \mathbb{N}$, where the random variable $\mathsf{Exp}^{\mathsf{KA}}_{\Psi,\Pi,E}(n)$ is defined via the following experiment:

1. $(k_{\mathcal{A}}, k_{\mathcal{B}}, T) \leftarrow \langle \mathcal{A}^{\Psi}(1^n), \mathcal{B}^{\Psi}(1^n) \rangle$.
2. $k' \leftarrow E^{\Psi}(1^n, T)$.
3. *If $k' = k_{\mathcal{A}}$ then output 1, and otherwise output 0.*

3 Limits on the Power of Indistinguishability Obfuscation

In this section we present our negative result for constructing a collision-resistant function family from a general-purpose indistinguishability obfuscator and a one-way permutation (and even an enhanced trapdoor permutation family). First, in Section 3.1 we formally define the class of constructions to which our negative result applies. Then, in Section 3.2 we present the structure of our proof, which is provided in Sections 3.3–3.5.

3.1 The Class of Reductions

We consider fully black-box constructions of a collision-resistant function family from a general-purpose indistinguishability obfuscator and a one-way permutation (and even an enhanced trapdoor permutation family). we model these primitives as two independent building blocks due to the following four reasons. First, although indistinguishability obfuscation is known to imply one-way functions under reasonable assumptions [KMN$^+$14], it is not known to imply one-way permutations unless one assumes sub-exponential hardness [BPW15]. Second, this enables us to prove an *unconditional* result. Third, as we show in Section 3.5.3, our proof easily extends to the case where the one-way permutation is replaced by an enhanced trapdoor permutations family. Finally, and most importantly, this enables us to capture constructions that may apply the indistinguishability obfuscator to any primitive that can be constructed in a fully black-box manner from a one-way permutation (and, more general, from trapdoor permutations). For example, this enables us to capture constructions that may apply the indistinguishability obfuscator to any circuit that uses a puncturable pseudorandom function or a pseudorandom generator as a sub-routine[9].

Moreover, this also enables us to capture constructions that may apply the indistinguishability obfuscator to any primitive that can be constructed in a fully black-box manner from a one-way permutation (or trapdoor permutations) *and indistinguishability obfuscation*: Whenever an "outer" indistinguishability obfuscator is applied to a circuit that uses an "inner" indistinguishability obfuscator as a sub-routine, the functionality and security of the outer obfuscator imply that the inner obfuscator can be simply replaced by the identity function (with a suitable padding of its output). For example, this enables us to capture constructions that may apply the indistinguishability obfuscator to the decryption circuit of a general-purpose functional encryption scheme (such a scheme was recently constructed by Waters [Wat15], where in our setting we view his construction as relying on indistinguishability obfuscation for circuits that use one-way functions in a fully black-box manner). We emphasize that *black-box* access in our setting is only required with respect to the indistinguishability obfuscator and the one-way permutation themselves (e.g., applying an indistinguishability obfuscator to a circuit that uses a pseudorandom generator as a sub-routine is considered black-box access).

We now formally define the class of constructions considered in this section, tailoring our definitions to the specific primitives under consideration. We consider any implementation of a one-way permutation f and an indistinguishability obfuscator $i\mathcal{O}$ for the class of all polynomial-size oracle-aided circuits C^f. Before providing the formal definition we remind the reader that two oracle-aided circuits, C_0 and C_1, are functionally equivalent relative to a function f if for any input x it holds that $C_1^f(x) = C_2^f(x)$ (see Definition 2.1). The following definition is directly inspired by those of [Lub96, Gol00, Wee07, HHR$^+$15].

Definition 3.1. *A fully black-box construction of a collision-resistant function family from a one-way permutation and an indistinguishability obfuscator for the class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of all polynomial-size oracle-aided circuits consists of a pair of probabilistic polynomial-time oracle-aided algorithms* $(\mathsf{Gen}, \mathsf{Eval})$*, an oracle-aided algorithm M that runs in time $T_M(\cdot)$, and functions $\epsilon_{M,1}(\cdot)$ and $\epsilon_{M,2}(\cdot)$, such that the following two conditions hold:*

- **Correctness:** *For any $n \in \mathbb{N}$, for any permutation f, for any function $i\mathcal{O}$ such that $i\mathcal{O}(C; r)^f \equiv C^f$ for all $C \in \mathcal{C}$ and $r \in \{0,1\}^*$, and for any function index σ produced by $\mathsf{Gen}^{f,i\mathcal{O}}(1^n)$, it holds that $\mathsf{Eval}^{f,i\mathcal{O}}(\sigma, \cdot) : \{0,1\}^n \to \{0,1\}^{n-1}$.*

[9] We note that both puncturable pseudorandom functions and pseudorandom generators can be built from any one-way function in a fully black-box manner [GGM86, HIL$^+$99, KPT$^+$13, BW13, SW14, BGI14].

- **Black-box proof of security:** *For any permutation f, for any function $i\mathcal{O}$ such that $i\mathcal{O}(C;r)^f \equiv C^f$ for all $C \in \mathcal{C}$ and $r \in \{0,1\}^*$, for any probabilistic oracle-aided algorithm $\mathcal{A}$ that runs in time $T_{\mathcal{A}} = T_{\mathcal{A}}(n)$, and for any function $\epsilon_{\mathcal{A}} = \epsilon_{\mathcal{A}}(n)$, if*

$$\Pr\left[\begin{array}{c} x \neq x' \text{ and} \\ \mathsf{Eval}^{f,i\mathcal{O}}(\sigma,x) = \mathsf{Eval}^{f,i\mathcal{O}}(\sigma,x') \end{array} \middle| \begin{array}{c} \sigma \leftarrow \mathsf{Gen}^{f,i\mathcal{O}}(1^n) \\ (x,x') \leftarrow \mathcal{A}^{f,i\mathcal{O}}(1^n,\sigma) \end{array}\right] \geq \epsilon_{\mathcal{A}}(n)$$

for infinitely many values of n, then either

$$\Pr_{x \leftarrow \{0,1\}^n}\left[M^{f,i\mathcal{O},\mathcal{A}}(f(x)) = x\right] \geq \epsilon_{M,1}\left(T_{\mathcal{A}}(n) \cdot \epsilon_{\mathcal{A}}^{-1}(n)\right) \cdot \epsilon_{M,2}(n)$$

or

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{iO}}_{(f,i\mathcal{O}),i\mathcal{O},M^{\mathcal{A}},\mathcal{C}}(n) = 1\right] - \frac{1}{2}\right| \geq \epsilon_{M,1}\left(T_{\mathcal{A}}(n) \cdot \epsilon_{\mathcal{A}}^{-1}(n)\right) \cdot \epsilon_{M,2}(n)$$

for infinitely many values of n (see Definition 2.2 for the description of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{(f,i\mathcal{O}),i\mathcal{O},M^{\mathcal{A}},\mathcal{C}}$).

At this point we would like to highlight the following aspects of the above definition:

- ***$i\mathcal{O}$* can obfuscate *oracle-aided* circuits C^f.** As discussed above, our definition captures an indistinguishability obfuscator $i\mathcal{O}$ for the class of all polynomial-size oracle-aided circuits C^f, and thus *black-box* access in our setting is only required with respect to the indistinguishability obfuscator and the one-way permutation themselves. For example, this enables us to capture constructions that may apply the indistinguishability obfuscator to the evaluation circuit of a puncturable pseudorandom function, or to any circuit that uses a pseudorandom generator as a sub-routine.

  We note that our definition does not capture, for example, constructions that use non-interactive zero-knowledge (or witness-indistinguishable) proofs for languages that are defined relative to a one-way function. We leave it as an open problem to extend our framework to such constructions, noting that the approach of Brakerski et al. [BKS+11] seems quite promising in this direction.
- **The "security loss" functions.** Black-box constructions are typically formulated with a reduction algorithm M that runs in *polynomial* time and offers a *polynomial* security loss. In our setting, as we are interested in capturing constructions that may be based on super-polynomial security assumptions, we allow the algorithm M to run in arbitrary time $T_M(n)$ and to have an arbitrary security loss.

  In general, the security loss of a reduction is a function of the adversary's running time $T_{\mathcal{A}}(n)$, of its success probability $\epsilon_{\mathcal{A}}(n)$, and of the security parameter $n \in \mathbb{N}$. Following Luby [Lub96] and Goldreich [Gol00], we simplify the presentation by considering Levin's unified security measure $T_{\mathcal{A}}(n) \cdot \epsilon_{\mathcal{A}}^{-1}(n)$. Specifically, our definition captures the security loss of a reduction by considering an "adversary-dependent" security loss $\epsilon_{M,1}(T_{\mathcal{A}}(n) \cdot \epsilon_{\mathcal{A}}^{-1}(n))$, and an "adversary-independent" security loss $\epsilon_{M,2}(n)$. By considering arbitrary security loss functions, we are indeed able to capture constructions that rely on super-polynomial security assumptions. For example, in the recent construction of Bitansky et al. [BPW15] (and in various other recent constructions based on indistinguishability obfuscation), the adversary-dependent loss is polynomial whereas the adversary-independent loss is sub-exponential[10].

[10] This is also the situation, for example, when using "complexity leveraging" for arguing that any selectively-secure identity-based encryption scheme is in fact adaptively secure.

3.2 Proof Overview and the Oracle Γ

Our result is obtained by presenting an oracle Γ relative to which there exist an exponentially-secure one-way permutation f and an exponentially-secure indistinguishability obfuscator $i\mathcal{O}$ for the class of all polynomial-size oracle-aided circuits C^f, but any collision-resistant function family can be easily broken. We prove the following theorem:

Theorem 3.2. *Let* $(\mathsf{Gen}, \mathsf{Eval}, M, T_M, \epsilon_{M_1}, \epsilon_{M_2})$ *be a fully black-box construction of a collision-resistant function family from a one-way permutation* f *and an indistinguishability obfuscator for the class of all polynomial-size oracle-aided circuits* C^f *(see Definition 3.1). Then, at least one of the following propertied holds:*

1. $T_M(n) \geq 2^{\zeta n}$ *for some constant* $\zeta > 0$ *(i.e., the reduction runs in exponential time).*
2. $\epsilon_{M,1}(n^c) \cdot \epsilon_{M,2}(n) \leq 2^{-n/50}$ *for some constant* $c > 1$ *(i.e., the security loss is exponential).*

In particular, the theorem implies that if the running time $T_M(\cdot)$ of the reduction is sub-exponential and the adversary-dependent security loss $\epsilon_{M,1}(\cdot)$ is polynomial as in the vast majority of constructions, then the adversary-independent security loss $\epsilon_{M,2}(\cdot)$ must be exponential (thus ruling out even constructions that rely on sub-exponential security assumptions).

In what follows we describe the oracle Γ (which is in fact a distribution over oracles), and then explain the structure of our proof. The proof is inspired by a combination of ideas that were developed in the work of Haitner et al. [HHR+15] (generalizing ideas of Simon [Sim98], Gennaro et al. [GGK+05], and Wee [Wee07]) and in the work of Brakerski et al. [BKS+11], as discussed below.

The oracle Γ. The oracle Γ is a quadruple $\left(f, \mathcal{O}, \mathsf{Eval}^{f,\mathcal{O}}, \mathsf{CollFinder}^{f,\mathcal{O},\mathsf{Eval},\mathcal{R}}\right)$ that is defined as follows:

- **The function $f = \{f_n\}_{n \in \mathbb{N}}$.** For every $n \in \mathbb{N}$ the function f_n is a uniformly chosen permutation over $\{0,1\}^n$.
  Looking ahead, we will prove that f is a one-way permutation relative to Γ (as detailed in Section 3.5.3, our result easily extends to the case where f is replaced by a family of trapdoor permutations).
- **The functions $\mathcal{O} = \{\mathcal{O}_n\}_{n \in \mathbb{N}}$ and $\mathsf{Eval}^{f,\mathcal{O}}$.** For every $n \in \mathbb{N}$ the function $\mathcal{O}_n$ is a uniformly chosen permutation over $\{0,1\}^{2n}$. The function $\mathsf{Eval}^{f,\mathcal{O}}$ on input $(y, x) \in \{0,1\}^*$ finds the unique oracle-aided circuit $C \in \{0,1\}^{|y|/2}$ (i.e., C is encoded as a $|y|/2$-bit string) and the unique string $r \in \{0,1\}^{|y|/2}$ such that $\mathcal{O}_{|y|/2}(C, r) = y$ (note that the uniqueness is guaranteed by the fact that $\mathcal{O}_{|y|/2}$ is a permutation over $\{0,1\}^{|y|}$). Then, it computes and outputs $C^f(x)$.
  Looking ahead, we will use $\mathcal{O}$ and Eval for realizing an indistinguishability obfuscator $i\mathcal{O}$ relative to Γ for the class of all polynomial-size oracle-aided circuits C^f.
- **The function $\mathsf{CollFinder}^{f,\mathcal{O},\mathsf{Eval},\mathcal{R}}$.** On input an encoding of an oracle-aided circuit C that may access f, $\mathcal{O}$ and Eval, the function $\mathsf{CollFinder}^{f,\mathcal{O},\mathsf{Eval},\mathcal{R}}$ outputs a uniform pair (w, w') such that $C^{f,\mathcal{O},\mathsf{Eval}}(w) = C^{f,\mathcal{O},\mathsf{Eval}}(w')$.
  Looking ahead, we will use $\mathsf{CollFinder}$ for finding a non-trivial collision in any circuit that compresses its input. More specifically, $\mathsf{CollFinder}$ is provided with an infinite collection $\mathcal{R}$ of permutations, where for every possible circuit $C^{f,\mathcal{O},\mathsf{Eval}} : \{0,1\}^m \to \{0,1\}^{m'}$ the collection $\mathcal{R}$ contains two uniformly and independently sampled permutations, π^1_C and π^2_C, over $\{0,1\}^m$. Now, given a circuit $C^{f,\mathcal{O},\mathsf{Eval}}$, the oracle $\mathsf{CollFinder}$ sets $w = \pi^1_C(0^m)$, and then computes $w' = \pi^2_C(t)$ for the lexicographically smallest $t \in \{0,1\}^m$ such that $C(\pi^2_C(t)) = C(w)$.[11]

[11] Note that w is uniformly distributed over $\{0,1\}^m$, and that w' is uniformly distributed over $\{0,1\}^m$ subject to forming a collision with w.

Equipped with the oracle Γ, our proof consists of the following three parts.

Part 1: Finding collisions in any compressing circuit. As in the work of Simon [Sim98], it is straightforward to observe that there are no collision-resistant function families relative to Γ. Specifically, for any $n \in \mathbb{N}$, for any functions f and $\mathcal{O}$ as above, and for any oracle-aided circuit $C = C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}} : \{0,1\}^n \to \{0,1\}^{n-1}$, querying $\mathsf{CollFinder}$ with C results in a non-trivial collision with probability at least $1/4$ over the choice of the relevant permutations in $\mathcal{R}$. More generally, n independent such queries result in at least one non-trivial collision with probability at least $1-(3/4)^n$.[12] We refer the reader to Section 3.3 for the formal proof.

Part 2: The existence of an indistinguishability obfuscator. The most challenging part in this section is in proving that there exists a general-purpose indistinguishability obfuscator relative to Γ. Our construction of such an obfuscator $i\mathcal{O}^{\mathcal{O}}$ is quite intuitive: For obfuscating an oracle-aided circuit $C = C^f \in \{0,1\}^n$ (i.e., a circuit that is encoded as an n-bit string for some $n \in \mathbb{N}$), the obfuscator $i\mathcal{O}^{\mathcal{O}}$ samples $r \leftarrow \{0,1\}^n$ uniformly at random, computes $\widehat{C} = \mathcal{O}_n(C, r)$, and outputs the circuit $C'(\cdot) = \mathsf{Eval}(\widehat{C}^f, \cdot)$ (i.e., the obfuscated circuit C' consists of a single Eval-gate with hardwired input $\widehat{C}^f$). The definition of the function Eval guarantees that $i\mathcal{O}$ preserves the functionality of the obfuscated circuit C.

The vast majority of our effort is focused on showing that obfuscations of any two circuits that are functionally equivalent relative to f are computationally indistinguishable even for algorithms that can access the oracle Γ. Essentially, the technical challenge here is that the oracle $\mathsf{CollFinder}$ may perform an *exponential* number of queries to the oracles $\mathcal{O}$ and Eval, and thus most of the standard arguments that are typically used in black-box impossibility results are not applicable here. By generalizing the approach of Gennaro et al. [GGK$^+$05] and its extensions by Haitner et al. [HHR$^+$15] (for dealing with Simon's oracle $\mathsf{CollFinder}$) and by Brakerski et al. [BKS$^+$11] (for dealing with indistinguishability experiments), we prove that it is hard to distinguish between the obfuscations of any two functionally-equivalent circuits.

We note that our approach heavily relies on the specific setting of *indistinguishability* obfuscation, and it is currently not clear whether or not it can be extended to stronger notions of indistinguishability such as *differing-input* obfuscation [ABG$^+$13, BCP14, GGH$^+$14b]. The following is a simplified variant of the theorem that we prove in Section 3.4:

Theorem 3.3 (Simplified variant)**.** *For any probabilistic oracle-aided algorithm $\mathcal{A}$ that runs in time at most $2^{n/15}$, and for any permutation f, it holds that*

$$\left| \Pr_{\mathcal{O},\mathcal{R}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n) = 1 \right] - \frac{1}{2} \right| \leq 2^{-n/40}.$$

Part 3: The existence of a one-way permutation. Finally, we prove that f is a one-way permutation relative to Γ. The following is a simplified variant of the theorem that we prove in Section 3.5, and also here our inspiration is rooted at ideas originated in [GGK$^+$05, HHR$^+$15, BKS$^+$11] that we generalize to our oracle Γ:

[12] $\mathsf{CollFinder}$ is deterministic, and therefore n independent responses for same input circuit can be obtained, for example, by appending to the description of the circuit a "dummy" counter (thus having n different circuits with the same functionality).

Theorem 3.4 (Simplified variant)**.** *For any probabilistic oracle-aided algorithm $\mathcal{A}$ that runs in time at most $2^{n/50}$, and for any function $\mathcal{O}$, it holds that*

$$\Pr_{\substack{f,\mathcal{R}\\ x\leftarrow\{0,1\}^n}}\left[\mathcal{A}^{\Gamma}(f(x))=x\right]\leq 2^{-n/50}.$$

Given the above discussion we are now ready to prove Theorem 3.2.

Proof of Theorem 3.2. Let $(\mathsf{Gen},\mathsf{Eval},M,T_M,\epsilon_{M,1},\epsilon_{M,2})$ be a fully black-box construction of a collision-resistant function family from a one-way permutation f and an indistinguishability obfuscator $i\mathcal{O}$ for the class $\mathcal{C}$ of all polynomial-size oracle-aided circuits C^f (recall Definition 3.1). Note that in our setting, relative to the oracle Γ, this means we allow the algorithms Gen and Eval to access f, $\mathcal{O}$ and Eval (but we do not allow them to access $\mathsf{CollFinder}$).

Then, as discussed above, there exists an oracle-aided algorithm $\mathcal{A}$ that runs in polynomial time $T_{\mathcal{A}}(n)$, such that for any f and $\mathcal{O}$, it holds that

$$\Pr_{\mathcal{R}}\left[\begin{array}{c} x\neq x' \textit{ and} \\ \mathsf{Eval}^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(\sigma,x)=\mathsf{Eval}^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(\sigma,x')\end{array}\middle|\begin{array}{c}\sigma\leftarrow\mathsf{Gen}^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(1^n)\\ (x,x')\leftarrow\mathcal{A}^{\Gamma}(1^n,\sigma)\end{array}\right]\geq\epsilon_{\mathcal{A}}(n), \tag{3.1}$$

where $\epsilon_{\mathcal{A}}(n)=1/4$ for all values of $n\in\mathbb{N}$ (as noted above, we can in fact use $\epsilon_{\mathcal{A}}(n)=1-(3/4)^n$). Definition 3.1 then states that there are two possible cases to consider: $\mathcal{A}$ can be used either for breaking the indistinguishability obfuscator $i\mathcal{O}$, or for inverting the one-way permutation f.

In the first case, noting that Eq. (3.1) holds for any f and $\mathcal{O}$, we obtain from Definition 3.1 that for any f it holds that

$$\left|\Pr_{\mathcal{O},\mathcal{R}}\left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},M^{\mathcal{A}},\mathcal{C}}(n)=1\right]-\frac{1}{2}\right|\geq\epsilon_{M,1}\left(T_{\mathcal{A}}(n)\cdot 4\right)\cdot\epsilon_{M,2}(n),$$

where M runs in time $T_M(n)$. The algorithm M may invoke $\mathcal{A}$ on various security parameters (i.e., in general M is not restricted to invoking $\mathcal{A}$ only on security parameter n), and we denote by $\ell(n)$ the maximal security parameter on which M invokes $\mathcal{A}$ (when M itself is invoked on security parameter n). Thus, viewing $M^{\mathcal{A}}$ as a single algorithm, its running time $T_{M^{\mathcal{A}}}(n)$ satisfies $T_{M^{\mathcal{A}}}(n)\leq T_M(n)\cdot T_{\mathcal{A}}(\ell(n))$ (this follows since M may invoke $\mathcal{A}$ at most $T_M(n)$ times, and the running time of $\mathcal{A}$ on each such invocation is at most $T_{\mathcal{A}}(\ell(n))$). Theorem 3.3 then implies that either $2^{n/15}\leq T_{M^{\mathcal{A}}}(n)$ or $\epsilon_{M,1}\left(T_{\mathcal{A}}(n)\cdot 4\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/40}$. In the first sub-case, noting that $\ell(n)\leq T_M(n)$, we obtain that

$$2^{n/15}\leq T_{M^{\mathcal{A}}}(n)\leq T_M(n)\cdot T_{\mathcal{A}}(\ell(n))\leq T_M(n)\cdot T_{\mathcal{A}}(T_M(n)).$$

The running time $T_{\mathcal{A}}(n)$ of the adversary $\mathcal{A}$ is some fixed polynomial in n, and therefore $T_M(n)\geq 2^{\zeta n}$ for some constant $\zeta>0$. In the second sub-case, we have that $\epsilon_{M,1}\left(T_{\mathcal{A}}(n)\cdot 4\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/40}$, and since $T_{\mathcal{A}}(n)$ is some fixed polynomial in n we obtain that $\epsilon_{M,1}\left(n^c\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/40}$ for some constant $c>1$.

In the second case, noting again that Eq. (3.1) holds for any f and $\mathcal{O}$, we obtain from Definition 3.1 that for any $\mathcal{O}$ it holds that

$$\Pr_{\substack{f,\mathcal{R}\\ x\leftarrow\{0,1\}^n}}\left[\left(M^{\mathcal{A}}\right)^{\Gamma}(f(x))=x\right]\geq\epsilon_{M_1}\left(T_{\mathcal{A}}(n)\cdot 4\right)\cdot\epsilon_{M_2}(n),$$

where M runs in time $T_M(n)$. As in the first case, viewing $M^{\mathcal{A}}$ as a single algorithm, its running time $T_{M^{\mathcal{A}}}(n)$ satisfies $T_{M^{\mathcal{A}}}(n)\leq T_M(n)\cdot T_{\mathcal{A}}(\ell(n))$. Theorem 3.4 then implies that either $2^{n/50}\leq T_{M^{\mathcal{A}}}(n)$ or $\epsilon_{M_1}\left(T_{\mathcal{A}}(n)\cdot 4\right)\cdot\epsilon_{M_2}(n)\leq 2^{-n/50}$. As in the first case, this implies that either $T_M(n)\geq 2^{\zeta n}$ for some constant $\zeta>0$ or $\epsilon_{M,1}\left(n^c\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/50}$ for some constant $c>1$. ■

3.3 Breaking Any Collision-Resistant Function Family Relative to Γ

The following is a straightforward claim, which shows that there does not exist any collision-resistant function family relative to Γ:

Claim 3.5. *There exists a probabilistic polynomial-time algorithm $\mathcal{A}$, such that for any functions f and $\mathcal{O}$, for any $n \in \mathbb{N}$, and for any oracle-aided circuit $C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}} : \{0,1\}^n \to \{0,1\}^{n-1}$, it holds that:*

$$\Pr_{\mathcal{R}}\left[\begin{array}{c} w \neq w' \text{ and} \\ C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w) = C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w') \end{array} \middle| (w,w') \leftarrow \mathcal{A}^{\Gamma}(1^n, C) \right] \geq \frac{1}{4} \ .$$

Proof. Fix n, f, and $\mathcal{O}$. On input $(1^n, C)$, the algorithm $\mathcal{A}$ simply sends the circuit C as a query to the oracle $\mathsf{CollFinder}$, and receives back a pair (w, w'). We claim that the probability of the event in which $C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w) = C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w')$ and $w \neq w'$ is at least $1/4$ over the choice of w and w'. Indeed, the circuit C maps n-bits strings into $(n-1)$-bit strings, and therefore at least 2^{n-1} inputs have some non-trivial collision. $\mathsf{CollFinder}$ evaluates C on some random w, and thus there exists a non-trivial collision for w with probability at least $2^{n-1}/2^n = 1/2$. In addition, $\mathsf{CollFinder}$ samples a uniform w' in $C^{-1}(C(w))$, and therefore with probability at least $1/2$ this value is different than w. That is, $\mathsf{CollFinder}$ finds a non-trivial collision with probability of at least $1/4$ over the choice of w and w'. ■

3.4 $i\mathcal{O}$ is an Indistinguishability Obfuscator Relative to Γ

In this section we show that there exists a general-purpose indistinguishability obfuscator relative to Γ. The construction is as follows:

Construction 3.6. *The (randomized) algorithm $i\mathcal{O}(1^n, C)$ is given as input an oracle-aided circuit C (represented as n-bit string), chooses a random $r \leftarrow \{0,1\}^n$, and computes $\mathcal{O}_n(C, r) = \widehat{C}$. Then, it outputs the oracle-aided circuit $C'(\cdot) = \mathsf{Eval}(\widehat{C}, \cdot)$.*

We now prove that this construction is an indistinguishability obfuscator for the class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of all polynomial-size oracle-aided circuits C^f. Before proceeding to a formal statement of the main theorem in this section, we first define the notion of a q-query algorithm:

Definition 3.7. *We say that an oracle-aided algorithm $\mathcal{A}$ is a $(q_f, q_{\mathcal{O}}, q_{\mathsf{Eval}}, q_{\mathsf{CollFinder}})$-query algorithm if for every $n \in \mathbb{N}$ and for any input $y \in \{0,1\}^n$, an execution $\mathcal{A}^{\Gamma}(1^n, y)$ makes at most $q_f(n)$,$q_{\mathcal{O}}(n)$,$q_{\mathsf{Eval}}(n)$, and $q_{\mathsf{CollFinder}}(n)$ queries to the oracles f,$\mathcal{O}$,Eval,and $\mathsf{CollFinder}$, respectively, and its oracles queries to Eval and $\mathsf{CollFinder}$ are with circuits of size at most $q_{\mathsf{Eval}}(n)$ and $q_{\mathsf{CollFinder}}(n)$, respectively. In case $\mathcal{A}$ is allowed to make an unbounded number of queries to some oracle $\gamma \in \{f, \mathcal{O}, \mathsf{Eval}, \mathsf{CollFinder}\}$, we write $q_\gamma = \infty$. We say that $\mathcal{A}$ is a q-query algorithm if it is a (q, q, q, q)-query algorithm.*

It is clear that Construction 3.6 preserves the functionality of the obfuscated circuits since $\mathcal{O}$ is injective. Regarding indistinguishability, we show that the advantage of any adversary in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*)$ is very small (Definition 2.2). Recall that in this experiment, the adversary is invoked with no input, queries the oracle Γ and outputs a pair (C_0, C_1) of oracle-aided circuits that are functionally equivalent relative to f. The challenger obfuscates C_b using randomness r^*, and the adversary has to guess which one of them was obfuscated. In the remainder of this section we prove the following theorem:

Theorem 3.8. *For any q-query algorithm $\mathcal{A}$ with $q(n) \leq 2^{n/15}$ it holds that*

$$\left| \Pr_{\substack{\Gamma \\ (b,r^*) \leftarrow \{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*) = 1 \right] - \frac{1}{2} \right| \leq 2^{-n/40} \ .$$

In fact, we show that the above holds even for any fixing of the oracle oracles $f = \{f_n\}_{n\in\mathbb{N}}$ and $\mathcal{O}_{-n} = \{\mathcal{O}_k\}_{k\in\mathbb{N}, k\neq n}$.

Proof overview. Consider an algorithm $\mathcal{A}$ participating in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$. We claim that the adversary can output the correct bit b only if it obtains a specific piece of useful information about $\widehat{C} = i\mathcal{O}(C_b)$ via one of its oracle queries. We explicitly define when $\mathcal{A}$ obtains such information, and show that it happens with a very small probability.

In particular, we claim that $\mathcal{A}$ obtains information about $\widehat{C}$ only if one of its queries "hits" the values $\mathcal{O}_n(C_0, r^*)$ or $\mathcal{O}_n(C_1, r^*)$ during the execution. Such a hit may be a triggered by a direct query to the oracle $\mathcal{O}$, or an indirect query to $\mathcal{O}$ through some $\mathsf{CollFinder}$ query or some Eval query. Moreover, this query may occur *before* the adversary receives the challenge $\widehat{C}$ (i.e., in Step 2 of the exepriment), or *after* it receives the challenge $\widehat{C}$ (i.e., in Step 4 of the experiment). We now formally define the event in which $\mathcal{A}$ obtains useful information about $\widehat{C}$. We distinguish between the case where it makes such a query before receiving $\widehat{C}$ and after receiving it.

We define the following event which indicates that $\mathcal{A}_1$ hits the randomness r^*, used by the experiment to obfuscate the challenged circuit:

Definition 3.9. *For a given $n \in \mathbb{N}$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we denote by $\mathsf{InitHit}_{\mathcal{A}}$ the event in which one of the following queries are made by $\mathcal{A}_1$ in Step 2 of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$:*

1. *$\mathcal{A}_1$ queries $\mathcal{O}_n$ on some (C, r^*), with $|C| = n$ for some $C \in \{0,1\}^n$.*
2. *$\mathcal{A}_1$ queries Eval on some $\left(\widetilde{C}, x\right)$ with $\left|\widetilde{C}\right| = 2n$, for which $\mathcal{O}_n^{-1}\left(\widetilde{C}\right) = \left(C, r^*\right)$ for some $C \in \{0,1\}^n$.*
3. *$\mathcal{A}_1$ queries $\mathsf{CollFinder}$ on some circuit D and receives back (w, w') such that some gate in the computation of $D^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w)$ or $D^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w')$ is an $\mathcal{O}_n$-gate with input (C, r^*) as above, or an Eval gate with input $(\widetilde{C}, x)$ with $|\widetilde{C}| = 2n$ for which $\mathcal{O}_n^{-1}(\widetilde{C}) = (C, r^*)$ for some $C \in \{0,1\}^n$ as above.*

We next define the event which indicates that $\mathcal{A}_2$ hits $\mathcal{O}_n$ on (C_0, r^*) or (C_1, r^*), where C_0 and C_1 are the pair of oracle-aided circuits outputted by $\mathcal{A}_1$, and r^* is the randomness used by the experiment to obfuscate the challenge circuit C_b:

Definition 3.10. *For a given $n \in \mathbb{N}$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we consider the following two events that may occur during Step 4 of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$, where C_0 and C_1 are the two oracle-aided circuits that were outputted by $\mathcal{A}$ in Step 2 of the experiment:*

1. *We denote by r^*-$\mathsf{hit}_{\mathcal{A}}$ the event in which $\mathcal{A}_2$ makes either one of the two following direct oracle queries to $\mathcal{O}_n$: (C_0, r^*) or (C_1, r^*).*
2. *We denote by $\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}$ the event in which $\mathcal{A}_2$ makes a $\mathsf{CollFinder}$ query C and $\mathsf{CollFinder}$ responds with (w, w'), such that one of the computations of $C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w)$ or $C^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w')$ has an $\mathcal{O}_n$-gate with input (C_0, r^*) or (C_1, r^*).*

The proof proceeds in three modular parts. First, we claim that if there exists an adversary $\mathcal{A}$ that obtains any advantage during the experiment, then this advantage can occur only if one of the above events holds. That is, in Section 3.4.1 we prove the following Claim:

Claim 3.11. *For every $n \in \mathbb{N}$, for every q-query algorithm $\mathcal{A}$ and for every $\epsilon > 0$, if*

$$\left| \Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r^*) \leftarrow \{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*) = 1 \right] - \frac{1}{2} \right| \geq \epsilon \ ,$$

then

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r^*) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{A}} \ \vee \ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \ \vee \ \mathsf{InitHit}_{\mathcal{A}} \ \right] \geq \epsilon \ .$$

In the second part of the proof, we show in Section 3.4.2 that the case where either r^*-$\mathsf{hit}_{\mathcal{A}}$, $\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}$ or $\mathsf{InitHit}_{\mathcal{A}}$ occur can be reduced to the case where only r^*-$\mathsf{hit}_{\mathcal{A}}$-occurs. That is, we show that if there exists an adversary $\mathcal{A}$ for which one of the events r^*-$\mathsf{hit}_{\mathcal{A}}, \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}$ or $\mathsf{InitHit}_{\mathcal{A}}$ occurs during an execution of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$, then there exists an adversary $\mathcal{B}$, that performs a comparable number of oracle queries, for which in the execution of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{B}, \mathcal{C}}(n; b, r^*)$ only $\mathsf{InitHit}_{\mathcal{B}}$ occurs (with a polynomially-small loss). We show:

Claim 3.12. *For every $n \in \mathbb{N}$, for any q-query algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with $q(n) \leq 2^{n/4}$, if:*

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r^*) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{A}} \ \vee \ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \ \vee \ \mathsf{InitHit}_{\mathcal{A}} \ \right] \geq \frac{1}{q(n)} \ ,$$

in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$, then there exists a $Q(n)$-query algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ with $Q(n) = 2q(n)^2$ for which in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{B}, \mathcal{C}}(n; b, r^*)(n; b, r^*)$ it holds that:

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r^*) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}} \ \wedge \ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}} \ \wedge \ \overline{\mathsf{InitHit}_{\mathcal{B}}} \ \right] \geq 1/q(n)^5 \ .$$

In the third part of the proof, in Section 3.4.3 we show that the above event can occur only with small probability. In particular, we show that the existence of an adversary that succeeds in hitting r^* implies a short representation for the function $\mathcal{O}_n$. We show that:

Claim 3.13. *For any $Q(n)$-query algorithm $\mathcal{B}$ with $Q(n) = 2^{n/7}$, the following holds in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{B}, \mathcal{C}}(n; b, r^*)$:*

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r^*) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}} \ \wedge \ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}} \ \wedge \ \overline{\mathsf{InitHit}_{\mathcal{B}}} \ \right] < 2^{-n/8} \ .$$

From Claims 3.11, 3.12 and 3.13 we conclude Theorem 3.8.

Proof of Theorem 3.8. Assume towards contradiction that there exists a q-query algorithm $\mathcal{A}$ where $q(n) = 2^{n/15}$, and infinitely many n's such that:

$$\left| \Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r) \leftarrow \{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*) = 1 \right] - \frac{1}{2} \right| > 2^{-n/40} \ .$$

From Claim 3.11, this implies that

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b, r) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{A}} \ \vee \ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \ \ \vee \ \mathsf{InitHit}_{\mathcal{A}} \ \right] > 2^{-n/40} \ .$$

However, using Claim 3.12, this implies the existence of a $Q(n)$-query algorithm $\mathcal{B}$ with $Q(n) = 2q(n)^2 = 2 \cdot 2^{2n/15} \leq 2^{n/7-1}$, for which in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n; b, r^*)$ it holds that:

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r)\leftarrow\{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}} \ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}} \ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right] > \left(2^{-n/40}\right)^5 = 2^{-n/8},$$

and this is in contradiction to Claim 3.13. ■

3.4.1 From Distinguishing to Hitting

In this section, we claim that if there exists an adversary $\mathcal{A}$ that obtains any advantage during the execution of the experiment, then one of the events r^*-hit, $\mathsf{CollFinder\text{-}hit}$ or $\mathsf{InitHit}$ occurs. We prove:

Claim 3.14. *For every $n \in \mathbb{N}$, for every q-query algorithm $\mathcal{A}$ and for every $\epsilon > 0$, if*

$$\left| \Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*) = 1\right] - \frac{1}{2} \right| \geq \epsilon\ ,$$

then

$$\Pr_{\substack{O_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{A}} \ \vee\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \ \vee\ \mathsf{InitHit}_{\mathcal{A}}\ \right] \geq \epsilon\ .$$

Proof. Note that:

$$\begin{aligned}
&\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*) = 1\right] \\
&\quad\leq \Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*) = 1 \ \mid\ \overline{r^*\text{-}\mathsf{hit}} \ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}} \ \wedge\ \overline{\mathsf{InitHit}}\ \right] \\
&\qquad + \Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit} \ \vee\ \mathsf{CollFinder\text{-}hit} \ \vee\ \mathsf{InitHit}\ \right]
\end{aligned}$$

We prove the claim by showing that the following holds:

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}} \left[\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*) = 1 \mid \overline{r^*\text{-}\mathsf{hit}} \ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}} \ \wedge\ \overline{\mathsf{InitHit}}\ \right] = \frac{1}{2}\ .$$

We start by fixing the entire probability space, except for the oracle $\mathcal{O}_n$ on all the values $\mathcal{O}_n \backslash \mathcal{O}_n(\cdot; r^*)$, that is, we fix everything except for the answers for queries of type $\mathcal{O}_n(C, r^*)$ for some $C \in \{0,1\}^n$. We now claim that giving this fixing, and assuming that $\mathsf{InitHit}$ does not occur, the two circuits C_0, C_1 that $\mathcal{A}_1$ outputs in Step 2 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n; b, r^*)$ are fully-determined. In particular, we show that all queries that $\mathcal{A}_1$ may produce in step 2 can be answered. Specifically:

- The oracle f is fully-determined.
- All answers to queries to $\mathcal{O}$ are determined except for queries $\mathcal{O}_n(\cdot, r^*)$. However, since $\mathsf{InitHit}$ does not occur, there are no queries on undetermined values.
- All answers to queries $\left(\widetilde{C}, x\right)$ to Eval where $\widetilde{C} \in \{0,1\}^{2m}$ and $n \neq m$ are determined. Specifically, the oracle $\mathcal{O}_m$ is fully-determined.

- All answers to queries $\big(\widetilde{C}, x\big)$ to Eval where $\widetilde{C} \in \{0,1\}^{2n}$ are determined. This is because $\mathsf{InitHit}$ does not occur, and therefore there are no Eval queries for value $\widetilde{C}$, for which $\mathcal{O}_n^{-1}\big(\widetilde{C}\big) = (C, r^*)$ for some C. As a result, for all values $\widetilde{C}$ that are queried it holds that $\mathcal{O}_n^{-1}\big(\widetilde{C}\big) = (C, r)$ for some $C \in \{0,1\}^n$ and $r' \neq r^*$. Therefore, the answer $C(x)$ is fully-determined.
- All answers to queries $C^{f,\mathcal{O},\mathsf{Eval}}$ to $\mathsf{CollFinder}$ are fully determined. Recall that $\mathcal{R}$ is determined. We can compute $C^{f,\mathcal{O},\mathsf{Eval}}(\pi_C^1(0^m))$ since there is no $\mathsf{InitHit}$, and therefore this computation does not involve $\mathcal{O}_n(\cdot; r^*)$. Moreover, we can enumerate over all $s \in \{0,1\}^m$ in lexicographically increasing order, and output $w' = \pi_C^2(s)$ for the minimal s such that $C^{f,\mathcal{O},\mathsf{Eval}}(w') = y$ and this evaluation does not produce a r^*-hit (i.e., there is no $\mathcal{O}_n$-gate with input (C, r^*) or Eval-gate of some $\widetilde{C} = \mathcal{O}_n(C, r^*)$). If such a hit occurs, we move to the next s.

As a result, the two circuits C_0, C_1 that $\mathcal{A}_1$ outputs at Step 2 and the state state are fully-determined. We now proceed with fixing the oracle $\mathcal{O}_n(\cdot, r^*)$, with the exception of two values only: $\mathcal{O}_n(C_0, r^*), \mathcal{O}_n(C_1, r^*)$. Let $t, t' \in \{0,1\}^{2n}$ the only two images of $\mathcal{O}_n$ that are not fixed. We will consider two cases: One corresponds to $b = 0$, where we set $\mathcal{O}_n(C_0, r^*)$ to t (and $\mathcal{O}_n(C_1, r^*)$ to t') and one (corresponds to $b = 1$) where we assign the opposite values. In both cases $\mathcal{A}_2$ is given the value t. The two cases are equally likely but yield different values to b. We show that if $\mathcal{A}_2^\Gamma$ makes no hits, then the view of $\mathcal{A}_2$ is independent of b and it must output the same value in the two cases. In particular, we show that all queries that $\mathcal{A}_2$ may query in Step 4 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$ can be answered. Specifically:

- The oracle f is fully determined.
- All queries to $\mathcal{O}$ are determined except for (C_0, r^*) and (C_1, r^*). However, since r^*-hit does not occur, there are no queries on undetermined values.
- All queries $\big(\widetilde{C}, x\big)$ to Eval where $\widetilde{C} \in \{0,1\}^{2m}$ and $n \neq m$ are determined. Specifically, the oracle $\mathcal{O}_m$ is fully-determined, and one can find the pair (C, r') where $|C| = |r'| = m$ for which $\mathcal{O}_m(C, r') = \widetilde{C}$ and reply with $C^f(x)$. Recall that the oracle-aided circuit C does not have an access to $\mathcal{O}$.
- On a query $\big(\widetilde{C}, x\big)$ to Eval, where $\widetilde{C} \in \{0,1\}^{2n}$, we have the following two options. In case $\widetilde{C} \neq t$ and $\widetilde{C} \neq t'$, then the value $\mathcal{O}_n^{-1}\big(\widetilde{C}\big)$ is determined, and so the query $\mathsf{Eval}\big(\widetilde{C}, x\big)$ is determined. In the second case, we have that $\widetilde{C} = t$ or $\widetilde{C} = t'$, and thus the two possible answers are $C_0^f(x)$ or $C_1^f(x)$. However, since $C_0^f \equiv C_1^f$, the two values $C_0^f(x)$, $C_1^f(x)$ are the same, and therefore the query can be answered even though we did not fully determine $\mathcal{O}_n$.
- On a query C to $\mathsf{CollFinder}$, recall that $\mathcal{R}$ is determined. We can compute the value $y = C^{f,\mathcal{O},\mathsf{Eval}}(\pi_C^1(0^m))$ since there is no $\mathsf{CollFinder}$-hit, and therefore there is no need to compute $\mathcal{O}_n(C_0^f, r^*)$ or $\mathcal{O}_n(C_1^f, r^*)$ during this evaluation. Moreover, we can enumerate over all $s \in \{0,1\}^m$ in lexicographically increasing order, and output $w' = \pi_C^2(s)$ for the minimal s such that $C^{f,\mathcal{O},\mathsf{Eval}}(w') = y$ and this evaluation does not produce a $\mathsf{CollFinder}$-hit (i.e., there is no $\mathcal{O}_n$-gate with input (C_0^f, r^*) or (C_1^f, r^*)). If such a hit occurs, we move to the next s.

The claim follows. ■

3.4.2 Avoiding Hits

We move to show that the events $\mathsf{CollFinder}$-hit and $\mathsf{InitHit}$ can be avoided with only a small loss. Before proceeding to the main claim in this section, we first show that the event $\mathsf{InitHit}$ occurs with relatively small probability. Intuitively, the view of the adversary $\mathcal{A}_1$, i.e., the view of $\mathcal{A}$ *before* it outputs the pair of circuits (C_0, C_1), is completely independent of r^*. Since it has only limited amount evaluations of the oracle $\mathcal{O}_n$ (even using some indirect queries through the $\mathsf{CollFinder}$-oracle), the probability to hit r^* is relatively small. We have:

Claim 3.15. *For every $n \in \mathbb{N}$, for every q-query algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the execution of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$ it holds that:*

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b,r^*) \leftarrow \{0,1\}^{n+1}}} [\ \mathsf{InitHit}\ ] \leq 2q(n)^2 \cdot 2^{-n}\ .$$

Proof. Fix the entire probability space, with the exception of r^*. We define the set $F_{\mathcal{O}}$ containing all queries to $\mathcal{O}_n$ during Step 2 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$. In particular, these values include:

- *Direct $\mathcal{O}_n$-queries.* $F_{\mathcal{O}}$ contains any direct query (C, r) (with $|(C, r)| = 2n$) that was sent to $\mathcal{O}_n$ by $\mathcal{A}^{\Gamma}$ in Step 2 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$.
- *$\mathcal{O}_n$ values, as a result of* Eval*-queries.* For any Eval-query $\big(\widetilde{C}, x\big)$ with $|\widetilde{C}| = 2n$, we add to $F_{\mathcal{O}}$ the value $\mathcal{O}_n^{-1}\big(\widetilde{C}\big)$.
- *Indirect $\mathcal{O}_n$ values, as a result of* $\mathsf{CollFinder}$ *queries.* For a query C to $\mathsf{CollFinder}$, where the size of this circuit is bounded by $q(n)$, let (w, w') be the output pair of $\mathsf{CollFinder}$ on this query. We follow the computation of $C^{f, \mathcal{O}, \mathsf{Eval}}(w)$ and $C^{f, \mathcal{O}, \mathsf{Eval}}(w')$, and add to $F_{\mathcal{O}}$ at most $2q(n)$ values, which are either inputs to $\mathcal{O}_n$ gates, or Eval-queries, as above.

Note that f-gates do not add any elements to $F_{\mathcal{O}}$. Since $\mathcal{A}_1$ makes at most $q(n)$-oracle queries, the size of $F_{\mathcal{O}}$ is bounded by $2q(n)^2$, and therefore define at most $2q(n)^2$ different values of r. Since r^* is chosen uniformly at random after $F_{\mathcal{O}}$ is fully determined, we have that:

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b,r^*) \leftarrow \{0,1\}^{n+1}}} [\ \mathsf{InitHit}\ ] \leq \Pr_{r^* \leftarrow \{0,1\}^n} [\ \exists C \in \{0,1\}^n, \ \text{ s.t. } (C, r^*) \in F_{\mathcal{O}}\ ] \leq \frac{|F_{\mathcal{O}}|}{2^n} \leq 2q(n)^2 \cdot 2^{-n}\ .$$

■

We now proceed to the main claim of this section. We show that if there exists an adversary $\mathcal{A}$ for which during the execution of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$ one of the events r^*-hit, $\mathsf{CollFinder}$-hit or $\mathsf{InitHit}$ occurs, then there exists an adversary $\mathcal{B}$ for which during the execution of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{B}, \mathcal{C}}(n; b, r^*)$ only r^*-hit-occurs with roughly the same probability, while the other two event do not occur. Formally:

Claim 3.16. *For every $n \in \mathbb{N}$, for any q-query algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ where $q(n) \leq 2^{n/4}$, if:*

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b,r^*) \leftarrow \{0,1\}^{n+1}}} [\ r^*\text{-}\mathsf{hit}_{\mathcal{A}}\ \vee\ \mathsf{CollFinder}\text{-}\mathsf{hit}_{\mathcal{A}}\ \vee\ \mathsf{InitHit}_{\mathcal{A}}\ ] \geq \frac{1}{q(n)} \tag{3.2}$$

in the execution of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{A}, \mathcal{C}}(n; b, r^*)$, then there exists $Q(n)$-query algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ with $Q(n) = 2q(n)^2$, such that in an execution of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma, i\mathcal{O}, \mathcal{B}, \mathcal{C}}(n; b, r^*)$ the following holds:

$$\Pr_{\substack{\mathcal{O}_n, \mathcal{R} \\ (b,r^*) \leftarrow \{0,1\}^{n+1}}} \left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder}\text{-}\mathsf{hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right] \geq 1/q(n)^5\ .$$

Proof. Intuitively, we have already seen that $\mathsf{InitHit}_{\mathcal{A}}$ rarely occurs. We now show that if $\mathsf{CollFinder}$-$\mathsf{hit}_{\mathcal{A}}$ occurs with relatively high probability, then we can translate it into an r^*-hit. We define the algorithm $\mathcal{B}$ as follows:

The algorithm $\mathcal{B}$. We separate between the algorithm $\mathcal{B}_1$ and $\mathcal{B}_2$ (i.e., between steps 2 and 4 of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*))$. In particular, algorithm $\mathcal{B}_1$ is identical to $\mathcal{A}_1$ and acts exactly the same. Formally, $\mathcal{B}_1$ on input 1^n invokes $\mathcal{A}_1$ on the same input, delivers all its oracle queries to Γ and outputs the pair of circuits (C_0, C_1) and the state state that $\mathcal{A}$ outputs.

The algorithm $\mathcal{B}_2$, upon receiving the challenge $\widehat{C}$, invokes $\mathcal{A}_2$ on $(\mathsf{state}, \widehat{C})$ and delivers all of $\mathcal{A}_2$ queries to Γ with the following exception: for each $\mathsf{CollFinder}$-query C that $\mathcal{A}_2$ submits, $\mathcal{B}_2$ first chooses a random $z \in \{0,1\}^m$ and evaluates $C^{f,\mathcal{O},\mathsf{Eval}}(z)$. If some $\mathcal{O}_n$ gate during this evaluation has output $\widehat{C}$ and input (C_0, r^*) or (C_1, r^*), then $\mathcal{B}_2$ halts and in the former case (i.e., $\mathcal{O}_n(C_0, r^*) = \widehat{C}$) it outputs 0 whereas in the latter case it outputs 1. Otherwise, $\mathcal{B}_2$ submits C to $\mathsf{CollFinder}$ and delivers the results (w, w') back to $\mathcal{A}_2$. If $\mathcal{B}_2$ did not halt before the termination of $\mathcal{A}_2$'s computation, then it outputs the output of $\mathcal{A}_2$ and halts.

Without loss of generality, we assume that the queries of $\mathcal{A}$ to the oracle $\mathsf{CollFinder}$ in Step 4 in the experiment are distinct then those of Step 2. Otherwise, $\mathcal{B}$ stores the queries and responses for $\mathsf{CollFinder}$ in Step 2, and in case of a repeating query, $\mathcal{B}$ can answer it immediately.

We remark that $\mathcal{B}$ makes at most the same amount of queries to $\mathsf{CollFinder}$ as $\mathcal{A}$, and makes at most $2q(n)^2$ queries to $\mathcal{O}_n$: $\mathcal{A}$ may make at most $q(n)$ direct queries to $\mathcal{O}_n$, and each one of the $q(n)$ queries to $\mathsf{CollFinder}$ may lead to additional $q(n)$ queries (as each circuit of size $q(n)$ may contain up to $q(n)$ gates of type $\mathcal{O}_n$). As a result, $\mathcal{B}$ is a Q-query algorithm, with $Q(n) = 2q(n)^2$.

Note that:

$$\begin{aligned}
&\Pr\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{A}} \;\vee\; \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \;\vee\; \mathsf{InitHit}_{\mathcal{A}} \,\right] \\
&\quad = \Pr\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}} \;\wedge\; \overline{\mathsf{InitHit}_{\mathcal{A}}} \,\right] + \Pr\left[\, \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \;\vee\; \mathsf{InitHit}_{\mathcal{A}} \,\right] \\
&\quad = \Pr\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}} \;\wedge\; \overline{\mathsf{InitHit}_{\mathcal{A}}} \,\right] \\
&\qquad\quad + \Pr\left[\, \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{InitHit}_{\mathcal{A}}} \,\right] \;+\; \Pr\left[\, \mathsf{InitHit}_{\mathcal{A}} \right]
\end{aligned}$$

where the probabilities are taken over $\mathcal{O}_n, \mathcal{R}, (b, r^*) \leftarrow \{0,1\}^{n+1}$, and that the same holds for $\mathcal{B}$ in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)(n)$. Recall that $q(n) \leq 2^{n/4}$, and from Claim 3.15 we have that:

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R} \\ (b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\, \mathsf{InitHit}_{\mathcal{A}} \,\right] \leq 2q(n)^2 \cdot 2^{-n} \leq 2 \cdot 2^{-n/2} \leq \frac{1}{2q(n)} \;.$$

Therefore, assuming that Eq. (3.2) occurs, we have:

$$\Pr\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}} \;\wedge\; \overline{\mathsf{InitHit}_{\mathcal{A}}} \,\right] + \Pr\left[\, \mathsf{CollFinder\text{-}hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{InitHit}_{\mathcal{A}}} \,\right] \geq \frac{1}{2q(n)}$$

where the probability is taken over $\mathcal{O}_n, \mathcal{R}$ and $(b, r^*) \leftarrow \{0,1\}^{n+1}$. It is easy to see that if $\mathcal{A}$ wins in $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n;b,r^*)$ without producing any $\mathsf{CollFinder\text{-}hit}$, $\mathsf{InitHit}$ in its queries, then so does $\mathcal{B}$. Formally, if

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R} \\ (b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{A}} \;\wedge\; \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}} \;\wedge\; \overline{\mathsf{InitHit}} \,\right] \geq \frac{1}{4q(n)} \;,$$

then

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R} \\ (b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\, r^*\text{-}\mathsf{hit}_{\mathcal{B}} \;\wedge\; \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}} \;\wedge\; \overline{\mathsf{InitHit}} \,\right] \geq \frac{1}{4q(n)} \geq \frac{1}{q(n)^5} \;,$$

and the claim follows. We therefore focus on the case where the above does not occur, and thus, we assume:

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \right] \geq \frac{1}{4q(n)}\ .$$

We will prove the following Lemma below:

Lemma 3.17. *For every choice of $f, \mathcal{O}$, for every $r^* \in \{0,1\}^n$ and $b \in \{0,1\}$ if*

$$\Pr_{\mathcal{R}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{A}}}\ \right] \geq \frac{1}{8q(n)} \tag{3.3}$$

in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n;b,r^*)$*, then*

$$\Pr_{\mathcal{R}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}}\ \right] \geq \frac{1}{1024q(n)^3} \tag{3.4}$$

in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)(n;b,r)$.

This lemma will be proven below, and it helps us to conclude the proof via a standard averaging argument. Let

$$T = \left\{(\mathcal{O}_n, r^*, b) \mid \Pr_{\mathcal{R}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \right] \geq \frac{1}{8q(n)}\right\}\ .$$

Then, $\Pr_{\mathcal{O}_n,(b,r^*)\leftarrow\{0,1\}^{n+1}}\left[(\mathcal{O}_n,b,r^*)\in T\right] \geq 1/8q(n)$, and Lemma 3.17 implies that for every such $(\mathcal{O}_n,b,r^*)\in T$ we have that Eq. (3.4) follows. This implies that:

$$\begin{aligned}&\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right]\\ \geq\ &\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[(\mathcal{O}_n,b,r^*)\in T\right]\cdot\frac{1}{1024q(n)^3} \geq \frac{1}{8q(n)}\cdot\frac{1}{1024\cdot q(n)^3} \geq \frac{1}{q(n)^5}\ .\end{aligned}$$

This completes the proof of Claim 3.16. ■

Proof of Lemma 3.17. Fix f, O, r^*, b. The adversary $\mathcal{B}$ acts exactly as $\mathcal{A}$ up to the Step 4 of the experiment. Thus, it outputs the same pair of circuits (C_0, C_1). Moreover, recall that $\mathcal{A}$ does not make a query to $\mathsf{CollFinder}$ in Step 4 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n;b,r^*)$ that it has already made before.

- We denote by $D_1,\ldots,D_q$ the random variables corresponding to $\mathcal{A}_2$'s $\mathsf{CollFinder}$-queries in Step 4 of $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n;b,r^*)$. In addition, we denote by $(w_1,w_1'),\ldots,(w_q,w_q')$ the random variables corresponding to the answers returned by $\mathsf{CollFinder}$-oracle.
- Given a circuit D_i and an input w, we say that w produces an (D_i,r^*)-hit if some $\mathcal{O}_n$-gate in the execution of $D_i^{f,\mathcal{O},\mathsf{Eval}^{f,\mathcal{O}}}(w)$ has input (C_0^f,r^*) or (C_1^f,r^*).
- For every $1\leq i\leq q$ let α_i denote the probability that w_i produces a (D_i,r^*)-hit. That is: $\alpha_i \overset{\text{def}}{=} \Pr_{\mathcal{R}}\left[w_i \text{ produces a } (D_i,r^*)\text{−hit}\right]$. Note that this is the same probability as w_i' produces a (D_i,r^*) hit, since the probability is taken over $\mathcal{R}$ and therefore each of w_i,w_i' is uniform.
- For every $1\leq i\leq q$, we denote by JUMP_i the event that $\alpha_i > 1/(32q^2)$ and let $\mathsf{JUMP} = \cup_{i=1}^{q}\mathsf{JUMP}_i$.

We have that:

$$\Pr_{\mathcal{R}}\left[\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{A}}}\ \right] \geq \Pr_{\mathcal{R}}\left[\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{A}}}\ \mid \mathsf{JUMP}\right] \cdot \Pr\left[\mathsf{JUMP}\right] \tag{3.5}$$

We now prove lower bounds for these two terms. We start with bounding the probability that JUMP occurs. First, if JUMP does not occur, then we claim that the α_i's are too small in order to produce an r^*-hit with noticeable probability. That is:

Claim 3.18. $\Pr_{\mathcal{R}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \mid \overline{\mathsf{JUMP}}\ \right] \leq 1/(16q)$.

Proof. Assuming that JUMP does not occur, that is $\alpha_i \leq 1/(32q^2)$ for every $1 \leq i \leq q$. It holds that:

$$\begin{aligned} &\Pr_{\mathcal{R}}\left[\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \mid \overline{\mathsf{JUMP}}\right] \\ &\leq\ \sum_{i=1}^{q} \Pr_{\mathcal{R}}\left[w_i \text{ or } w_i' \text{ produces a } (D_i, r^*)\text{-hit}\right] \leq 2 \cdot q \cdot \frac{1}{32q^2} = \frac{1}{16q}\ . \end{aligned}$$

■

Eq. (3.3) implies that in particular:

$$\Pr_{\mathcal{R}}\left[\mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \right] \geq \frac{1}{8q}\ .$$

In addition:

$$\begin{aligned} \Pr_{\mathcal{R}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \right] &\leq\ \Pr_{\mathcal{R}}\left[\mathsf{JUMP}\right] + \Pr_{\mathcal{R}}\left[\ \mathsf{CollFinder\text{-}hit}_{\mathcal{A}}\ \wedge\ \overline{\mathsf{InitHit}}\ \mid\ \overline{\mathsf{JUMP}}\right] \\ &\leq\ \Pr_{\mathcal{R}}\left[\ \mathsf{JUMP}\ \right] + \frac{1}{16q} \end{aligned}$$

Thus,

$$\Pr_{\mathcal{R}}\left[\mathsf{JUMP}\right] \geq \frac{1}{8q} - \frac{1}{16q} = \frac{1}{16q}\ .$$

Note that both JUMP_i and JUMP have the exact same probability in the two experiments $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{A},\mathcal{C}}(n;b,r^*)$ and $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)(n;b,r)$.

We now consider the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)(n;b,r^*)$. Assume that the event JUMP occurs, and denote by i^* the minimal $1 \leq i \leq q$ for which JUMP_i occurs. When $\mathcal{A}$ submits the query D_{i^*}, then $\mathcal{B}$ has probability α_{i^*} to retrieve r^* without submitting the query to $\mathsf{CollFinder}$. In addition, since i^* is the minimal $1 \leq i \leq q$ for which JUMP_i occurs, then with high probability $\mathsf{CollFinder}$'s answers to $D_1, \ldots, D_{i^*-1}$ do not produce an hit. We consider the following events:

- None of the queries $D_1, \ldots, D_{i^*-1}$ produces an r^*-hit. Since for every such D_i the event JUMP_i does not occur, then, exactly as Claim 3.18, the probability of this event is at least $1 - 1/(16q)$.
- Given D_{i^*}, $\mathcal{B}$ samples a random z which produces a (D_{i^*}, r^*)-hit (that is, $r^*\text{-}\mathsf{hit}_{\mathcal{B}}$ occurs). Since JUMP_{i^*} occurs, the probability of this event is α_{i^*}.

Note that these two events are independent (since the permutations in $\mathcal{R}$ are chosen independently). Putting these together, we obtain:

$$\Pr_{\mathcal{R}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \mid\ \mathsf{JUMP}\ \right] \geq \left(1 - \frac{1}{16q}\right) \cdot \frac{1}{32q^2} \geq \frac{1}{64q^2}$$

Therefore, we get that:

$$\begin{aligned}&\Pr_{\mathcal{R}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right]\\ &\quad\geq\ \Pr_{\mathcal{R}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \mid\ \mathsf{JUMP}\ \right]\cdot\Pr\left[\ \mathsf{JUMP}\ \right]\geq\frac{1}{64q^2}\cdot\frac{1}{16q}=\frac{1}{1024q^3}\end{aligned}$$

■

3.4.3 From Hitting to Compressing

Let $\mathcal{B}$ be a Q-query algorithm for which in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)$ it holds that

$$\Pr_{\substack{\mathcal{O}_n,\mathcal{R}\\(b,r^*)\leftarrow\{0,1\}^{n+1}}}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right]\geq\epsilon(n)\ .$$

We now show that the probability $\epsilon(n)$ is very small. In particular, we show that if the above occurs, then $\mathcal{O}_n$ can be compressed. Fix $\mathcal{R}$ and $b\in\{0,1\}$, and let $\mathcal{O}_{-n}=\{\mathcal{O}_m\}_{m\in\mathbb{N},m\neq n}$. We show that:

Claim 3.19. *For any Q-query $\mathcal{B}=(\mathcal{B}_1,\mathcal{B}_2)$ with $Q(n)\leq 2^{n/7}$, for any f, $\mathcal{R}$ and $\mathcal{O}_{-n}$, the following holds in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)$:*

$$\Pr_{\mathcal{O}_n,r^*\leftarrow\{0,1\}^n}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right]<2^{-n/8}\ .$$

Proof. We first prove that for any $f,\mathcal{R},\mathcal{O}_{-n}$ and for any algorithm $\mathcal{B}$ that makes at most $Q(n)$ queries to $\mathcal{O}_n$ and at most $Q(n)$ queries to $\mathsf{CollFinder}$ of circuits up to size $Q(n)$, if

$$\Pr_{r^*\leftarrow\{0,1\}^n}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right]\geq\epsilon\ ,$$

then, given $f,\mathcal{O}_{-n},\mathcal{R}$ and $\mathcal{B}$, the function $\mathcal{O}_n$ has succinct representation.

Succinct description of $\mathcal{O}_n$. We show that $\mathcal{O}_n$ can be described using some partial truth table Z, a set of pre-images X and set of images Y. We start by building the set Y.

Note that since $\mathsf{InitHit}$ does not occur, the first step in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)$ is independent of r^*. We now follow the computation of $\mathcal{B}_1$ in the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)$ and store in Z all the inputs-outputs pairs of $\mathcal{O}_n$ that are necessary for Step 2 of the experiment. These values include:

- All direct $\mathcal{O}_n$-queries.
- For any Eval-query $\big(\widetilde{C},x\big)$ with $\big|\widetilde{C}\big|=2n$, we add to Z the pair $\big(\mathcal{O}_n^{-1}\big(\widetilde{C}\big),\widetilde{C}\big)$.
- For any $\mathsf{CollFinder}$-query C of size up to $Q(n)$, let (w,w') be the output of $\mathsf{CollFinder}$. We follow the computations of $C^{f,\mathcal{O},\mathsf{Eval}}(w)$ and $C^{f,\mathcal{O},\mathsf{Eval}}(w')$ and add to Z at most $2Q(n)$ values, which are the input-output pair of $\mathcal{O}_n$ gates in C, or Eval-queries that involve $\mathcal{O}_n$ values.

As in Claim 3.15, the number of the values that we store in Z at this point is at most $2Q(n)^2$. Note that at the end of this stage, the circuits C_0,C_1 and the state state that the adversary $\mathcal{B}_1$ outputs in Step 2 of the experiment are fully-determined, given $\mathcal{B}_1,f,\mathcal{R},\mathcal{O}_{-n}$ and Z.

At this point, according to the experiment $\mathsf{Exp}^{\mathsf{iO}}_{\Gamma,i\mathcal{O},\mathcal{B},\mathcal{C}}(n;b,r^*)$, the randomness r^* is chosen, $\mathcal{B}_2$ is given state and $t=\mathcal{O}_n(C_b;r^*)$, and during the execution, $r^*\text{-}\mathsf{hit}_{\mathcal{B}}$ occurs but without producing $\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}$ and no $\mathsf{InitHit}_{\mathcal{B}}$. In order to clear the notation, we consider the function $\mathcal{O}_{n,C_b}(\cdot)\overset{\text{def}}{=}$

$\mathcal{O}_n(C_b,\cdot)$. We first add to the set Z all the input/ output pairs of all functions $\mathcal{O}_n \setminus \mathcal{O}_n(C_b,\cdot)$. Moreover, we change $\mathcal{B}_2$ such that it outputs the randomness r^* once it produces such a r^*-$\mathsf{hit}_{\mathcal{B}}$. Then, we have that:

$$\Pr_{r^*\leftarrow\{0,1\}^n}\left[\mathcal{B}_2^{\Gamma}(\mathsf{state},\mathcal{O}_n(C_b,r^*)) = r^* \ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right] > \epsilon\ .$$

where state and C_b are defined as above.

Denote by $I \subseteq \{0,1\}^{2n}$, the set of elements $t \in \{0,1\}^{2n}$ on which $\mathcal{B}_2$ successfully inverts $\mathcal{O}_n(\cdot)$ (in fact, it inverts $\mathcal{O}_{n,C_b}(\cdot) \overset{\text{def}}{=} \mathcal{O}_n(C_b;\cdot)$) with no $\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}$ and no $\mathsf{InitHit}_{\mathcal{B}}$. We first remove from I all elements that already in Z (the necessary elements for the evaluation of $(\mathsf{state},C_0,C_1) \leftarrow \mathcal{B}_1^{\Gamma}(1^n)$), which are at most $2Q(n)^2$ elements. We claim that there exists a relatively large set $Y \subseteq I$, such that the value of $\mathcal{O}_{n,C_b}$ on the set Y is fully determined given $\mathcal{B}, I, Z, f, \mathcal{O} \setminus \mathcal{O}_{n,C_b}$ and $\mathcal{R}$.

We define the set Y via the following sequential process. Initially, Y is empty, and we remove the lexicographically smallest element t from I and insert it into Y. Then, we follow the computation of $\mathcal{B}_2^{\Gamma}(\mathsf{state},t)$ and denote by $F_t = \{t_1,\ldots,t_k\}$ the set of images of $\mathcal{O}_n$ that are necessary for this computation. These values includes:

1. At most $Q(n)$ values which are direct $\mathcal{O}_{n,C_b}$ queries of $\mathcal{B}_2^{\Gamma}$.
2. No value is added due to Eval-query. We show later how we can simulate this oracle.
3. At most $2 \cdot Q(n) \cdot Q(n)$ outputs as a result of $\mathsf{CollFinder}$-gates. In particular, let $(C_1,\ldots,C_q)$ denote the queries to $\mathsf{CollFinder}$, and let $(w_1,w_1'),\ldots,(w_q,w_q')$ denote the corresponding answers. We follow the computations of both $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$ and $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$. Each is a circuit of size $Q(n)$, which may contain at most $Q(n)$ gates of type $\mathcal{O}_{n,C_b}$ (note that the circuits that are sent to Eval cannot contain $\mathcal{O}$-gates, but only f-gates). We add to F_t all the inputs-outputs of $\mathcal{O}_{n,C_b}$-gates.

From the above, we conclude that $|F_t| \leq 3Q(n)^2$. We now remove from I the set F_t (which not all of them are necessarily in the set I). We continue to the next iteration, until the set I is emptied.

On the size of Y. In each iteration one element is inserted to Y, and at most $3Q(n)^2$ elements are removed from I. The set I initially contains at least $\epsilon \cdot 2^n - 2Q(n)^2 \geq \epsilon \cdot 2^n - 3Q(n)^2$ elements, then when the process is terminated we have that $|Y| \geq \epsilon \cdot 2^n/3Q(n)^2 - 1$.

Reconstruction. We now show that $\mathcal{O}_n$ is completely determined given $X, Y, Z, \mathcal{B}, f, \mathcal{O}\setminus\mathcal{O}_{-n}$ and $\mathcal{R}$. In fact, we show that the entire truth table of $\mathcal{O}_n$ can be reconstructed. Since Z contains all pairs of $\mathcal{O}_n \setminus \mathcal{O}_{n,C_b}$, it is enough to show that the truth table of $\mathcal{O}_{n,C_b}$ can be reconstructed.

We start by showing that the computation of $(\mathsf{state},C_0,C_1) \leftarrow \mathcal{B}_1^{\Gamma}(1^n)$ can be simulated, and always outputs the same pair of circuits (C_0,C_1) and state. We show that all the oracle queries of $\mathcal{B}_1$ can be answered. Specifically:

- The simulator is able to answer all oracle queries of type f, since f is fully-determined.
- Any direct query to $\mathcal{O}_{-n}$ can be answered since $\mathcal{O}_{-n}$ is fully-determined. Any direct query of $\mathcal{B}_1$ to $\mathcal{O}_n$ can be answered since this computation is deterministic, and all the queries to $\mathcal{O}_n$ are stored in Z.
- Any Eval query with $\left(\widetilde{C},x\right)$, with $|\widetilde{C}| \neq 2n$ can be inverted since $\mathcal{O}_{-n}$ is fixed. An Eval query with $\left|\widetilde{C}\right| = 2n$ can be inverted since the pre-image is explicitly stored in Z.

- Any $\mathsf{CollFinder}$ query C_i, with $|C_i| \leq Q(n)$ can be answered. In particular, $\mathcal{R}$ is fully-determined, and therefore (w, w') can be reconstructed. Specifically, the simulator computes $\pi^1_{C_i}(0^m)$ and receives w. Then, it enumerates over all $t \in \{0,1\}^m$ in lexicographically increasing order, and stops with the first $w' = \pi^2_{C_i}(t)$ for which $C_i^{f,\mathcal{O},\mathsf{Eval}}(w')$ can be computed (i.e., all queries to $\mathcal{O}_n$ can be answered and appear in Z) and for which $C_i^{f,\mathcal{O},\mathsf{Eval}}(w) = C_i^{f,\mathcal{O},\mathsf{Eval}}(w')$.

At this point, $\mathcal{B}_1$ outputs the pair of circuits (C_0, C_1) and state. For the rest, we focus on the computation of $\mathcal{B}_2$ on the input state and some $t \in \{0,1\}^{2n}$.

For each $t \in Y$, taken in lexicographical increasing order, we reconstruct $r^* = \mathcal{O}^{-1}_{n,C_b}(t)$, by following the computation of $B_2^{\Gamma}(\mathsf{state}, t)$ and answering all its oracle queries. Specifically, since f is given and fixed, all oracle queries to f can be answered. Answering oracle queries to $\mathcal{O} \setminus \mathcal{O}_{n,C_b}$ is also trivial, since it is given. We will show below how to answer Eval-queries, $\mathcal{O}_{n,C_b}$-queries and $\mathsf{CollFinder}$-queries

Answering Eval-queries. On each Eval query $(\widetilde{C}, x)$, we proceed as follow. If $\widetilde{C} = t$, then return $C_b(x)$ and continue with the computation. Otherwise, we look for some pair (C, r') with $m = |C| = |r'| = |\widetilde{C}|/2$ and $C \neq C_b$, for which $\widetilde{C} = \mathcal{O}_m(C, r')$. If such a pair (C, r') exists, we return $C^f(x)$. Otherwise, if no answer was given so far, then it must hold that $\widetilde{C} = \mathcal{O}_n(C_b, r')$ for some randomness r'. Although we do not explicitly know r', we can still return $C_b^f(x)$.

Answering $\mathcal{O}_{n,C_b}$-queries. We now show how to answer a query of $\mathcal{O}_{n,C_b}$ (i.e., $\mathcal{O}_n$ query with $\mathcal{O}_n(C_b, \cdot)$). On input r', if $\mathcal{O}_{n,C_b}(r')$ is known, i.e., it was either reconstructed earlier or is in Z, then this value is determined and we can return it. Otherwise, if the value is not known, we claim that it must be that $\mathcal{O}_{n,C_b}(r') = t$ and the simulator can return r' and halt. This is because the following cases covers all possibilities:

1. If $r' \notin X$, then we have the pair $(r', \mathcal{O}_{n,C_b}(r'))$ in Z.
2. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') <_{lex} t$ then the simulator already reconstructed image of r' and we can return it.
3. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') >_{lex} t$ then this case is impossible. We have that both $t \in Y$ and $\mathcal{O}_{n,C_b}(r') \in Y$, but t was inserted to Y before $\mathcal{O}_{n,C_b}(r')$ was inserted to Y. Therefore, $\mathcal{O}_{n,C_b}(r')$ should have been removed from I, and in particular not inserted into Y.
4. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') = t$.

Answering $\mathsf{CollFinder}$-queries. We proceed to show how queries to $\mathsf{CollFinder}$ can be answered. On input $C_i^{f,\mathcal{O},\mathsf{Eval}}$, the simulator computes $w_i = \pi^1_{C_i}(0^m)$ and evaluates $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$. Note that this computation may involve some additional $\mathcal{O}_{n,C_b}$-queries, which we claim that these can be handled and discuss it below. It may also involve some $f, \mathcal{O} \setminus \mathcal{O}_{n,C_b}$ and Eval queries, which can be handled as above. Then, it enumerates over all $t \in \{0,1\}^m$ in lexicographically increasing order, and stops with the first $w'_i = \pi^2_{C_i}(t)$ for which $C_i^{f,\mathcal{O},\mathsf{Eval}}(w'_i)$ can be computed (i.e., it is able to answer all $\mathcal{O}_{n,C_b}$-queries as we discuss below) and that $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i) = C_i^{f,\mathcal{O},\mathsf{Eval}}(w'_i)$. It then answers to $\mathcal{B}_2$ with the pair (w_i, w'_i).

We now show that the simulator can evaluate $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$ and $C_i^{f,\mathcal{O},\mathsf{Eval}}(w'_i)$, and answers all $\mathcal{O}_{n,C_b}$-queries. Specifically, on a query r' for $\mathcal{O}_{n,C_b}$, the simulator proceeds as follows:

1. If $r' \notin X$, then the value $\mathcal{O}_{n,C_b}(r')$ is given in Z, and the simulator can return it.

2. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') <_{lex} t$ then the simulator has already reconstructed t and returns it.
3. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') >_{lex} t$ then this case is impossible (as above).
4. If $r' \in X$ and $\mathcal{O}_{n,C_b}(r') = t$, then this is impossible, as we condition on the event that $\mathsf{CollFinder\text{-}hit}$ does not occur.

Concluding the proof. Note that our representation of $\mathcal{O}_n$ consist of the following parts: the set of pre-images X, set of images Y and set of pairs Z. The set of pairs Z contains the pairs of all the collection of functions $\mathcal{O}_n \setminus \mathcal{O}_{n,C_b}$, in addition to all pairs of $\mathcal{O}_{n,C_b}$ that are necessary for the evaluation of $\mathcal{B}_1^{\Gamma}(1^n)$, in addition to some pairs that $\mathcal{B}_2$ also uses.

In order to describe the collection of functions $\mathcal{O}_n \setminus \mathcal{O}_{n,C_b}$, we first require to describe the images of these functions, and pairing all sources and images. This costs $\log{(2^{2n}-2^n)!} + \log\binom{2^{2n}}{2^{2n}-2^n}$ bits. Note that this already represents the image of $\mathcal{O}_{n,C_b}$ as well.

In order to represent $\mathcal{O}_{n,C_b}$, we store the sets X, Y (where $|X| = |Y|$) and the partial truth-table Z. This requires $2\log\binom{2^n}{|X|} + \log(2^n - |X|)!$ bits. Since there are $2^{2n}!$ permutations over $\{0,1\}^{2n}$, the fraction of functions $\mathcal{O}_{n,C_b}$ for which:

$$\Pr_{r^*\leftarrow\{0,1\}^n}\left[\mathcal{B}_2^{\Gamma}(\mathsf{state}, \mathcal{O}_{n,C_b}(r^*)) = r^* \ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right] \geq \epsilon\ ,$$

is at most:

$$\frac{(2^{2n}-2^n)!\cdot\binom{2^{2n}}{2^{2n}-2^n}\cdot\binom{2^n}{|X|}^2\cdot(2^n-|X|)!}{2^{2n}!} = \frac{\binom{2^n}{|X|}^2\cdot(2^n-|X|)!}{2^n!} = \frac{\binom{2^n}{|X|}}{|X|!}\ .$$

Using the inequalities $a! \geq (a/e)^a$ and $\binom{2^n}{a} \leq (2^n e/a)^a$, we get:

$$\frac{\binom{2^n}{|X|}}{|X|!} \leq \left(\frac{2^n\cdot e}{|X|}\right)^{|X|}\cdot\left(\frac{e}{|X|}\right)^{|X|} = \left(\frac{2^n e^2}{|X|^2}\right)^{|X|}$$

Taking $\epsilon = 2^{-n/7}$, and recall that $Q(n) \leq 2^{n/7}$, we get that:

$$|X| \geq \epsilon\cdot\frac{2^n}{3(Q(n))^2} - 1 \geq \frac{2^n}{8\cdot 2^{3n/7}} = 2^{4n/7-3}\ ,$$

and thus for sufficiently large n's the above is upper bound by:

$$\left(\frac{2^n e^2}{|X|^2}\right)^{|X|} \leq \left(\frac{64\cdot 2^n\cdot e^2}{2^{8n/7}}\right)^{|X|} \leq \left(\frac{64\cdot e^2}{2^{n/7}}\right)^{|X|} \leq 2^{-|X|} \leq 2^{-2^{4n/7}+3}\ .$$

Therefore, we conclude that for any algorithm $\mathcal{B}$ that makes at most $Q(n) = 2^{n/7}$ queries to $\mathcal{O}_{n,C_b}$ and $Q(n)$ queries to $\mathsf{CollFinder}$, and its $Q(n)$-oracle queries to $\mathsf{CollFinder}$ are bounded to circuits of size $Q(n)$, it holds that:

$$\Pr_{\mathcal{O}_n, r^*\leftarrow\{0,1\}^n}\left[\ r^*\text{-}\mathsf{hit}_{\mathcal{B}}\ \wedge\ \overline{\mathsf{CollFinder\text{-}hit}_{\mathcal{B}}}\ \wedge\ \overline{\mathsf{InitHit}_{\mathcal{B}}}\ \right] < 2^{-n/7} + 2^{-2^{-4n/7+3}} \leq 2^{-n/8}\ .$$

■

3.5 f is a One-Way Permutation Relative to Γ

In this section we prove that the function f is one-way permutation relative to the oracle Γ. We prove the following theorem:

Theorem 3.20. *For every q-query algorithm $\mathcal{A}$ with $q(n) = 2^{n/50}$, it holds that:*

$$\Pr_{\Gamma, y \leftarrow \{0,1\}^n} \left[\mathcal{A}^{\Gamma}(y) = f^{-1}(y) \right] \leq 2^{-n/50} \ .$$

We will prove that the theorem holds even for any fixing of the oracles $\mathcal{O} = \{\mathcal{O}_n\}_{n \in \mathbb{N}}$ and $f_{-n} = \{f_m\}_{m \in \mathbb{N}, m \neq n}$. That is, we will show that for every $\mathcal{O}$, f_{-n}, and for every algorithm $\mathcal{A}$ as above:

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n} \left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y) \right] \leq 2^{-n/50} \ .$$

In the following we do not bound the number of queries the algorithms may make to the $\mathcal{O}$-oracle; recall that a (q, ∞, q, q)-query algorithm is an algorithm that may make at most q queries to f, Eval and $\mathsf{CollFinder}$, but unbounded number of queries to the oracle $\mathcal{O}$.

Consider an algorithm $\mathcal{A}$ that is given a random input $y \in \{0,1\}^n$ and has oracle access to $\Gamma = (f, \mathcal{O}, \mathsf{Eval}, \mathsf{CollFinder})$. We distinguish between two cases: one in which $\mathcal{A}$ "obtains information" on $f_n^{-1}(y)$ via one of its $\mathsf{CollFinder}$ queries, and the other in which none of $\mathcal{A}$'s queries provide "sufficient information" to obtain $f^{-1}(y)$. We explicitly define what events may provide information to $\mathcal{A}$ in some of the $\mathsf{CollFinder}$-queries. We define:

Definition 3.21. *A $\mathsf{CollFinder}$-query $C^{f,\mathcal{O},\mathsf{Eval}}$ produces a y-hit if $\mathsf{CollFinder}$ outputs (w, w') such that at least one of the following holds:*

1. *Some f_n-gate in the computation of $C(w)$ or $C(w')$ has output y.*
2. *Some Eval-gate in the computation of $C(w)$ or $C(w')$ has input $(\widehat{D}, a)$ for which the following holds:*
   - *(a) Let (D, r) be the pair such that $|r| = |D| = m$ and $\mathcal{O}_m(D, r) = \widehat{D}$.*
   - *(b) Some f_n-gate in the computation of $D^f(a)$ has output y.*

We denote by $\mathsf{CollHit}_y$ the event in which one of the $\mathsf{CollFinder}$-queries made by $\mathcal{A}$ in the computation of $\mathcal{A}^{\Gamma}(y)$ produces a y-hit.

The proof proceeds in two modular parts. In the first part of the proof, we show that the case $\mathsf{CollHit}_y$ occurs can be reduced to the case where the event $\mathsf{CollHit}_y$ does not occur. In particular, we show that if $\mathcal{A}$ succeeds in inverting y with some probability $\epsilon(n)$, then there exists a machine M that succeeds in inverting y almost as well as $\mathcal{A}$ (and with comparable number of oracle-queries), but without producing any hits. Formally, in Section 3.5.1 we prove the following Claim:

Claim 3.22. *For every q-query algorithm $\mathcal{A}$, if:*

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n} \left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y) \right] \geq 1/q(n)$$

for infinitely many n's, then there exists a $(3q^3, \infty, q, q)$-query algorithm M, for which it holds that:

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n} \left[M^{\Gamma}(y) \in f_n^{-1}(y) \wedge \overline{\mathsf{CollHit}_y} \right] \geq 1/q(n)^5$$

for infinitely many n's.

In the second part of the proof, we consider the case that the event $\mathsf{CollHit}_y$ does not occur, and prove that if $\mathcal{A}$ manages to invert f_n on a relatively large set of images, then f_n has a short representation given $\mathcal{A}$. This enable us to prove the following claim, which is proved in Section 3.5.2:

Claim 3.23. *For every (Q, ∞, Q, Q)-query algorithm $\mathcal{A}^\Gamma$ with $Q(n) \le 2^{n/9}$, it holds that:*

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n}\left[\mathcal{A}^\Gamma(y) \in f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y}\right] \le 2^{-n/10} \ .$$

Giving these two claim, we prove Theorem 3.20.

Proof of Theorem 3.20. Assume towards contradiction that there exists a q-query algorithm $\mathcal{A}$ with $q(n) = 2^{n/50}$, and that:

$$\Pr_{\Gamma, y \leftarrow \{0,1\}^n}\left[\mathcal{A}^\Gamma(y) = f_n^{-1}(y)\right] > 2^{-n/50}$$

for infinitely many values of n. By Claim 3.22, this implies that there exists a $(3q^3, \infty, q, q)$-query algorithm M, where $3q(n)^3 = 3 \cdot 2^{3n/50} \le 2^{n/9}$, such that:

$$\Pr_{\Gamma, y \leftarrow \{0,1\}^n}\left[M^\Gamma(y) = f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y} \ \right] > \left(2^{-n/50}\right)^5 = 2^{-n/10} \ ,$$

for infinitely many n's – which is a contradiction to Claim 3.23. ■

3.5.1 Avoiding Hits

We prove the following claim:

Claim 3.24. *For every q-query algorithm $\mathcal{A}$, if:*

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n}\left[\mathcal{A}^\Gamma(y) = f_n^{-1}(y)\right] \ge 1/q(n)$$

for infinitely many n's, then there exists a $(3q^3, \infty, q, q)$-query algorithm M, for which it holds that:

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n}\left[M^\Gamma(y) \in f_n^{-1}(y) \wedge \ \overline{\mathsf{CollHit}_y}\right] \ge 1/q(n)^5 \ .$$

Proof. We start with a description of the algorithm M. The algorithm M follows the computation of $\mathcal{A}$ with the following exception: whenever $\mathcal{A}$ makes a $\mathsf{CollFinder}$-query with some circuit C_i (which may cause a y-hit), $\mathcal{A}$ first tries to make this hit by itself by evaluating $C_i^f(z)$ for some random z, and without submitting the query to $\mathsf{CollFinder}$. Only if y-hit does not occur while evaluating $C_i^f(z)$, it passes the query to $\mathsf{CollFinder}$. The analysis shows that if some query C_i produces a y-hit in the execution of $\mathcal{A}$ with some noticeable probability, then also the evaluation of $C_i^f(z)$ on some random z in the execution of M, and therefore M may avoid the event $\mathsf{CollHit}_y$.

The algorithm M. On input $y \in \{0,1\}^n$ the algorithm M^Γ invokes $\mathcal{A}$ on y. With each oracle query to f, Eval or $\mathcal{O}$, the algorithm M just delivers the queries to Γ and returns the responses. On a query C of size at most $q(n)$ to the oracle $\mathsf{CollFinder}$, M proceeds as follows:

1. M first chooses a random value $z \in \{0,1\}^m$ and locally evaluates $C^{f,\mathcal{O},\mathsf{Eval}}(z)$. Recall that $C^{f,\mathcal{O},\mathsf{Eval}}$ may contain f,$\mathcal{O}$ and Eval-gates. While evaluating an f gate, M queries its f-oracle and learns the output of that f-gate. If this output is y, it halts and returns the input to that gate. While evaluating an Eval-gate, on input $(\widehat{D}, x)$ for which $|\widehat{D}| = 2m$, M looks for a pair (D, r) with $|D| = |r| = m$ and evaluates the circuit $D^f(x)$. If some f_n-gate during the computation of $D^f(x)$ contains output y, then M halts and outputs the input for that gate.

2. If M has not yet halted, then it submits C to its $\mathsf{CollFinder}$-oracle and delivers the result (w, w') back to $\mathcal{A}$.

Finally, if M did not halt before the termination of $\mathcal{A}$'s computation, then it outputs $\mathcal{A}$'s output and halts.

It is easy to see that M does not make any additional $\mathsf{CollFinder}$-queries other than those made by $\mathcal{A}$. Moreover, M may perform at most $q(n)$ queries to f as a direct queries of $\mathcal{A}$ to f, and at most $q(n)^3$ queries to f as queries of $\mathcal{A}$ to $\mathsf{CollFinder}$ (specifically, each $\mathsf{CollFinder}$-query may involve $q(n)$ gates of type Eval, where each such a gate may cause to $q(n)$ evaluations of f) . Overall, it makes no more than $3q(n)^3$ queries to f_n. Since M inverts the oracle $\mathcal{O}$ in order to be able to evaluate the Eval-gates by itself, it may make unbounded number of queries to $\mathcal{O}$.

Note that if $\mathcal{A}$ inverts y without producing any $\mathsf{CollHit}_y$ in its queries, then so does M. Formally, if

$$\Pr_{f_n,\mathcal{R},y\leftarrow\{0,1\}^n}\left[\mathcal{A}(y) = f_n^{-1}(y)\ \wedge\ \overline{\mathsf{CollHit}_y}\right] \geq \frac{1}{2q(n)}\ ,$$

then

$$\Pr_{f_n,\mathcal{R},y\leftarrow\{0,1\}^n}\left[M(y) \in f_n^{-1}(y)\ \wedge\ \overline{\mathsf{CollHit}_y}\right] \geq \frac{1}{2q(n)} \geq \frac{1}{q(n)^5}$$

and the claim follows. We therefore focus on the case where the above does not occur, and thus, we assume:

$$\Pr_{f_n,\mathcal{R},y\leftarrow\{0,1\}^n}\left[\mathcal{A}(y) = f_n^{-1}(y)\ \wedge\ \mathsf{CollHit}_y\right] \geq \frac{1}{2q(n)}\ .$$

We will prove the following Lemma below:

Lemma 3.25. *For every choice of $f, \mathcal{O}$ and for every $y \in \{0,1\}^n$, if*

$$\Pr_{\mathcal{R}}\left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y)\ \wedge\ \mathsf{CollHit}_y\right] \geq \frac{1}{8q(n)} \tag{3.6}$$

then

$$\Pr_{\mathcal{R}}\left[M^{\Gamma}(y) = f_n^{-1}(y)\ \wedge\ \overline{\mathsf{CollHit}_y}\right] \geq \frac{1}{1024q(n)^3} \tag{3.7}$$

This Lemma will be proven below, and it helps us to conclude the proof via a standard averaging argument. Let

$$T = \left\{(y, f) \mid \Pr_{\mathcal{R}}\left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y)\ \wedge\ \mathsf{CollHit}_y\right] \geq \frac{1}{8q(n)}\right\}\ .$$

Then, $\Pr_{y,f_n}\left[(y, f_n) \in T\right] \geq 1/8q(n)$, and Lemma 3.25 implies that for every such $(y, f_n) \in T$ we have that Eq. (3.7) follows. This implies that:

$$\begin{aligned}
&\Pr_{f_n,\mathcal{R},y\leftarrow\{0,1\}^n}\left[M^{\Gamma}(y) = f_n^{-1}(y)\ \wedge\ \overline{\mathsf{CollHit}_y}\ \right] \\
&\quad\geq\ \Pr_{f_n,y}\left[(f_n, y) \in T\right] \cdot \Pr_{\mathcal{R}}\left[M^{\Gamma}(y) \in f_n^{-1}(y)\ \wedge\ \overline{\mathsf{CollHit}_y}\right] \\
&\quad\geq\ \frac{1}{8q(n)} \cdot \frac{1}{1024q(n)^3} \geq \frac{1}{q(n)^5}\ .
\end{aligned}$$

■

Proof of Lemma 3.25. The proof shows that if $\mathsf{CollHit}_y$ occurs, then from the fact that $\mathsf{CollFinder}$, on a query C first evaluates $C^{f,\mathcal{O},\mathsf{Eval}}(w)$ on some random $w \in \{0,1\}^m$, and that M first evaluates $C^{f,\mathcal{O},\mathsf{Eval}}(z)$ for some random $z \in \{0,1\}^m$, then there is a good possibility that M will find y-hit as well. Fix $y, f, \mathcal{O}$.

- We denote by $C_1, \ldots, C_q$ the random variables corresponding to $\mathcal{A}$'s $\mathsf{CollFinder}$-queries. In addition, we denote by $(w_1, w_1'), \ldots, (w_q, w_q')$ the random variables corresponding to the answers returned by $\mathsf{CollFinder}$-oracle.
- Given a circuit $C^{f,\mathcal{O},\mathsf{Eval}}$ and an input w, we say that w produces a (C,x)-hit if some f_n-gate has input x. Note that such a f_n-gate may be an explicit gate in C, or may be an implicit gate in some circuit D as part of an Eval-gate.
- For every $1 \leq i \leq q$ let α_i denote the probability that w_i produces a (C_i, x)-hit. That is: $\alpha_i \overset{\text{def}}{=} \Pr_{\mathcal{R}}[w_i \text{ produces a } (C_i,x)\text{−hit}]$. Note that this is the same probability as w_i' produces a (C_i, x) hit, since each of w_i and w_i' is uniformly distributed.
- For every $1 \leq i \leq q$, we denote by JUMP_i the event that $\alpha_i > 1/(32q^2)$ and let $\mathsf{JUMP} = \cup_{i=1}^{q} \mathsf{JUMP}_i$.

We have that:

$$\Pr_{\mathcal{R}}\left[M^{\Gamma}(y) \in f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y}\right] \geq \Pr_{\mathcal{R}}[\mathsf{JUMP}] \cdot \Pr_{\mathcal{R}}\left[M^{\Gamma}(y) = f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y} \mid \mathsf{JUMP}\right] \cdot \qquad (3.8)$$

We now prove lower bounds for these two terms. We start with bounding the probability that JUMP occurs. First, if JUMP does not occur, then we claim that the α_i's are too small in order to produce a y-hit with noticeable probability. That is:

Claim 3.26. $\Pr_{\mathcal{R}}\left[\mathsf{CollHit}_y \mid \overline{\mathsf{JUMP}}\right] \leq 1/(16q)$.

Proof. Assuming that JUMP does not occur, that is $\alpha_i \leq 1/(32q^2)$ for every $1 \leq i \leq q$, it holds that:

$$\Pr_{\mathcal{R}}\left[\mathsf{CollHit}_y \mid \overline{\mathsf{JUMP}}\right] \leq \sum_{i=1}^{q} \Pr_{\mathcal{R}}[w_i \text{ or } w_i' \text{ produce a } (C_i,x)\text{-hit}] \leq 2 \cdot q \cdot \frac{1}{32q^2} = \frac{1}{16q} \ .$$

■

Eq. (3.6) implies that in particular:

$$\Pr_{\mathcal{R}}[\mathsf{CollHit}_y] \geq \frac{1}{8q}$$

In addition:

$$\Pr_{\mathcal{R}}[\mathsf{CollHit}_y] \quad \leq \quad \Pr_{\mathcal{R}}[\mathsf{JUMP}] + \Pr_{\mathcal{R}}\left[\mathsf{CollHit}_y \mid \overline{\mathsf{JUMP}}\right] \leq \Pr_{\mathcal{R}}[\mathsf{JUMP}] + \frac{1}{16q}$$

Thus,

$$\Pr_{\mathcal{R}}[\mathsf{JUMP}] \geq \frac{1}{8q} - \frac{1}{16q} = \frac{1}{16q} \ .$$

Assume now that the event JUMP occurs, and denote by i^* the minimal $1 \leq i \leq q$ for which JUMP_i occurs. When $\mathcal{A}$ submits the query C_{i^*}, then M has probability α_{i^*} to retrieve x without submitting the query to $\mathsf{CollFinder}$. In addition, since i^* is the minimal $1 \leq i \leq q$ for which JUMP_i occurs, then with high probability $\mathsf{CollFinder}$'s answers to $C_1, \ldots, C_{i^*-1}$ do not produce an hit. We consider the following events:

- None of the queries $C_1, \dots, C_{i^*-1}$ produces a y-hit. Since for every such C_i the event JUMP_i does not occur, then, the probability of this event is at least $1 - 1/(16q)$, exactly as in the proof of Claim 3.26.
- Given C_{i^*}, M samples a random z which produces a (C_{i^*}, x)-hit. Since JUMP_{i^*} occurs, the probability of this event is $\alpha_{i^*} > 1/(32q^2)$.

Note that these two events are independent (since the permutations in $\mathcal{R}$ are chosen independently). Putting these together, we obtain:

$$\Pr_{\mathcal{R}}\left[M^\Gamma(y) = f_n^{-1}(y) \ \wedge\ \overline{\mathsf{CollHit}_y} \mid \mathsf{JUMP}\right] \geq \left(1 - \frac{1}{16q}\right) \cdot \frac{1}{32q^2} \geq \frac{1}{64q^2}$$

Plugging into Eq. (3.8), we get:

$$\Pr_{\mathcal{R}}\left[M^\Gamma(y) \in f_n^{-1}(y) \ \wedge\ \overline{\mathsf{CollHit}_y}\right] \geq \frac{1}{64q^2} \cdot \frac{1}{16q} = \frac{1}{1024q^3}$$

■

3.5.2 From Inverting to Compressing

In this section we show that if $\mathcal{A}$ succeeds to invert f_n with high probably (that is, it succeeds to invert many y's) then the function f_n has a short representation given $\mathcal{A}$. In general a random permutation f_n can be represented using $\log(2^n!)$ bits. The proof shows a different representation of f_n which consists of three parts: some partial truth table, a set of pre-images X and a the corresponding set of images Y. Note that we do not store what pre-image maps to what image, and this is the core of the saving in this representation. The mapping between these two sets can be reconstructed by following the computation of $\mathcal{A}$. We show that:

Claim 3.27. *For every algorithm (q, ∞, q, q)-query algorithm $\mathcal{A}$ with $q(n) = 2^{n/9}$, it holds that:*

$$\Pr_{f_n, \mathcal{R}, y \leftarrow \{0,1\}^n}\left[\mathcal{A}^\Gamma(y) \in f_n^{-1}(y) \ \wedge\ \overline{\mathsf{CollHit}_y}\right] \leq 2^{-n/10} .$$

Proof. Fix $\mathcal{R}$ and recall that we already fixed $\mathcal{O}, f_{-n}$ (that is, all the oracles are fixed except for f_n). We show that for every f_n and $\mathcal{A}$ as above, if

$$\Pr_{y \leftarrow \{0,1\}^n}\left[\mathcal{A}^\Gamma(y) = f_n^{-1}(y) \ \wedge\ \overline{\mathsf{CollHit}_y}\right] \geq \epsilon$$

then there is a succinct description of the function f_n.

Succinct description of f_n. We show that f_n can be represented using some small partial truth table Z, some set of pre-images X and some set of images Y. We start by building the set Y.

Denote by $I \subseteq \{0,1\}^n$ the set of points $y \in \{0,1\}^n$ for which $\mathcal{A}$ successfully inverts f_n with no $\mathsf{CollFinder}$-hits. We claim that there exists a relatively large set $Y \subseteq I$, such that the value of f_n^{-1} on the set Y is determined by $\mathcal{A}, \mathcal{O}, \mathcal{R}$, f_{-n} and the partial truth table Z (i.e., the value of f_n on $\{0,1\}^n \setminus X$).

We define the set Y via the following sequential process. Initially, Y is empty and we remove the lexicographically smallest element y from I and insert it into Y. Then, we follow the computation of $\mathcal{A}^\Gamma(y)$ and construct a set $F_y = \{y_1, \dots, y_k\}$ of all images that are necessary for this computation (i.e., all the outputs of all f_n-gates during this computation). These values are defined as follows:

- At most $q(n)$ outputs which are direct f_n-gates in $\mathcal{A}$. That is, for each direct f_n-gate in $\mathcal{A}$, we add the output the gate to the set of images F_y.
- At most $q(n) \cdot q(n)$ outputs which are results of Eval-gates. In particular, on a query $(\widehat{D}, x)$ with $|\widehat{D}| = 2m$, the Eval-oracle looks for a pair (D, r) with $|D| = |r| = m$ for which $\mathcal{O}_m(D, r) = \widehat{D}$. Since $|D| = m \leq q(n)$, the circuit D contains at most $q(n)$ gates of type f_n. We follow the computation of $D^f(x)$, and add all the outputs of these f_n gates to F_y.
- At most $2 \cdot (q(n))^3$ outputs as a result of $\mathsf{CollFinder}$-gates. In particular, let $(C_1, \ldots, C_q)$ denote the queries made by $\mathcal{A}$ to $\mathsf{CollFinder}$-oracle, and let $(w_1, w_1'), \ldots, (w_q, w_q')$ the corresponding answers. We follow the computations of both $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$ and $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$. Each one of them is a circuit of size $q(n)$ and may contain some direct f_n gates, and at most $q(n)$ gates of type Eval, where each may consist of at most $q(n)$ additional f_n-gates, exactly as above. We add all the outpost of all f_n-gates that are involved in this computation to the set F_y.

From the above, $|F_y| \leq 4(q(n))^3$. We then remove from I the set F_y (note that these values are not necessarily in the set I). We now continue to the next iteration, where we remove the lexicographically smallest element y' for the remaining elements of I, insert it into Y, follow the computation of $\mathcal{A}^{\Gamma}(y')$, define $F_{y'}$ and continue in the same manner until the set I is emptied.

On the size of the set Y. In each iteration one element is inserted into Y, and at most $4(q(n))^3$ elements are removed from the set I. Since the set I initially contains at least $\epsilon \cdot 2^n$ elements, then when the process terminates we have that $|Y| \geq \epsilon \cdot 2^n / 4(q(n))^3$.

Now, let $X = f_n^{-1}(Y) = \{x \in \{0,1\}^n \mid f_n(x) \in Y\}$. We now represent the function f_n using the sets X, Y, and the partial truth table of f_n^{-1} on the set $\overline{Y} = \{0,1\}^n \setminus Y$, which we denote by Z. This description of f_n requires at most $2 \log \binom{2^n}{|Y|} + \log((2^n - |Y|)!)$ bits.

Reconstruction. We now claim that f_n can be completely reconstructed given X, Y, Z, $\mathcal{A}$, f_{-n}, $\mathcal{O}$ and $\mathcal{R}$. We present a simulator that reconstructs the entire truth table of f_n^{-1}. For each $y \in Y$, taken in lexicographical increasing order, the simulator reconstructs $x = f_n^{-1}(y)$. These, together with the partial truth table Z, determine the entire truth table of f_n^{-1}. For each $y \in Y$, in order to reconstruct $f_n^{-1}(y)$, the simulator follows the computation of $\mathcal{A}^{\Gamma}(y)$, answers all the necessary f_n-gates of $\mathcal{A}$ and learn $f_n^{-1}(y)$.

We now show how the simulator can answer all the necessary queries in the computation of $\mathcal{A}^{\Gamma}(y)$. Answering all queries of f_{-n} and $\mathcal{O}$ is easy, since these oracles are explicitly given. We show how it can answer all f_n-queries, Eval and $\mathsf{CollFinder}$-queries.

Answering f_n-query. We start by showing how it can answer any direct f_n query of $\mathcal{A}$. Whenever $\mathcal{A}$ asks for value $f_n(x)$ for some x, the simulator acts as follows: if this value is already known (was either reconstructed earlier or is in Z), then the simulator can return it. Otherwise, if the value is not known, we claim that it must be that $f_n(x) = y$ and in this case the simulator outputs y and continues to the next iteration. This is because the following cases covers all possibilities:

1. If $x \notin X$, then $f_n(x)$ is explicitly known (using Z).
2. If $x \in X$ and $f_n(x) <_{lex} y$ then the simulator has already reconstructed $f_n(x)$.
3. If $x \in X$ and $f_n(x) >_{lex} y$ then this case is impossible. We have that both $y \in Y$ and $f_n(x) \in Y$, but y was inserted to Y before $f_n(x)$ was inserted to Y. Therefore, $f_n(x)$ should have been removed from I, and in particular not inserted into Y.
4. If $x \in X$ and $f_n(x) = y$.

Answering Eval-queries. Similarly to the above, the simulator can answer any Eval query made by $\mathcal{A}$. In particular, given an input $(\widehat{D}, x)$, with $|\widehat{D}| = 2m$, the simulator looks for a pair (D, r) for which $\mathcal{O}_m(D, r) = \widehat{D}$. Then, it follows the computation of $D^f(x)$, and answers each f_n-query during this computation exactly as above.

Answering CollFinder-queries. We now consider the case where $\mathcal{A}$ makes $\mathsf{CollFinder}$-query. On input oracle-aided circuit C_i, the simulator computes $w_i = \pi_1^C(0)$ and begins with evaluating $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$. Note that this computation may involve some additional f_n and Eval gates, which we claim that these can be handled and discuss it below. Then, it enumerates over all $t \in \{0,1\}^m$ in lexicographically increasing order, and stops with the first $w_i' = \pi_2^C(t)$ for which $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$ can be computed (i.e., it is able to answer all f_n-queries as discussed below) and that $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i) = C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$. It then answers to $\mathcal{A}$ with the pair (w_i, w_i').

We now show that the simulator can evaluate $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i), C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$ and answers the queries to f_n. Note that queries to f_n may be as a direct gates in $C_i^{f,\mathcal{O},\mathsf{Eval}}$ or be some gates in circuits that are evaluated as byproduct of some Eval-gate. On a query x to f_n-gate, we have:

1. If $x \notin X$, then the value $f_n(x)$ is given by Z.
2. If $x \in X$ and $f_n(x) <_{lex} y$ then the simulator already reconstructed $f_n(x)$.
3. If $x \in X$ and $f_n(x) >_{lex} y$ then this case is impossible (as above).
4. If $x \in X$ and $f_n(x) = y$, then this is impossible, as we condition on the event that $\mathsf{CollHit}_y$ does not occur.

We remark that when the simulator evaluates $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i)$ and $C_i^{f,\mathcal{O},\mathsf{Eval}}(w_i')$, all the f_n-gates that may appear in this computation are of case 1 and 2 only. In contrast, when it enumerates over all $t \in \{0,1\}^m$ and compute $C_i(\pi_2^C(t))$, it may reach f_n-gates for which cases 3 or 4 occur. In such a case, whenever it is unable to reply to an f_n-gate, it proceeds to the next value t.

Concluding the proof. As discussed above $|Y| \geq \epsilon \cdot 2^n/4(q(n))^3$, and our representation of f_n requires $2\log\binom{2^n}{|Y|} + \log((2^n - |Y|)!) = \log\left(\binom{2^n}{|Y|}^2 \cdot (2^n - |Y|)!\right)$ bits. Since there are $(2^n)!$ permutations over $\{0,1\}^n$, the fraction of permutations f_n over $\{0,1\}^n$ for which:

$$\Pr_{y \leftarrow \{0,1\}^n}\left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y}\right] \geq \epsilon ,$$

is at most:

$$\frac{\binom{2^n}{|Y|}^2 \cdot (2^n - |Y|)!}{(2^n)!} = \frac{\binom{2^n}{|Y|}}{|Y|!} \leq \left(\frac{2^n \cdot e}{|Y|}\right)^{|Y|} \cdot \left(\frac{e}{|Y|}\right)^{|Y|} = \left(\frac{2^n e^2}{|Y|^2}\right)^{|Y|} .$$

where the inequality uses the inequalities $a! \geq (a/e)^a$ and $\binom{2^n}{a} \leq (2^n e/a)^a$. Taking $\epsilon = 2^{-n/9}$, $q(n) \leq 2^{n/9}$, and recall that $|Y| \geq \epsilon \cdot 2^n/4(q(n))^3$, we get:

$$|Y| \geq \epsilon \cdot \frac{2^n}{4(q(n))^3} \geq \frac{2^n}{4 \cdot 2^{4n/9}} = 2^{5n/9-2} ,$$

thus, for sufficiently large n's the above is upper bound by:

$$\left(\frac{2^n e^2}{|Y|^2}\right)^{|Y|} \leq \left(\frac{16 \cdot 2^n \cdot e^2}{2^{10n/9}}\right)^{|Y|} \leq \left(\frac{16 \cdot e^2}{2^{n/9}}\right)^{|Y|} \leq 2^{-|Y|} \leq 2^{-2^{5n/9-2}} .$$

We conclude that for any algorithm $\mathcal{A}$ that makes at most $q(n) = 2^{n/9}$ queries to $(f, \mathcal{O}, \mathsf{Eval}, \mathsf{CollFinder})$, and its oracle queries to $\mathsf{CollFinder}$ are bounded to circuits of size $q(n)$, it holds that:

$$\Pr_{f_n, y \leftarrow \{0,1\}^n} \left[\mathcal{A}^{\Gamma}(y) = f_n^{-1}(y) \ \wedge \ \overline{\mathsf{CollHit}_y} \right] \leq 2^{-n/9} + 2^{-2^{5n/9}-2} \leq 2^{-n/10} \ .$$

■

3.5.3 Extension to Trapdoor Permutations

Our negative result on the power of indistinguishability obfuscation holds even when replacing the permutation f with a trapdoor permutation family τ. Our proofs in Sections 3.3 and 3.4 are not affected by this generalization[13], and here we provide the main ideas for showing that τ is one way relative to Γ. This generalization is a natural generalization to that of Haitner et al. [HHR+15].

We generalize our result by considering the oracle $\Gamma' = \left(\tau, \mathcal{O}, \mathsf{Eval}^{\tau,\mathcal{O}}, \mathsf{CollFinder}^{\tau,\mathcal{O},\mathsf{Eval},\mathcal{R}}\right)$, where $\mathcal{O}$, Eval and $\mathsf{CollFinder}$ are as defined in Section 3.2, and $\tau = \{\tau_n\}_{n \in \mathbb{N}}$ is defined as follows. For every $n \in \mathbb{N}$, the function family τ_n is a triplet (G_n, F_n, F_n^{-1}), where:

- The function G_n is a uniformly chosen permutation over $\{0,1\}^n$.
- For every $pk \in \{0,1\}^n$, the function $F_n(pk, \cdot)$ is a uniformly chosen permutation over $\{0,1\}^n$.
- For every $td \in \{0,1\}^n$ and $y \in \{0,1\}^n$ we define $F_n^{-1}(td, y) = x$ for the unique $x \in \{0,1\}^n$ that satisfies $F_n(pk, x) = y$ where $pk = G_n(td)$.

By relying on our proof showing that a random permutation f is one way relative to Γ, we obtain the following theorem:

Theorem 3.28 (Simplified variant). *For any probabilistic oracle-aided algorithm $\mathcal{A}$ that runs in time $2^{O(n)}$ and for any function $\mathcal{O}$, it holds that*

$$\Pr_{\substack{\tau, \mathcal{R} \\ pk, x \leftarrow \{0,1\}^n}} \left[\mathcal{A}^{\Gamma'}(pk, F(pk, x)) = x \right] \leq 2^{-\Omega(n)}.$$

Proof (sketch). Given $\mathcal{A}$ and pk we let $td = G^{-1}(pk)$ and denote by tdHit the event in which at least one of the following occur:

1. $\mathcal{A}$ queries G with td.
2. $\mathcal{A}$ queries Eval with some $(\widehat{D}, a)$ for which the following hold:
   (a) Let (D, r) be the pair such that $|r| = |D| = m$ and $\mathcal{O}_m(D, r) = \widehat{D}$.
   (b) Some G-gate in the computation of $D^{\tau}(a)$ has input td.
3. $\mathcal{A}$ queries $\mathsf{CollFinder}$ with a circuit C and obtains a pair (w, w') such that at least one of the following occur:
   (a) Some G-gate in the computation of $C^{\tau}(w)$ or $C^{\tau}(w')$ has input td.
   (b) Some Eval-gate in the computation of $C^{\tau}(w)$ or $C^{\tau}(w')$ has input $(\widehat{D}, a)$ for which the following holds:
      i. Let (D, r) be the pair such that $|r| = |D| = m$ and $\mathcal{O}_m(D, r) = \widehat{D}$.
      ii. Some G-gate in the computation of $D^{\tau}(a)$ has input td.

[13] These proofs hold for any fixing of f, and similarly for any fixing of τ without making any non-trivial adjustments.

We now consider two cases, depending on whether or not the event tdHit occurs. If $\mathcal{A}$ is successful in outputting x and the event tdHit occurs, then we can use $\mathcal{A}$ to invert the uniformly sampled trapdoor permutation G_n on pk. If $\mathcal{A}$ is successful in outputting x and the event tdHit does not occur, then the function $F^{-1}(td, \cdot)$ is essentially useless for $\mathcal{A}$, which is able to invert a uniformly sampled trapdoor permutation $F(pk, \cdot)$. In both cases, we can apply our result stating that a uniformly sampled permutation is one way relative to Γ, and the theorem follows. We refer the reader to [HHR+15] for more details. ■

4 Limits on the Power of Private-Key Functional Encryption

In this section we present our negative result for constructing a perfectly-complete key-agreement protocol from a general-purpose private-key functional encryption scheme and a one-way permutation. First, in Section 4.1 we formally define the class of constructions to which our negative result applies. Then, in Section 4.2 we present the structure of our proof, which is then provided in Sections 4.3–4.5. Finally, in Section 4.6 we show that our result can be extended for separating indistinguishability obfuscation for oracle-aided circuits from private-key functional encryption for oracle-aided circuits.

4.1 The Class of Reductions

We consider fully black-box constructions of a perfectly-complete bit-agreement protocol from a general-purpose private-key functional encryption scheme and a one-way permutation. Similarly to our approach from Section 3, we model these primitives as two independent building blocks due to the following two reasons. First, although any private-key functional encryption scheme clearly implies the existence of a one-way function, it is not known whether any such scheme implies the existence of a one-way permutation. Second, this enables us to capture constructions that may use the underlying functional encryption scheme for generating functional keys to any circuit that can be constructed in a fully black-box manner from a one-way permutation. For example, as in Section 3, this enables us to capture constructions that may generate functional keys to any circuit that uses a puncturable pseudorandom function or a pseudorandom generator as a sub-routine.

We now formally define the class of constructions considered in this section, tailoring our definitions to the specific primitives under consideration. We consider key-agreement protocols in which the parties agree on a single bit, and we refer to such protocols as bit-agreement protocols. We consider any implementation of a one-way permutation f and a private-key functional encryption scheme Π for the class of all polynomial-size oracle-aided circuits C^f.

Definition 4.1. *A fully black-box construction of a perfectly-correct bit-agreement protocol from a one-way permutation and a private-key functional encryption scheme for the class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of all polynomial-size oracle-aided circuits consists of a pair of probabilistic polynomial-time oracle-aided algorithms $(\mathcal{A}, \mathcal{B})$, an oracle-aided algorithm M that runs in time $T_M(\cdot)$, and functions $\epsilon_{M,1}(\cdot)$ and $\epsilon_{M,2}(\cdot)$, such that the following two conditions hold:*

- **Correctness:** *For any $n \in \mathbb{N}$, for any permutation f and for any correct private-key functional encryption scheme Π, it holds that*

$$\Pr\left[k_{\mathcal{A}} = k_{\mathcal{B}} \middle| (k_{\mathcal{A}}, k_{\mathcal{B}}, T) \leftarrow \left\langle \mathcal{A}^{f,\Pi}(1^n), \mathcal{B}^{f,\Pi}(1^n) \right\rangle \right] = 1,$$

  where the probability is taken over the internal randomness of $\mathcal{A}$ and $\mathcal{B}$.

- **Black-box proof of security:** *For any permutation f, for any correct private-key functional encryption scheme Π, for any probabilistic oracle-aided algorithm E that runs in time $T_E = T_E(n)$, and for any function $\epsilon_E = \epsilon_E(n)$, if*

$$\left| \Pr \left[\mathsf{Exp}^{\mathsf{KA}}_{(f,\Pi),(\mathcal{A},\mathcal{B}),E}(n) = 1 \right] - \frac{1}{2} \right| \geq \epsilon_E(n)$$

for infinitely many values of n (see Definition 2.6 for the description of the experiment $\mathsf{Exp}^{\mathsf{KA}}_{(f,\Pi),(\mathcal{A},\mathcal{B}),E}$), then either

$$\Pr_{x \leftarrow \{0,1\}^n} \left[M^{f,\Pi,E}(f(x)) = x \right] \geq \epsilon_{M,1}\left(T_E(n) \cdot \epsilon_E^{-1}(n) \right) \cdot \epsilon_{M,2}(n)$$

or

$$\left| \Pr \left[\mathsf{Exp}^{\mathsf{FE}}_{(f,\Pi),\Pi,M^E,\mathcal{C}}(n) = 1 \right] - \frac{1}{2} \right| \geq \epsilon_{M_1}\left(T_E(n) \cdot \epsilon_E^{-1}(n) \right) \cdot \epsilon_{M,2}(n)$$

for infinitely many values of n (see Definition 2.4 for the description of the experiment $\mathsf{Exp}^{\mathsf{FE}}_{(f,\Pi),\Pi,M^E,\mathcal{C}}$).

Similarly to the discussion in Section 3.1, we emphasize the fact that our definition captures an underlying functional encryption scheme Π for the class of all polynomial-size oracle-aided circuits C^f, and thus *black-box* access in our setting is only required with respect to the functional encryption scheme Π and the one-way permutation f themselves. We refer the reader to Section 3.1 for a more elaborated discussion of this issue, as well as a discussion on the roles of the "security loss" functions T_M, $\epsilon_{M,1}$ and $\epsilon_{M,2}$.

4.2 Proof Overview and the Oracle Ψ

Our result is obtained by presenting an oracle Ψ relative to which there exist an exponentially-secure one-way permutation f and an exponentially-secure private-key functional encryption scheme Π for the class of all polynomial-size oracle-aided circuits C^f, but any bit-agreement protocol with perfect completeness can be broken. We prove the following theorem:

Theorem 4.2. *Let $(\mathcal{A}, \mathcal{B}, M, T_M, \epsilon_{M,1}, \epsilon_{M,2})$ be a fully black-box construction of a perfectly-correct bit-agreement protocol from a one-way permutation f and a private-key functional encryption scheme Π for the class of all polynomial-size oracle-aided circuits C^f (see Definition 4.1). Then, at least one of the following propertied holds:*

1. *$T_M(n) \geq 2^{\zeta n}$ for some constant $\zeta > 0$ (i.e., the reduction runs in exponential time).*
2. *$\epsilon_{M,1}(n^c) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$ for some constant $c > 1$ (i.e., the security loss is exponential).*

In particular, the theorem implies that if the running time $T_M(\cdot)$ of the reduction is sub-exponential and the adversary-dependent security loss $\epsilon_{M,1}(\cdot)$ is polynomial as in the vast majority of constructions, then the adversary-independent security loss $\epsilon_{M,2}(\cdot)$ must be exponential (thus ruling out even constructions that rely on sub-exponential security assumptions).

In what follows we describe the oracle Ψ and then explain the structure of our proof.

The oracle Ψ. The oracle Ψ is a quadruple $\left(f, \mathcal{K}, \mathcal{E}, \mathcal{D}^{f,\mathcal{K},\mathcal{E}}\right)$ that is defined as follows:

- **The function $f = \{f_n\}_{n \in \mathbb{N}}$.** For every $n \in \mathbb{N}$ the function f_n is a uniformly chosen permutation over $\{0,1\}^n$. Looking ahead, we will prove that f is one way relative to Ψ.
- **The functions $\mathcal{K} = \{\mathcal{K}_n\}_{n \in \mathbb{N}}$ and $\mathcal{E} = \{\mathcal{E}_n\}_{n \in \mathbb{N}}$.** For every $n \in \mathbb{N}$ the functions $\mathcal{K}_n$ and $\mathcal{E}_n$ are uniformly chosen functions $\mathcal{K}_n : \{0,1\}^{2n} \to \{0,1\}^{10n}$ and $\mathcal{E}_n : \{0,1\}^{3n} \to \{0,1\}^{10n}$. Looking ahead, we will use $\mathcal{K}$ and $\mathcal{E}$ for implementing the (deterministic) key-generation algorithm and the (randomized) encryption algorithm, respectively, of the functional encryption scheme Π.
- **The function $\mathcal{D}^{f,\mathcal{K},\mathcal{E}} = \{\mathcal{D}_n^{f_n,\mathcal{K}_n,\mathcal{E}_n}\}_{n \in \mathbb{N}}$.** For every $n \in \mathbb{N}$ the function $\mathcal{D}_n^{f,\mathcal{K},\mathcal{E}} : \{0,1\}^{20n} \to \{0,1\}^n$ parses its input a pair $(\mathsf{sk}, c) \in \{0,1\}^{10n} \times \{0,1\}^{10n}$, and is defined as follows: If there exist $\mathsf{msk}, C, m, r \in \{0,1\}^n$ such that $\mathsf{sk} = \mathcal{K}_n(\mathsf{msk}, C)$ and $c = \mathcal{E}_n(\mathsf{msk}, m, r)$ then it outputs $C^f(m)$ for the lexicographically-first such quadruple, and otherwise it outputs $\perp$. Looking ahead, we will use $\mathcal{D}$ for implementing the decryption algorithm of the functional encryption scheme Π.

Equipped with the oracle Ψ, our proof consists of the following two parts.

Part 1: The existence of a one-way permutation and a functional encryption scheme. We first prove that f is one way relative to Ψ, and this is rather standard at least when observing that each query to $\mathcal{D}$ requires only a bounded (and not too large) number of queries to f. Then, we show that relative to Ψ there exists a functional encryption scheme Π that is naturally defined given the way we set up Ψ (we refer the reader to Section 4.4 for the description of Π). Moreover, the functional encryption scheme Π is even "compact" [AJ15, BV15] in the sense that the efficiency of its encryption algorithm depends only on the security parameter and on length of the plaintext (in particular, it is independent of the complexity of the function family supported by the scheme). Proving that Π is secure is more tricky, since each query to $\mathcal{D}$ may require an exponential number of queries to $\mathcal{K}$ and $\mathcal{E}$. The following are simplified variants of the theorems that we prove in Sections 4.3 and 4.4:

Theorem 4.3 (Simplified variant)**.** *For any oracle-aided algorithm $\mathcal{A}$ that makes at most $2^{n/4}$ oracle queries, and for any functions $\mathcal{K}$ and $\mathcal{E}$, it holds that*

$$\Pr_{\substack{f \\ x \leftarrow \{0,1\}^n}} \left[\mathcal{A}^{\Psi}(f(x)) = x\right] \leq 2^{-n/4}.$$

Theorem 4.4 (Simplified variant)**.** *For any oracle-aided valid adversary $\mathcal{A}$ that makes at most $2^{n/4}$ oracle queries, and for any permutation f, it holds that*

$$\left| \Pr_{\mathcal{K},\mathcal{E}} \left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n) = 1\right] - \frac{1}{2} \right| \leq 2^{-n/4}.$$

Part 2: Breaking any perfectly-complete bit-agreement protocol. The most challenging part in this section is in proving that any perfectly-complete bit-agreement protocol can be broken using a polynomial number of oracle queries. Our proof is inspired by a combination of ideas that were developed in the early work of Impagliazzo and Rudich [IR89] and its improvement by Barak and Mahmoody-Ghidary [BM09]. Specifically, for the case of perfectly-complete bit-agreement protocols, our proof generalizes the approach of Brakerski, Katz, Yerukhimovich and Segev [BKS$^+$11] to the setting of our oracle Ψ. The following is a simplified variant of the theorem that we prove in Section 4.5:

Theorem 4.5 (Simplified variant)**.** *For any polynomial-time oracle-aided perfectly-complete bit-agreement protocol* $(\mathcal{A},\mathcal{B})$*, there exists an oracle-aided algorithm* E *making a polynomial number of oracle queries such that*

$$\left|\Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{KA}}_{\Psi,(\mathcal{A},\mathcal{B}),E}(n)=1\right]-\frac{1}{2}\right|\geq 1/4.$$

Finally, we now show that the above two parts imply Theorem 4.2 via a proof that is essentially identical to that of Theorem 3.2.

Proof of Theorem 4.2. Let $(\mathcal{A},\mathcal{B},M,T_M,\epsilon_{M,1},\epsilon_{M,2})$ be a fully black-box construction of a perfectly-complete bit-agreement protocol from a one-way permutation f and a private-key functional encryption scheme Π for the class $\mathcal{C}$ of all polynomial-size oracle-aided circuits C^f (recall Definition 4.1). Note that in our setting, relative to the oracle Ψ, this means we allow the algorithms $\mathcal{A}$ and $\mathcal{B}$ to access f, $\mathcal{K}$, $\mathcal{E}$ and $\mathcal{D}$ (i.e., to access Ψ).

Theorem 4.5 guarantees the existence of an oracle-aided algorithm E that makes a polynomial number $T_E(n)$ of queries to the oracle Ψ, such that

$$\left|\Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{KA}}_{\Psi,(\mathcal{A},\mathcal{B}),E}(n)=1\right]-\frac{1}{2}\right|\geq \epsilon_E(n),$$

where $\epsilon_E(n)=1/4$ for all values of $n\in\mathbb{N}$. Definition 4.1 then states that there are two possible cases to consider: E can be used either for breaking the functional encryption scheme Π, or for inverting the one-way permutation f.

In the first case, we obtain from Definition 4.1 that

$$\left|\Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,M^E,\mathcal{C}}(n)=1\right]-\frac{1}{2}\right|\geq \epsilon_{M,1}\left(T_E(n)\cdot 4\right)\cdot\epsilon_{M,2}(n),$$

where M runs in time $T_M(n)$. The algorithm M may invoke E on various security parameters (i.e., in general M is not restricted to invoking E only on security parameter n), and we denote by $\ell(n)$ the maximal security parameter on which M invokes E (when M itself is invoked on security parameter n). Thus, viewing M^E as a single algorithm, its number of queries $T_{M^E}(n)$ to the oracle Ψ satisfies $T_{M^E}(n)\leq T_M(n)\cdot T_E(\ell(n))$ (this follows since M runs in time $T_M(n)$ and in each step of its execution it may query Ψ directly at most once or invoke E at most once where each such invocation results in at most $T_E(\ell(n))$ queries to Ψ). Theorem 4.4 then implies that either $2^{n/4}\leq T_{M^E}(n)$ or $\epsilon_{M,1}\left(T_E(n)\cdot 4\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/4}$. In the first sub-case, noting that $\ell(n)\leq T_M(n)$, we obtain that

$$2^{n/4}\leq T_{M^E}(n)\leq T_M(n)\cdot T_E(\ell(n))\leq T_M(n)\cdot T_E(T_M(n)).$$

The number of queries $T_E(n)$ made by the adversary E is some fixed polynomial in n, and therefore $T_M(n)\geq 2^{\zeta n}$ for some constant $\zeta>0$. In the second sub-case, we have that $\epsilon_{M,1}\left(T_E(n)\cdot 4\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/4}$, and since $T_E(n)$ is some fixed polynomial in n we obtain that $\epsilon_{M,1}\left(n^c\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/4}$ for some constant $c>1$.

In the second case, we obtain from Definition 4.1 that

$$\Pr_{\substack{\Psi\\ x\leftarrow\{0,1\}^n}}\left[\left(M^E\right)^{\Psi}(f(x))=x\right]\geq \epsilon_{M,1}\left(T_E(n)\cdot 4\right)\cdot\epsilon_{M,2}(n),$$

where M runs in time $T_M(n)$. As in the first case, viewing M^E as a single algorithm, its number of queries $T_{M^E}(n)$ to the oracle Ψ satisfies $T_{M^E}(n)\leq T_M(n)\cdot T_E(\ell(n))$. Theorem 4.3 then implies that either $2^{n/4}\leq T_{M^E}(n)$ or $\epsilon_{M,1}\left(T_E(n)\cdot 4\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/4}$. As in the first case, this implies that either $T_M(n)\geq 2^{\zeta n}$ for some constant $\zeta>0$ or $\epsilon_{M,1}\left(n^c\right)\cdot\epsilon_{M,2}(n)\leq 2^{-n/4}$ for some constant $c>1$. ■

4.3 f is a One-Way Permutation Relative to Ψ

In this section we prove that f is one way relative to the oracle Ψ. This is a rather standard proof, relying on the fact that each query to $\mathcal{D}_n$ requires at most n queries to f_n. We prove the following claim:

Claim 4.6. *For any oracle-aided algorithm $\mathcal{A}$ making at most $q(n)$ oracle queries it holds that:*

$$\Pr_{\Psi, x \leftarrow \{0,1\}^n} \left[\, \mathcal{A}^{\Psi}(f_n(x)) = x \, \right] \leq \frac{n \cdot q(n)}{2^n - q(n)}$$

Proof. We show that the claim in fact holds for any fixing of $\mathcal{K}$, $\mathcal{E}$, and $f_{-n} = \{f_m\}_{m \in \mathbb{N}, m \neq n}$, where the probability is taken only over the choice of f_n (and over the randomness of $\mathcal{A}$). An execution of a q-query adversary $\mathcal{A}$ can be simulated using an adversary $\mathcal{B}$ that makes at most $n \cdot q(n)$ oracle queries to f_n, an unlimited amount of queries to f_{-n}, $\mathcal{K}$ and $\mathcal{E}$, and no oracle queries to $\mathcal{D}$. Specifically, $\mathcal{B}$ follows the computation of $\mathcal{A}^{\Psi}(f_n(x))$, and responds to its oracle queries as follows:

- Whenever $\mathcal{A}$ queries f,$\mathcal{K}$ or $\mathcal{E}$, the algorithm $\mathcal{B}$ simply forwards the query and delivers back the result.
- Whenever $\mathcal{A}$ queries $\mathcal{D}$ with some input $(\mathsf{sk}_C, c) \in \{0,1\}^{10s} \times \{0,1\}^{10s}$, for some $s \in \mathbb{N}$, the algorithm $\mathcal{B}$ can enumerate over all possible $\mathsf{msk}, C, m, r \in \{0,1\}^s$ (by lexicographic increasing order) and check whether $\mathsf{sk}_C = \mathcal{K}_s(\mathsf{msk}, C)$ and $c = \mathcal{E}_s(\mathsf{msk}, m, r)$. If so, it evaluates the circuit $C^f(m)$, which may lead to additional s queries of f, and returns the result to $\mathcal{A}$. Otherwise, it returns $\perp$ to $\mathcal{A}$. Note that if $s \neq n$, this does not require any queries to f_n. In addition, if $s = n$, this requires at most n queries to f_n.

Since $\mathcal{A}$ makes at most $q(n)$ queries to f_n and to $\mathcal{D}$, then $\mathcal{B}$ makes at most $n \cdot q(n)$ queries to f_n. Since f_n is a random permutation, any such $\mathcal{B}$ can invert $f_n(x)$ with probability at most $n \cdot q(n) / \left(2^n - q(n)\right)$. ■

4.4 Π is a Functional Encryption Scheme Relative to Ψ

In this section we show that relative to the oracle Ψ there exists a private-key function encryption scheme Π for the class $\mathcal{C}$ of all polynomial-time oracle-aided circuits C^f. We first describe the scheme and then prove its security.

Construction 4.7 (Private-key functional encryption.)**.** *The private-key functional encryption scheme* $\Pi^{\Psi} = \left(\mathsf{Setup}, \mathsf{KG}^{\mathcal{K}}, \mathsf{Enc}^{\mathcal{E}}, \mathsf{Dec}^{\mathcal{D}^{f,\mathcal{K},\mathcal{E}}}\right)$ *is defined as follows:*

- **Setup.** *The setup algorithm* Setup *on input* 1^n *samples and outputs* $\mathsf{msk} \leftarrow \{0,1\}^n$.
- **key generation.** *The key-generation algorithm* $\mathsf{KG}^{\mathcal{K}}$ *on input a master secret key* $\mathsf{msk} \in \{0,1\}^n$ *and description of an oracle-aided circuit* $C \in \{0,1\}^n$ *outputs* $\mathsf{sk}_C = \mathcal{K}_n(\mathsf{msk}, C)$.
- **Encryption.** *The encryption algorithm* $\mathsf{Enc}^{\mathcal{E}}$ *on input a master secret key* $\mathsf{msk} \in \{0,1\}^n$ *and a message* $m \in \{0,1\}^n$*, samples* $r \leftarrow \{0,1\}^n$*, and outputs* $c = \mathcal{E}_n(\mathsf{msk}, m, r)$.
- **Decryption.** *The decryption algorithm* $\mathsf{Dec}^{\mathcal{D}^{f,\mathcal{K},\mathcal{E}}}$ *on input a functional key* sk *and a ciphertext* c *outputs* $\mathcal{D}^{f,\mathcal{K},\mathcal{E}}(\mathsf{sk}, c) \in \{0,1\}^n \cup \{\perp\}$.

Correctness. Assume that the functions $\mathcal{K}_n$ and $\mathcal{E}_n$ are injective (this occurs with all but an exponentially-small probability), for any oracle-aided function $C \in \mathcal{C}_n$ for any message $m \in \{0,1\}^n$, the definition of the oracle $\mathcal{D}$ guarantees that

$$\mathsf{Dec}(\mathsf{KG}(\mathsf{msk}, C), \mathsf{Enc}(\mathsf{msk}, m)) = \mathcal{D}(\mathcal{K}_n(\mathsf{msk}, C), \mathcal{E}_n(\mathsf{msk}, m, r)) = C^f(m) \ .$$

The probability that a pair of elements $x, y \in \{0,1\}^{2n}$ is mapped to the same value $\mathcal{K}_n(x) = \mathcal{K}_n(y)$ is 2^{-10n}, and therefore the probability that $\mathcal{K}_n$ is not an injective function, is bounded by:

$$\Pr_{\mathcal{K}_n}[\mathcal{K}_n \text{ is not injective}] \leq \Pr_{\mathcal{K}_n}[\ \exists x \neq y \text{ s.t. } \mathcal{K}_n(x) = \mathcal{K}_n(y)\ ] \leq \binom{2^{2n}}{2} \cdot \left(\frac{1}{2^{10n}}\right) \leq \frac{2^{4n}}{2^{10n}} \leq 2^{-6n} .$$

Similarly, the probability that $\mathcal{E}_n : \{0,1\}^{3n} \to \{0,1\}^{10n}$ is not injective is bounded by $2^{-10n+6n} = 2^{-4n}$.

Security. We prove the following Theorem:

Theorem 4.8. *For any oracle-aided valid adversary $\mathcal{A}$ that makes at most $q(n) = 2^{n/4}$ oracle queries, and for all $n \in \mathbb{N}$, it holds that*

$$\left| \Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n) = 1\right] - \frac{1}{2} \right| \leq 2^{-n/4} .$$

In fact, we show that the above holds for any fixing of the functions f, $\mathcal{K}_{-n} = \{\mathcal{K}_m\}_{m \in \mathbb{N}, m \neq n}$ and $\mathcal{E}_{-n} = \{\mathcal{E}_m\}_{m \in \mathbb{N}, m \neq n}$. From this point and forward we fix $n \in \mathbb{N}$ and the functions f, $\mathcal{K}_{-n} = \{\mathcal{K}_m\}_{m \in \mathbb{N}, m \neq n}$ and $\mathcal{E}_{-n} = \{\mathcal{E}_m\}_{m \in \mathbb{N}, m \neq n}$.

The proof of Theorem 4.8 consists of two somewhat independent parts. First, in Section 4.4.1, we show that the decryption oracle $\mathcal{D}$ does not provide the adversary with any significant capabilities, and it can almost always be simulated by the adversary itself. Specifically, since the output spaces of $\mathcal{K}_n$ and $\mathcal{E}_n$ are much larger than their input spaces, the adversary should not be able to find a valid output of $\mathcal{K}_n$ or $\mathcal{E}_n$ without querying them beforehand. As a result, almost all queries to $\mathcal{D}$ on values that were not obtained from previous queries to $\mathcal{K}_n$ or $\mathcal{E}_n$ can be replied to with $\perp$.

Then, in Section 4.4.2, we show that the only way in which an adversary can obtain any advantage in the experiment $\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$ without accessing the decryption oracle $\mathcal{D}$ is by "hitting" the randomness r^* used for generating the challenge ciphertext in one of its $\mathcal{E}_n$ or $\mathsf{Enc}(\mathsf{msk}\cdot)$-queries. We then show that since the adversary makes a bounded number of oracle queries, the probability of hitting r^* is very small.

4.4.1 Simulating the Decryption Oracle

The event spoof will help us show that the oracle $\mathcal{D}$ can be simulated by the adversary itself. We formally define this event and then show that it occurs with very small probability. We have:

Definition 4.9. *For any oracle-aided algorithm M, consider the following event spoof_n that may occur during an execution of $M^{\Psi}(1^n)$: The algorithm makes a query $\mathcal{D}_n(\mathsf{sk}_C, c)$ with $|\mathsf{sk}_C| = |c| = 10n$ whose output is not $\perp$, yet sk_C was not an output of a previous $\mathcal{K}_n$-query or c was not an output of a previous $\mathcal{E}_n$-query.*

We show:

Claim 4.10. *For any oracle-aided algorithm M making at most q oracle queries, any n, for any f and any $\mathcal{K}_{-n} = \{\mathcal{K}_m\}_{m \in \mathbb{N}, m \neq n}$, $\mathcal{E}_{-n} = \{\mathcal{E}_m\}_{m \neq n}$, the probability that spoof_n occurs in an execution of $M^{\Psi}(1^n)$ is:*

$$\Pr_{\mathcal{K}_n, \mathcal{E}_n}[\mathsf{spoof}_n] \leq q \cdot 2^{-6n} .$$

Proof. Fix M, n, f, $\mathcal{K}_{-n}$ and $\mathcal{E}_{-n}$. The input space of $\mathcal{K}_n$ is of size 2^{2n} whereas its output space is 2^{10n}. Since $\mathcal{K}_n$ is chosen uniformly at random, there are at most 2^{2n} elements in the range of $\mathcal{K}_n$ and these are distributed uniformly in a space of size 2^{10n}. Any query to $\mathcal{K}_n$ reveals one point in the range of $\mathcal{K}_n$, but gives no information about other points in the range. Similarly, any $\mathcal{D}_n$ query may give information regarding one point in the range of $\mathcal{K}_n$, but nothing else. Therefore, the oracle queries do not give significant information regarding the range of $\mathcal{K}$, and an adversary cannot hit points in the range without previous queries of $\mathcal{K}$. We have a similar behaviour when considering the oracle $\mathcal{E}_n$.

Formally, we follow the computation of $M^{\Psi}(1^n)$. During the computation, we store tables $T(\mathcal{K})$ and $T(\mathcal{E})$ of oracle queries and answers for $\mathcal{K}_n$ and $\mathcal{E}_n$, both tables are initialized to $\emptyset$. Then:

- Each f-query can be answered since f is fixed, and this does not trigger the event spoof_n.
- With each $\mathcal{K}_n$ query $(\mathsf{msk}, C) \in \{0,1\}^n \times \{0,1\}^n$ we first check in $T(\mathcal{K})$ whether (msk, C) was queried before. If so - we answer the stored value in $T(\mathcal{K})$. Otherwise, we choose a random output $\mathsf{sk}_C \leftarrow \{0,1\}^{10n}$, return sk_C and store the pair $((\mathsf{msk}, C), \mathsf{sk}_C)$ in $T(\mathcal{K})$.
- With each $\mathcal{E}_n$ query in $\{0,1\}^{3n}$ we act in a similar way and check whether there exists an answer in $T(\mathcal{E})$. If not, we choose a random element in $\{0,1\}^{10n}$, store the query/answer in $T(\mathcal{E})$ and return the answer to the adversary.
- With each $\mathcal{D}_n$-query $(\mathsf{sk}_C, c) \in \{0,1\}^{20n}$, we check whether there exist a pair $((\mathsf{msk}, C), \mathsf{sk}_C) \in T(\mathcal{K})$ and a pair $((\mathsf{msk}, m, r), c) \in T(\mathcal{E})$.
  1. If there does not exist a pair $((\mathsf{msk}, C), \mathsf{sk}_C) \in T(\mathcal{K})$, then we toss a coin α with probability $p_j = (2^{2n} - j)/2^{10n}$ to be 1, where j is the number of times we previously tossed this coin. If $\alpha = 1$, then we choose a uniformly random $(\mathsf{msk}, C) \in \{0,1\}^n \times \{0,1\}^n$ such that $(\mathsf{msk}, C) \notin T(\mathcal{K})$. Otherwise, we store $(\perp, \mathsf{sk}_C) \in T(\mathcal{K})$.
  2. Similarly, if there does not exist a pair $((\mathsf{msk}', m, r), c) \in T(\mathcal{E})$, we determine whether c has a pre-image in $\mathcal{E}_n$, and choose a random coin β that equals one with probability $(2^{3n} - k)/2^{10n}$, where k is the number of times we previously tossed this coin. In case the coin is 1, we choose a random pre-image $(\mathsf{msk}', m, r) \in \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n$ and store it in $T(\mathcal{E})$.

  Given the pairs $((\mathsf{msk}, C), \mathsf{sk}_C) \in T(\mathcal{K})$ and $((\mathsf{msk}', m, r'), c) \in T(\mathcal{E})$, we immediately return $\perp$ if either $(\mathsf{msk}, C) = \perp$ or $(\mathsf{msk}', m, r') = \perp$. Otherwise, if $\mathsf{msk} \neq \mathsf{msk}'$, we answer the adversary with $\perp$ well. Only if $\mathsf{msk} = \mathsf{msk}'$, we evaluate $C^f(m)$ and give the result to the adversary.

Note that spoof_n occurs whenever $\alpha = 1$ or $\beta = 1$. With each $\mathcal{D}_n$-query, these coin may be tossed at most once. Since there are at most q queries overall, using union bound the probability that in one of the queries $\alpha = 1$ or $\beta = 1$ is bounded by:

$$\Pr_{\mathcal{K}_n, \mathcal{E}_n}[\mathsf{spoof}_n] \leq 2 \cdot \frac{q}{2^{7n}} = \frac{q}{2^{6n}} \ .$$

■

We denote by $\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$ an execution of the experiment, where the construction Π as an oracle access to Ψ, but the adversary $\mathcal{B}$ has an oracle access to $f, \mathcal{K}, \mathcal{E}, \mathcal{D}_{-n}$ (together with $\mathsf{KG}(\mathsf{msk}, \cdot), \mathsf{Enc}(\mathsf{msk}, \cdot)$) but has no access to the oracle $\mathcal{D}_n$. We now show that if there exists an adversary $\mathcal{A}$ that its advantage in the experiment $\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$ is ϵ, then there exists an adversary $\mathcal{B}$ that does not use the oracle $\mathcal{D}_n$ at all, and its advantage in the experiment $\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$ is very close to ϵ. Since we prove that Π is secure functional-encryption for any fixing of the function f, we let $\mathcal{B}$ make an unbounded amount of queries to f and analyze its complexity with respect to the amount of oracle queries to $\mathcal{K}_n, \mathcal{E}_n$. Formally:

Claim 4.11. *For every $n \in \mathbb{N}$, if there exists an oracle-aided adversary $\mathcal{A}$ making at most $q(n)$-oracle queries with $q(n) \leq 2^{n/2}$ such that*

$$\left| \Pr_{\substack{\mathcal{K},\mathcal{E} \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\ \mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n;\mathsf{msk},b,r^*) = 1 \ \right] - \frac{1}{2} \right| > \epsilon \ ,$$

then there exists an oracle-aided adversary $\mathcal{B}$ that makes at most $q(n)$ queries to $\mathcal{K}_n$ and $\mathcal{E}_n$, and does not query $\mathcal{D}_n$, such that:

$$\left| \Pr_{\substack{\mathcal{K},\mathcal{E} \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\ \widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n;\mathsf{msk},b,r^*) = 1 \ \right] - \frac{1}{2} \right| > \epsilon - 2^{-n} \ .$$

Proof. Fix n. Given the adversary $\mathcal{A}$, we build an adversary $\mathcal{B}$ that makes no oracle queries to $\mathcal{D}_n$. As long as spoof_n does not occur, all oracle queries to $\mathcal{D}_n$ can be answered by the previous queries to $\mathcal{K}_n$ and $\mathcal{E}_n$. We proceed with a formal description of $\mathcal{B}$.

A formal description of $\mathcal{B}$. On input security parameter 1^n, $\mathcal{B}$ invokes the adversary $\mathcal{A}$ on the same input, submits all its oracle queries to $\mathsf{KG}(\mathsf{msk},\cdot)$, $\mathsf{Enc}(\mathsf{msk},\cdot)$ to its own KG,Enc oracles, but keeps a log of these oracle queries/answers. Regarding Ψ, it submit all the oracle queries of f, as well as $\mathcal{K},\mathcal{E},\mathcal{D}_{-n}$, to its own oracle, but keeps track of these queries/answers to $\mathcal{K}_n,\mathcal{E}_n$. We later show how it answers $\mathcal{D}_n$-queries. Whenever $\mathcal{A}$ outputs a pair of messages (m_0,m_1), $\mathcal{B}$ outputs the same pair of messages, receives challenge c^*, passes it back to $\mathcal{A}$ and continues to answer its oracle queries as before. At the end of the computation, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

We now describe how $\mathcal{B}$ answers $\mathcal{A}$ queries (sk_C,c) for the $\mathcal{D}_n$-oracle. Informally, regarding sk_C, the algorithm $\mathcal{B}$ has to determine whether it is a valid secret-key, and to find the pair (msk',C) for which $\mathsf{sk}_C = \mathcal{K}_n(\mathsf{msk}',C)$. Similarly it has to determine whether c is a valid encryption, what message m it hides and under what master secret-key msk. If the two encryptions are valid, and agree with their master secret-key, then it evaluates $C^f(m)$. Otherwise, it just returns $\perp$. Formally, for any query (sk_C,c) to $\mathcal{D}_n$, where $c \neq c^*$ (the challenge ciphertext), $\mathcal{B}$ performs as follows:

- If sk_C is an output of a previous query to $\mathsf{KG}(\mathsf{msk},\cdot)$-oracle, then $\mathcal{B}$ has the pre-image C^f and it knows that sk_C was encrypted using the master secret key msk chosen by the experiment.
- If sk_C is an output of a previous query to $\mathcal{K}_n(\cdot)$, then $\mathcal{B}$ knows the pair (msk',C) that was used to obtain sk_C. It then queries its oracle $\mathsf{KG}(\mathsf{msk},\cdot)$ on C. If the result is sk_C, and assuming that $\mathcal{K}_n$ is injective, then it knows that sk_C was encrypted using the master key msk chosen by the experiment. Otherwise, it knows that sk_C was encrypted using msk' (differ than msk).
- If c is an output of a previous query to Enc, then $\mathcal{B}$ looks for the message m, and marks that it was encrypted master secret key msk chosen by the experiment.
- If c is an output of a previous query to $\mathcal{E}_n$, then $\mathcal{B}$ knows m and the master secret-key msk' that was used for the encryption.

If both sk_C, c relate to the same master secret key msk', then $\mathcal{B}$ replies with $C^f(m)$. If not, it replies with $\perp$. One additional case is the case where the query relates to the challenge ciphertext c^*. In this case:

- For a query (sk_C,c^*) where c^* is the challenge ciphertext, $\mathcal{B}$ performs as follows. Giving sk_C, the adversary $\mathcal{B}$ obtains the underling oracle-aided circuit C as before and check whether it was encrypted using the master-secret key msk that was chosen by the experiment. If sk_C was encrypted using msk, $\mathcal{B}$ can evaluate $C^f(m_0)$ as an answer, where m_0 is one of the pair of messages that $\mathcal{A}$ output as the challenge messages. This is correct since $\mathcal{A}$ is valid, and for the pair of messages m_0, m_1 that $\mathcal{A}$ outputs it holds that $C^f(m_0) = C^f(m_1)$. In case sk_C was encrypted using a master secret key differ than msk, $\mathcal{B}$ answers with $\perp$.

Clearly, the algorithm $\mathcal{B}$ makes no oracle-queries to $\mathcal{D}_n$, and is a valid adversary (assuming that $\mathcal{A}$ is valid). Moreover, it makes the same amount of oracle-queries to $\mathcal{K}_n, \mathcal{E}_n$ as $\mathcal{A}$ (but may query some more f-queries).

Let good denote the event where spoof_n does not occur (where the machine M that we consider includes the challenger and the adversary $\mathcal{A}$), and both $\mathcal{K}_n$ and $\mathcal{E}_n$ are injective functions. Then, conditioned on good, an execution of $\mathcal{B}$ without the oracle $\mathcal{D}_n$ is equivalent to an execution of $\mathcal{A}$ with that oracle, where both adversaries also have accesses to $\mathsf{KG}(\mathsf{msk}, \cdot), \mathsf{Enc}(\mathsf{msk}, \cdot), \mathcal{K}_n, \mathcal{E}_n, \mathcal{D}_{-n}$ and f. That is:

$$\begin{aligned}
&\Pr_{\substack{\mathcal{K},\mathcal{E}\\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\ \mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \wedge \ \mathsf{good} \ \right] \\
&\qquad = \Pr_{\substack{\mathcal{K},\mathcal{E}\\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\ \widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \wedge \ \mathsf{good} \ \right] \ .
\end{aligned}$$

Moreover, we have that:

$$\Pr\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1\right] \leq \Pr\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \ \wedge \ \mathsf{good} \ \right] + \Pr\left[\ \overline{\mathsf{good}} \ \right] \ ,$$

where the probability is taken over $\mathcal{K}_n, \mathcal{E}_n, (\mathsf{msk}, b, r^*) \leftarrow \{0,1\}^{2n+1}$. Finally, note that:

$$\begin{aligned}
\Pr\left[\ \overline{\mathsf{good}} \ \right] &\leq \Pr\left[\ \mathsf{spoof}_n \ \right] + \Pr\left[\ \mathcal{K}_n \text{ is not injective}\right] + \Pr\left[\ \mathcal{E}_n \text{ is not injective}\right] \\
&\leq q(n) \cdot 2^{-6n} + 2 \cdot 2^{-4n} \leq 2^{-n} \ .
\end{aligned}$$

We conclude:

$$\left| \Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n\\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \ \right] - \frac{1}{2} \right| > \epsilon - 2^{-n} \ .$$

■

4.4.2 From Distinguishing to Hitting

In this section we show that an adversary can gain an advantage in the experiment only if it "hits" the randomness r^* which is used in the encryption of the challenge message. We then show that the probability of hitting r^* using its oracles is very small. Formally:

Definition 4.12. *For a given $n \in \mathbb{N}$, oracle-aided and valid adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$, we consider the following event that may occur during the execution of $\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*)$. Let m_0, m_1 be the two messages that were outputted by $\mathcal{B}_1$. We denote by r^*-hit the event in which $\mathcal{B}$ queries $\mathcal{E}_n$ on input (msk, m_0, r^*) or (msk, m_1, r^*). This may be occur as $\mathcal{B}$ makes a direct query to $\mathcal{E}_n$, or as an indirect query to $\mathcal{E}_n$ by its $\mathsf{Enc}(\mathsf{msk}, \cdot)$-oracle.*

Claim 4.13. *For every $n \in \mathbb{N}$, valid oracle-aided adversary $\mathcal{B}$ that makes at most $q(n) < 2^{n/2}$ queries to $\mathcal{K}_n$ and $\mathcal{E}_n$, if*

$$\left| \Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n;\mathsf{msk},b,r^*) = 1 \right] - \frac{1}{2} \right| > \epsilon \ ,$$

then

$$\Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[r^*\text{-}\mathsf{hit} \right] > \epsilon \ .$$

Proof. Fix the entire probability space except for the values $\mathcal{E}_n(\mathsf{msk}, m_0, r^*)$ and $\mathcal{E}_n(\mathsf{msk}, m_1, r^*)$. Choose two random values $c, c' \in \{0,1\}^{10n}$. We will consider two cases: One corresponds to $b = 0$, where we set $\mathcal{E}_n(\mathsf{msk}, m_0, r^*)$ to c and $\mathcal{E}_n(\mathsf{msk}, m_1, r^*)$ to c', and the second case (corresponding to $b = 1$) in which we assign the opposite values. In both cases $\mathcal{B}$ is given the value c. The two cases are equally likely but yield different values to b. We show that if $\mathcal{B}$ makes no r^*-hit (i.e., for both $\mathcal{B}_1$ and $\mathcal{B}_2$), then the view of $\mathcal{B}$ is independent of b and it must output the same value in the two cases, and thus:

$$\Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n;\mathsf{msk},b,r^*) = 1 \ \middle| \ \overline{r^*\text{-}\mathsf{hit}} \right] = \frac{1}{2} \ .$$

In particular, we show that if the execution makes no hits, then we can simulate the execution given the (partially) determined probability space defined above. In particular, we show that for all queries that $\mathcal{B}$ may query can be answered. Specifically:

- All oracle queries to f can be answered, since f is fixed.
- All oracle queries to $\mathsf{KG}(\mathsf{msk}, \cdot)$ can be answered, since the oracle $\mathcal{K}(\cdot)$ is fully-determined.
- Likewise, all oracle queries $\mathcal{K}(\cdot)$ can be answered since $\mathcal{K}$ is fully-determined.
- All oracle queries to $\mathsf{Enc}(\mathsf{msk}, \cdot)$ can be answered. We assume that r^*-hit does not occur, thus there is no evaluation of $\mathcal{E}_n$ on the inputs (msk, m_0, r^*) or (msk, m_1, r^*) and therefore all the queries are fully-determined.
- Likewise, all the queries to $\mathcal{E}$ are fully-determined, since r^*-hit does not occur.

We therefore conclude that:

$$\Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[r^*\text{-}\mathsf{hit} \right] > \epsilon \ .$$

■

We now show that the probability that r^*-hit occurs is small. That is:

Lemma 4.14. *For every valid oracle-aided adversary $\mathcal{B}$, making at most $q(n)$ queries to $\mathcal{K}_n$ and $\mathcal{E}_n$ in the experiment $\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n;\mathsf{msk},b,r^*)$, it holds that:*

$$\Pr_{\substack{\mathcal{K}_n,\mathcal{E}_n \\ (\mathsf{msk},b,r^*)\leftarrow\{0,1\}^{2n+1}}} \left[r^*\text{-}\mathsf{hit} \right] < \frac{q(n)}{2^n - q(n)} \ .$$

Proof. Fix the entire probability space except for $\mathcal{E}_n$ and r^*. The view of the algorithm $\mathcal{B}_1$, i.e., the view of $\mathcal{B}$ prior to the challenge phase, is completely independent of r^*. Moreover, it makes at most $q(n)$-oracle queries to $\mathcal{E}_n$ (or Enc), and the responses to these queries are distributed uniformly in $\{0,1\}^{10n}$. Similarly, after receiving the challenge c^*, the adversary $\mathcal{B}_2$ (who does not have an oracle-access to $\mathcal{D}_n$) receives with each oracle query to $\mathcal{E}_n$ (or Enc) a uniformly chosen value in $\{0,1\}^{10n}$. These values do not provide any information regarding r^*, unless a direct query to r^* is performed. Since r^* is distributed uniformly in $\{0,1\}^n$, the probability that the ith query hits r^* is $1/(2^n - i)$. Thus, the success probability of $\mathcal{B}$ is bounded by $\frac{q(n)}{2^n - q(n)}$. ■

4.4.3 Concluding the Proof

We are now ready for the proof of Theorem 4.8.

Proof of Theorem 4.8. Assume towards a contradiction that there exists an oracle-aided adversary $\mathcal{A}$ that makes at most $q(n) < 2^{n/2}$ queries, such that

$$\left| \Pr_{\substack{\mathcal{K}_n, \mathcal{E}_n \\ (\mathsf{msk}, b, r^*) \leftarrow \{0,1\}^{2n+1}}} \left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{A},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \right] - \frac{1}{2} \right| > 2^{-n/4}$$

and infinitely many n's. By Claim 4.11, this implies the existence of a valid oracle-aided adversary $\mathcal{B}$ that makes at most $q(n)$ oracle queries to $\mathcal{K}_n$ and $\mathcal{E}_n$, and does not have oracle access to $\mathcal{D}_n$, for which:

$$\left| \Pr_{\substack{\mathcal{K}_n, \mathcal{E}_n \\ (\mathsf{msk}, b, r^*) \leftarrow \{0,1\}^{2n+1}}} \left[\widetilde{\mathsf{Exp}}^{\mathsf{FE}}_{\Psi,\Pi,\mathcal{B},\mathcal{C}}(n; \mathsf{msk}, b, r^*) = 1 \right] - \frac{1}{2} \right| > 2^{-n/4} - 2^{-n} .$$

By Claim 4.13, this implies that:

$$\Pr_{\substack{\mathcal{K}_n, \mathcal{E}_n \\ (\mathsf{msk}, b, r^*) \leftarrow \{0,1\}^{2n+1}}} \left[r^*\text{-}\mathsf{hit} \right] > 2^{-n/4} - 2^{-n} > 2^{-n/4+1} .$$

However, this is in contradiction to Lemma 4.14, which shows that this probability is bounded by:

$$\frac{q(n)}{2^n - q(n)} \leq \frac{2^{n/2}}{2^n - 2^{n/2}} = \frac{1}{2^{n/2} - 1} \leq 2^{-n/4+1} .$$

■

4.5 Breaking Any Perfectly-Complete Bit-Agreement Protocol Relative to Ψ

In this section we prove the following theorem:

Theorem 4.15. *For any polynomial-time perfectly-complete oracle-aided bit-agreement protocol (A, B) there exists an oracle-aided adversary E that makes a polynomial number of oracle queries such that*

$$\left| \Pr_{\Psi} \left[\mathsf{Exp}^{\mathsf{KA}}_{\Psi,(A,B),E}(n) = 1 \right] - \frac{1}{2} \right| \geq \frac{1}{4} .$$

Our proof of the above theorem is inspired by the simplified approach of Brakerski et al. [BKS+11] for breaking perfectly-complete bit-agreement protocols. First, for illustrating the main ideas underlying the proof, we present their attacker when there is only one oracle which is a random permutation. Then, we show that this can be generalized (dealing with various additional difficulties) to our oracle Ψ.

4.5.1 Warm-up: Breaking Perfectly-Complete Bit Agreement Relative to f

Let $(\mathcal{A}, \mathcal{B})$ be a black-box construction of perfectly-complete bit-agreement protocol from a random permutation f. Let $q_{\mathcal{A}}$ (resp., $q_{\mathcal{B}}$) be a polynomial upper bound on the number of queries made by $\mathcal{A}$ (resp., $\mathcal{B}$). Consider the following attacker E.

The attacker E.

- **Input:** *A transcript T of an execution of $\langle \mathcal{A}^f(1^n), \mathcal{B}^f(1^n)\rangle$.*
- **Oracle access:** *The oracle $f = \{f_n\}_{n\in\mathbb{N}}$, where each f_n is a random permutation over $\{0,1\}^n$.*
- **The adversary:** *E initializes a set $Q(E)$ of query/answer pairs of f, a multi-set of candidate keys K, and runs $2q_{\mathcal{B}}+1$ iterations of the following two steps:*
  1. Simulation phase: *E finds a view of $\mathcal{A}$ consistent with the given transcript T and with the queries $Q(E)$ that E made so far to f. The view consists of randomness $r_{\mathcal{A}}$ for $\mathcal{A}$ and of a set of oracle queries/answers $\widehat{Q}(\mathcal{A})$. The set $\widehat{Q}(\mathcal{A})$ is consistent with the set of oracle queries $Q(E)$ that E made so far to the true oracle f, but queries in $\widehat{Q}(\mathcal{A}) \setminus Q(E)$ may be inconsistent with the true oracle f. Let k denote the key computed by $\mathcal{A}$ according to the sampled view, then E adds k to K.*
  2. Update phase: *E makes all queries in $\widehat{Q}(\mathcal{A}) \setminus Q(E)$ to the true oracle f, and adds the resulting query/answer pairs to $Q(E)$.*
- **Output:** *After $2q_{\mathcal{B}}+1$ iterations as above, E outputs the majority value in K (either 0 or 1).*

Note that E makes in each iteration at most $q_{\mathcal{A}}$ queries to f. Thus, it makes $O(q_{\mathcal{A}} \cdot q_{\mathcal{B}})$ queries overall. We claim that E outputs the key computed by $\mathcal{A}$ and $\mathcal{B}$ with probability 1 based on the prefect completeness of the protocol. In particular, we show the following:

Claim 4.16. *Let k denote the actual key computed by $\mathcal{A}$ and $\mathcal{B}$ in an execution of the protocol, producing a transcript T. Then, in each iteration of the attack of $E^f(T)$, at least one of the following two events occur:*

1. *E queries f with one of the queries made by $\mathcal{B}$ in the real execution.*
2. *E adds k to K.*

Proof. Let $Q(\mathcal{B})$ denote the queries made by $\mathcal{B}$ in the real execution of the protocol. In a given iteration, there are two possibilities:

1. If $\widehat{Q}(\mathcal{A}) \cap Q(\mathcal{B}) \nsubseteq Q(E)$, that is, in the sampled view $\widehat{Q}(\mathcal{A})$ in the current iteration there is a query that $\mathcal{B}$ makes in the real execution which was not made by E to the true oracle f. Thus, in the update phase of this iteration, $\mathcal{B}$ will make all the queries $\widehat{Q}(\mathcal{A}) \setminus Q(E)$, and will make this query to the true oracle f, and Case 1 in the claim is satisfied.
2. If $\widehat{Q}(\mathcal{A}) \cap Q(\mathcal{B}) \subseteq Q(E)$, that is, all the shared queries in the sampled view of $\mathcal{A}$ in the current iteration and in the queries of $\mathcal{B}$ in the real execution, were already queried by E, and are therefore consistent with f. Thus, there is an execution of the protocol with some oracle $\widehat{f}$ that yields the observed transcript T, a view for $\mathcal{B}$ identical to the view of the real $\mathcal{B}$, and a view for $\mathcal{A}$ identical to the view generated by E in the current iteration. Perfect completeness implies that the key k computed by $\mathcal{A}$ in this case must match the (actual) key computed by $\mathcal{B}$, and thus E adds the correct key k to K.

Since $\mathcal{B}$ makes at most $q_{\mathcal{B}}$ queries, it follows that there are at most $q_{\mathcal{B}}$ iterations in which E adds an incorrect key to K, and so at least $q_{\mathcal{B}}+1$ iterations in which E adds the correct key to K. Since E outputs the key that occurs most often, E always outputs the correct key. ■

4.5.2 Breaking Perfectly-Complete Bit Agreement Relative to Ψ

Unfortunately, the proof provided above does not naturally extend relative to the oracle Ψ. Intuitively, the proof of the previous section can be interpreted as follows: The hope that two parties would have achieve a key agreement relative to f in the presence of an eavesdropping adversary, is essentially by making the same query to the oracle f. When we consider the oracle Ψ instead of just the oracle f, the oracles $\mathcal{K}$,$\mathcal{E}$ and $\mathcal{D}$ introduce additional dependencies between the parties. These dependencies can help the parties reach an agreement even if they do not perform the same queries to Ψ.

Consider for example, the following somewhat naive (and completely insecure) protocol:

- $\mathcal{A}$ samples $\mathsf{msk} \leftarrow \{0,1\}^n$ and $k \leftarrow \{0,1\}$, and computes $\mathsf{sk}_C = \mathcal{K}(\mathsf{msk}, C)$ where C is the identity circuit, and $c = \mathcal{E}(\mathsf{msk}, k, r)$ for some $r \in \{0,1\}^n$. Then $\mathcal{A}$ sends sk_C and c to $\mathcal{B}$, and outputs the key k.
- $\mathcal{B}$ receives sk_C and c from $\mathcal{A}$, and outputs the key $k = \mathcal{D}(\mathsf{sk}_C, c)$.

In this protocol the parties always agree on a uniformly-chosen bit k, but they make completely different oracle queries: $\mathcal{A}$ queries only $\mathcal{K}$ and $\mathcal{E}$, while $\mathcal{B}$ queries only $\mathcal{D}$ (and they do not even query f). Given that the parties never make the same query to Ψ, a natural generalization of the adversary E presented in the previous section fails to recover the key k. The fact that Ψ introduces additional dependencies requires a more subtle proof, and therefore we need to revise the attack and its analysis.

Proof of Theorem 4.15. Before we define the adversary E and analyze its success probability, we start with some preliminaries and necessary definitions.

Preliminaries. We let $Q(\mathcal{A})$, $Q(\mathcal{B})$ and $Q(E)$ denote the set of oracle queries made by $\mathcal{A}$,$\mathcal{B}$ and E, respectively. We write, e.g., $[\mathcal{K}_n(\mathsf{msk}, C) = \mathsf{sk}_C] \in Q(\mathcal{A})$ to denote that $\mathcal{A}$ made the query $\mathcal{K}_n(\mathsf{msk}, C)$ and received back sk_C. Likewise, $[\mathcal{E}_n(\mathsf{msk}, m, r) = c] \in Q(\mathcal{A})$ denotes that $\mathcal{A}$ made the query $\mathcal{E}_n(\mathsf{msk}, m, r)$ and received back c. We also use the symbol $\star$ to indicate an arbitrary value, for instance, $[\mathcal{E}_n(\mathsf{msk}, m, r) = \star] \in Q(\mathcal{B})$ denotes that $\mathcal{B}$ queried $\mathcal{E}_n$ on (msk, m, r) (but we are not interested in the value that was returned by the oracle).

The extended view of $\mathcal{A}$. Since the oracles $(f, \mathcal{K}, \mathcal{E}, \mathcal{D})$ have some dependencies (i.e., the answers for some queries depend on the answers on some queries in other oracles), we want that these dependencies will appear explicitly in the set of queries/answers that the adversary samples and will not be "hidden". In our attack, E will repeatedly sample an *extended* view of $\mathcal{A}$ and not just the view of $\mathcal{A}$. We denote an extended view by $(r_{\mathcal{A}}, \mathsf{Partial}(\Psi'))$, where $r_{\mathcal{A}}$ are random coins for $\mathcal{A}$, and $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ is a set of query/answer pairs that include all those made by $\mathcal{A}$ together with additional queries that will make it "consistent" as defined below (note that this set $\mathsf{Partial}(\Psi')$ is not necessarily consistent with the true oracle Ψ).

Specifically, when sampling the oracle queries/answers $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ we will make sure that $\mathcal{K}'$,$\mathcal{E}'$ and $\mathcal{D}'$ are consistent between themselves, and that f' contains "sufficient information" about f. That is, for instance, querying $\mathcal{D}'$ with (sk_C, c) such that $[\mathcal{K}_n(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathcal{K}'_n$ and $[\mathcal{E}_n(\mathsf{msk}, m, r) = c] \in \mathcal{E}'_n$ should be answered with $C^{f'}(m)$, where f' should contains all the necessary information for computing $C^{f'}(m)$. Formally:

Definition 4.17 (consistent oracle queries/answers)**.** *Let* $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ *be a set of queries/answers. We say it is* $\mathsf{consistent}$ *if:*

1. *For every pair of queries $[\mathcal{K}_s(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathcal{K}'$ and $[\mathcal{E}_s(\mathsf{msk}, m, r) = c] \in \mathcal{E}'$ with $|\mathsf{msk}| = |C| = |m| = |r| = s$ for any $s \in \mathbb{N}$:*
   (a) *The oracle f' contains queries/answers sufficient to evaluate $C^{f'}(m)$.*
   (b) *The oracle $\mathcal{D}'$ contains the query $[\mathcal{D}_s(\mathsf{sk}_C, c) = C^{f'}(m)]$.*
2. *For every $[\mathcal{D}_s(\mathsf{sk}_C, c) = \beta] \in \mathcal{D}'$ with $\beta \neq \perp$ and $|\mathsf{sk}_C| = |c| = 10s$, there exist $\mathsf{msk}, C, m, r \in \{0,1\}^s$ for which there exist queries $[\mathcal{K}_s(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathcal{K}'$ and $[\mathcal{E}_s(\mathsf{msk}, m, r) = c] \in \mathcal{E}'$. Moreover, $\beta = C^{f'}(m)$.*

Given that the sampled set $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ is consistent, we now define a consistent extended view:

Definition 4.18 (consistent extended view)**.** *Let T be a transcript of an execution between $\mathcal{A}(1^n)$ and $\mathcal{B}(1^n)$, and let $Q(E)$ be a set of queries/answers made by E so far to the real oracle $\Psi = (f, \mathcal{K}, \mathcal{E}, \mathcal{D})$. We say that the extended view $(r_{\mathcal{A}}, \mathsf{Partial}(\Psi'))$ is* consistent *with T and $Q(E)$ if $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ is consistent, and:*

1. *Every query in $Q(E)$ is in $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ and is answered the same way.*
2. *$\mathcal{A}^{f', \mathcal{K}', \mathcal{E}', \mathcal{D}'}(1^n; r_{\mathcal{A}})$ when fed with incoming messages as in T, would generate outgoing messages consistent with T.*

Cross-augmented oracle-queries. For the analysis, to the set of real queries $Q(\mathcal{AB}) = Q(\mathcal{A}) \cup Q(\mathcal{B})$, we add some additional queries to the real oracle Ψ, similarly to what we did with the extended view above. We stress that these oracle queries are not necessarily performed by either one of $\mathcal{A}$ or $\mathcal{B}$ in the actual protocol, and these additional queries are needed only for the analysis. Formally, for a set of oracle queries $Q(\mathcal{AB})$, we define the set of cross-augmented oracle queries $\mathsf{AugQ}(\mathcal{AB})$, initialized with all queries/answers in $Q(\mathcal{AB})$, and in addition:

1. For every pair of queries $[\mathcal{K}_s(\mathsf{msk}, C) = \mathsf{sk}_C] \in Q(\mathcal{AB})$ and $[\mathcal{E}_s(\mathsf{msk}, m, r) = c] \in Q(\mathcal{AB})$ with $|\mathsf{msk}| = |C| = |m| = |r| = s$:
   (a) The set $\mathsf{AugQ}(\mathcal{AB})$ contains also all queries/answers to the true oracle f that are needed in order to evaluate $C^f(m)$.
   (b) The set $\mathsf{AugQ}(\mathcal{AB})$ contains the query $[\mathcal{D}_s(\mathsf{sk}_C, c)]$ (which by definition its answer is $C^f(m)$).

Note that these additional queries corresponds to the consistent oracle queries/answers and extended view that the adversary E samples in the attack, as in Definition 4.17. Since the analysis assumes that spoof does no occur, the second requirement of consistent oracle queries/answers from Definition 4.17 always holds.

We denote by q be an upper bound on the size of the set $|\mathsf{AugQ}(\mathcal{AB})|$, and note that q is polynomial in $|Q(\mathcal{AB})|$.

Consistency between the real and sampled views. We want that the intersection of the set of oracle queries $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ that E maintains during the execution, and the set of queries $\mathsf{AugQ}(\mathcal{AB})$ will not contradict. That is, for any oracle query that appears in both $\mathsf{Partial}(\Psi')$ and $\mathsf{AugQ}(\mathcal{AB})$, will have the same answer in both sets. We therefore define the predicate $\mathsf{consistency}(\mathsf{Partial}(\Psi'), \mathsf{AugQ}(\mathcal{AB}))$ that receives two sets of queries/answers and equals 0 if at least one of the following occurs:

1. There exists some query in $\mathsf{Partial}(\Psi') \cap \mathsf{AugQ}(\mathcal{AB})$ that has different answers, one in $\mathsf{Partial}(\Psi')$ and the other in $\mathsf{AugQ}(\mathcal{AB})$.

2. There exists some $[f(x_1) = y] \in \mathsf{Partial}(\Psi')$ and $[f(x_2) = y] \in \mathsf{AugQ}(\mathcal{AB})$ such that $x_1 \neq x_2$.
3. There exists some $\mathcal{D}'$-query in $\mathsf{Partial}(\Psi')$ that is inconsistent with $\mathcal{K}, \mathcal{E}$-queries in $\mathsf{AugQ}(\mathcal{AB})$, or vice-versa (i.e., there exist some $\mathcal{D}$-query in $\mathsf{AugQ}(\mathcal{AB})$ that is inconsistent with $\mathcal{K}', \mathcal{E}'$-queries in $\mathsf{Partial}(\Psi')$). In particular:
   (a) There exists some $[\mathcal{D}(\mathsf{sk}_C, c) = \perp] \in \mathsf{Partial}(\Psi')$, but there exist msk, C, m, r such that $[\mathcal{K}(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathsf{AugQ}(\mathcal{AB})$ and $[\mathcal{E}(\mathsf{msk}, m, r) = c] \in \mathsf{AugQ}(\mathcal{AB})$.
   (b) There exist some $[\mathcal{K}(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathsf{Partial}(\Psi')$ and $[\mathcal{E}(\mathsf{msk}, m, r) = c] \in \mathsf{Partial}(\Psi')$, but $[\mathcal{D}(\mathsf{sk}_C, c) = \perp] \in \mathsf{AugQ}(\mathcal{AB})$.

   The events where $\mathcal{D}$-query in $\mathsf{AugQ}(\mathcal{AB})$ which is inconsistent with some $\mathcal{K}', \mathcal{E}'$-queries in $\mathsf{Partial}(\Psi')$ are defined analogously.

We now ready for a formal description of the attack.

The adversary E.

- **Input:** *A transcript T of an execution of $\langle \mathcal{A}(1^n), \mathcal{B}(1^n) \rangle$.*
- **Oracle access:** *The real oracle $\Psi = (f, \mathcal{K}, \mathcal{E}, \mathcal{D})$.*
- **The adversary:**
  1. **Avoiding spoof_s for small s.** *Let $t = 8 \log q$. The adversary E queries the oracle f on all inputs x with $|x| \leq t$; Queries $\mathcal{K}_s(\mathsf{msk}, C)$ for all $|\mathsf{msk}| = |C| = s \leq t$; Queries $\mathcal{E}_s(\mathsf{msk}, m, r)$ for all $|\mathsf{msk}| = |m| = |r| = s \leq t$; and queries $\mathcal{D}_s(\mathsf{sk}_C, c)$ on all $|\mathsf{sk}_C| = |c| = s/10 \leq t$. Denote these queries/answers by $Q^*(E)$.*
  2. *The adversary E initializes $Q(E) = Q^*(E)$ and $K = \emptyset$, and then runs $2q + 1$ iterations of the following two steps:*
     (a) **Simulation phase:** *E finds an extended view $(r_{\mathcal{A}}, \mathsf{Partial}(\Psi'))$ consistent with T and $Q(E)$ with $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ of size at most $|Q(E)| + q$. If no such extended view exists then E aborts. Otherwise, let k' be the key computed by $\mathcal{A}$ with this extended view, then E adds k' to K.*
     (b) **Update phase:** *E makes all queries in $\mathsf{Partial}(\Psi') \setminus Q(E)$ to the true oracle $\Psi = (f, \mathcal{K}, \mathcal{E}, \mathcal{D})$ and updates $Q(E)$.*
- **Output:** *E has a multiset K of $2q + 1$ possible keys. E outputs the majority value in K.*

Analysis. It is clear that E makes polynomially many queries to Ψ. For any s, define spoof_s to be the event that there is a query $[\mathcal{D}(\mathsf{sk}_C, c) \neq \perp] \in Q(\mathcal{A}) \cup Q(\mathcal{B})$, yet there is no query

$$\begin{aligned} [\mathcal{K}_s(\star, C) = \mathsf{sk}_C] &\in Q(\mathcal{A}) \cup Q(\mathcal{B}) \cup Q^*(E) \quad \text{or} \\ [\mathcal{E}_s(\star, m, \star) = c] &\in Q(\mathcal{A}) \cup Q(\mathcal{B}) \cup Q^*(E) \ . \end{aligned}$$

Let $\mathsf{spoof} = \bigvee_s \mathsf{spoof}_s$. We claim that spoof occurs with probability at most $1/16$. This is because, by construction, spoof_s cannot occur for all $s \leq t$, and by Claim 4.10 and using union bound, $\Pr[\bigvee_s \ \mathsf{spoof}_s] \leq 1/16$.

Similarly, let spoof_E be the event that, at some point during the attack, E queries $[\mathcal{D}_s(\mathsf{sk}_C, c) \neq \perp]$ to the real oracle, but there was no previous query $[\mathcal{K}_s(\star) = \mathsf{sk}_C]$ or $[\mathcal{E}_s(\star) = c]$ made by $\mathcal{A}$, $\mathcal{B}$ or E. By construction, this can only be possible if $s > 8 \log q$, since E makes at most polynomially many queries after the pre-processing phase. Moreover, spoof_E occurs with probability at most $1/16$, and note that spoof_E relates to a query of E, whereas spoof relates to queries of $\mathcal{A}$ and $\mathcal{B}$.

Let $\neg\mathsf{injective}$ denote the event when either $\mathcal{K}_s$ or $\mathcal{E}_s$ is not injective for some s. The probability of this event is upper bounded by $1/8$. In the rest of the analysis, we show that as long as the event

spoof, spoof_E or $\neg\mathsf{injective}$ do not occur, then the adversary E outputs the key computed by $\mathcal{A}$ and $\mathcal{B}$. As a result, the advantage of E in the experiment $\mathsf{Exp}^{\mathsf{KA}}_{\Psi,(A,B),E}(n)$ is at least 1/4. We have:

Claim 4.19. *Let k denote the actual key computed by $\mathcal{A}$ and $\mathcal{B}$ in an execution of the protocol, and assume neither* spoof, spoof_E *nor* $\neg\mathsf{injective}$ *occur. Then, E does not abort, and in each iteration of the attack either E adds k to K, or E adds to $Q(E)$ one of the queries in* $\mathsf{AugQ}(\mathcal{AB})$.

Proof. We first show that E never aborts; recall that E aborts whenever it cannot sample a consistent view at some iteration. Assume that $Q(E)$ is consistent at the beginning of some iteration; this is true by construction in the initial phase. Since spoof does not occur, a consistent, extended view is given by letting $\mathsf{Partial}(\Psi') = Q(E) \cup \mathsf{AugQ}(\mathcal{AB})$, which is of size at most $|Q(E)| + q$; Note that all the queries in this extended view are with respect to the true oracle Ψ. Moreover, any extended view that is actually sampled by E is consistent, unless spoof_E occurs.

Let $(r_{\mathcal{A}}, \mathsf{Partial}(\Psi'))$ be the consistent extended view chosen by E in some iteration. One of the following occurs in each iteration of E:

1. If $\mathsf{consistency}(\mathsf{Partial}(\Psi'), \mathsf{AugQ}(\mathcal{AB})) = 0$, i.e., there is some contradiction between $\mathsf{Partial}(\Psi')$ and $\mathsf{AugQ}(\mathcal{AB})$. Then, during the update phase, E corrects some query in $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ according to true oracle Ψ. This contradicting query must be a new query (i.e., a query that does not exist in $Q(E)$ at the beginning of the iteration), since $\mathsf{Partial}(\Psi')$ is always consistent with the set $Q(E)$. As a result, during the update phase of this iteration, E adds some new query of $\mathsf{AugQ}(\mathcal{AB})$ to $Q(E)$.
2. Otherwise (i.e., $\mathsf{consistency}(\mathsf{Partial}(\Psi'), \mathsf{AugQ}(\mathcal{AB})) = 1$):
   (a) If $\mathsf{Partial}(\Psi') \cap \mathsf{AugQ}(\mathcal{AB}) \nsubseteq Q(E)$ then E adds some query/answer from $\mathsf{AugQ}(\mathcal{AB})$ to $Q(E)$ in the update phase of the current iteration.
   (b) If $\mathsf{Partial}(\Psi') \cap \mathsf{AugQ}(\mathcal{AB}) \subseteq Q(E)$ then we claim that E adds the correct key k to K at the end of that iteration. We show this below.

We then claim that since $|\mathsf{AugQ}(\mathcal{AB})| \leq q$, after $2q + 1$ iterations the correct key k is inserted to K at least $q + 1$ times, and therefore the output of E is correct. It is easy to see that Cases 1 and 2a hold. We now show that at the end of Case 2b the correct key is inserted to K.

Case 2b: $\mathsf{Partial}(\Psi')$ and $\mathsf{AugQ}(\mathcal{AB})$ agree, $\mathsf{Partial}(\Psi') \cap \mathsf{AugQ}(\mathcal{AB}) \subseteq Q(E)$. That is, all the shared queries in the sampled extended view of $\mathcal{A}$ in the current iteration and in the cross-augmented queries of $\mathcal{A}$ and $\mathcal{B}$ in the real execution, were already queried by E and are thus consistent with the real oracle Ψ. We now show that this implies the existence of some oracle $\widehat{\Psi}$ that yields the observed transcript T, a view for $\mathcal{B}$ identical to the view of the real $\mathcal{B}$, and a view for $\mathcal{A}$ identical to the view of $\mathcal{A}$ in the extended view sampled by E in the current iteration. Perfect completeness implies that the key k computed by $\mathcal{A}$ in this case must match the (actual) key computed by $\mathcal{B}$ in the actual execution, and therefore E adds the correct key to K.

We now show that there exists an oracle $\widehat{\Psi} = (\widehat{f}, \widehat{\mathcal{K}}, \widehat{\mathcal{E}}, \widehat{\mathcal{D}})$ as above. Note that this oracle should agree (i.e., have the same answers) with all the queries in $\mathsf{Partial}(\Psi') = (f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ and with all the real queries $\mathsf{AugQ}(\mathcal{AB})$. We construct the oracle $\widehat{\Psi}$ as follows:

The oracle $\widehat{f}$. Note that for every $s \leq t$, the set of queries $Q^*(E)$ contains all the functions $\{f_s\}_{s \leq t}$ and thus agrees completely with Ψ (i.e., also with $\mathsf{AugQ}(\mathcal{AB})$). We therefore set $\widehat{f_s} = f_s$.

For every $s > t$, we define the function $\widehat{f_s}$ as follows. For every x such that $[f_s(x) = y'] \in \mathsf{Partial}(\Psi')$, we set $\widehat{f}(x) = y'$. For every $[f_s(x) = y] \in \mathsf{AugQ}(\mathcal{AB})$, we set $\widehat{f}(x) = y$. Since $\mathsf{Partial}(\Psi') \cap \mathsf{AugQ}(\mathcal{AB}) \subseteq Q(E)$, we have that there is no contradiction, i.e, there are no input x and outputs $y \neq y'$

such that $[f_s(x) = y'] \in f'$ and $[f_s(x) = y] \in \mathsf{AugQ}(\mathcal{AB})$. Moreover, since $\mathsf{Partial}(\Psi')$ and $\mathsf{AugQ}(\mathcal{AB})$ agree, there is no image with contradicting pre-images. For any other value $x \notin f' \cup \mathsf{AugQ}(\mathcal{AB})$, we set $\widehat{f}(x) = y$ for some arbitrarily y under the condition that $\widehat{f}$ is a permutation.

Before proceeding to the definitions of $\widehat{\mathcal{K}}$,$\widehat{\mathcal{E}}$ and $\widehat{\mathcal{D}}$, we first define sets $\mathsf{avoid}\text{-}\mathcal{K}$ and $\mathsf{avoid}\text{-}\mathcal{E}$, which (intuitively) consist of values that appear at some point in the execution of either the real or the sampled view of the adversary, and we want to avoid them whenever we define arbitrarily values for $\widehat{K}$ and $\widehat{\mathcal{E}}$. These sets are defined as follows:

$$\begin{aligned}\mathsf{avoid}\text{-}\mathcal{K}_s &= \left\{ \ \mathsf{sk}_C \in \{0,1\}^{10s} \ \middle| \ \exists \ [\mathcal{D}_s(\mathsf{sk}_C, \star) = \star] \in \mathsf{AugQ}(\mathcal{AB}) \cup \mathcal{D}'_s \ \right\} ,\\ \mathsf{avoid}\text{-}\mathcal{E}_s &= \left\{ \quad c \in \{0,1\}^{10s} \ \middle| \ \exists \ [\mathcal{D}_s(\star, c) = \star] \in \mathsf{AugQ}(\mathcal{AB}) \cup \mathcal{D}'_s \ \right\} .\end{aligned}$$

Note that the sizes of the sets $\mathsf{avoid}\text{-}\mathcal{K}_s$ and $\mathsf{avoid}\text{-}\mathcal{E}_s$ are at most q.

The oracle $\widehat{\mathcal{K}}$. The oracles $\{\widehat{\mathcal{K}}_s\}_{s \leq t}$ are set using $Q^*(E)$ which contains all functions $\{\mathcal{K}_s\}_{s \leq t}$ and therefore agrees with $\mathsf{AugQ}(\mathcal{AB})$. Then, for every $s > t$, we let $\widehat{\mathcal{K}}_s$ agree with $\mathsf{AugQ}(\mathcal{AB})$ and $\mathcal{K}'$ (this is possible since $\mathcal{K}' \cap \mathsf{AugQ}(\mathcal{AB}) \subseteq Q(E)$ – as was the case with $\widehat{f}$). Then, for any other value x for which $\widehat{\mathcal{K}}_s(x)$ is still undetermined, we let the output be some arbitrarily $\mathsf{sk}_C \in \{0,1\}^{|10x|} \notin \mathsf{avoid}\text{-}\mathcal{K}_s$ and that overall the resulting $\widehat{\mathcal{K}}_s$ function is injective.

The oracle $\widehat{\mathcal{E}}$. We define the oracle similarly to the above, such that it agrees with $\mathsf{AugQ}(\mathcal{AB})$, $\mathcal{E}'$ and arbitrarily for all other values, such that it avoids $\mathsf{avoid}\text{-}\mathcal{E}$, and that the resulting function $\widehat{\mathcal{E}}$ is injective.

The oracle $\widehat{\mathcal{D}}$. We define the oracle $\widehat{\mathcal{D}}$ using the oracles $\widehat{f}$,$\widehat{\mathcal{K}}$ and $\widehat{\mathcal{E}}$ exactly as the true oracle $\mathcal{D}$ is defined using the true oracles f,$\mathcal{K}$ and $\mathcal{E}$. We now show that $\widehat{\mathcal{D}}$ is consistent with $\mathsf{AugQ}(\mathcal{AB})$ and $\mathsf{Partial}(\Psi')$. That is, that every query $[\mathcal{D}(\mathsf{sk}_C, c)] \in \mathsf{AugQ}(\mathcal{AB}) \cup \mathcal{D}'$ has the same answer with $\widehat{\mathcal{D}}$. In particular:

1. Assume that there exists $[\mathcal{D}(\mathsf{sk}_C, c) = \beta] \in \mathcal{D}'$ for some sk_C, c and $\beta \neq \perp$. Since the oracles $(f', \mathcal{K}', \mathcal{E}', \mathcal{D}')$ are consistent (recall Definition 4.17), then there exist queries $[\mathcal{K}(\mathsf{msk}, C) = \mathsf{sk}_C] \in \mathcal{K}'$ and $[\mathcal{E}(\mathsf{msk}, m, r) = c] \in \mathcal{E}'$ for some msk,C,m and r. Moreover, from Definition 4.17, the existence of these two queries implies that f' contains all the necessary oracle queries/answers for the evaluation of $C^{f'}(m)$, and it also holds that $C^{f'}(m) = \beta$. However, since any query in $(f', \mathcal{K}', \mathcal{E}')$ has the exact same answer with $(\widehat{f}, \widehat{\mathcal{K}}, \widehat{\mathcal{E}})$, it holds that $C^{\widehat{f}}(m) = \beta$, $\widehat{K}(\mathsf{msk}, C) = \mathsf{sk}_C$, $\widehat{\mathcal{E}}(\mathsf{msk}, m, r) = c$ and so $\widehat{\mathcal{D}}(\mathsf{sk}_C, c) = \beta$ as well.
2. Assume that there exists $[\mathcal{D}(\mathsf{sk}_C, c) = \beta] \in \mathsf{AugQ}(\mathcal{AB})$ for some sk_C, c and $\beta \neq \perp$. Since spoof does not occur, there exist queries $[\mathcal{K}(\mathsf{msk}, C) = \mathsf{sk}_C], [\mathcal{E}(\mathsf{msk}, m, r) = c] \in \mathsf{AugQ}(\mathcal{AB})$ for some msk,C,m and r, and therefore $\mathsf{AugQ}(\mathcal{AB})$ contains also all the necessary f-values in order to evaluate $C^f(m)$. Since the oracles $\mathcal{K}$ and $\mathcal{E}$ are injective, from the definition of Ψ it holds that $\beta = C^f(m)$.
   However, since any $(f, \mathcal{K}, \mathcal{E})$-query in $\mathsf{AugQ}(\mathcal{AB})$ has the exact same answer in $(\widehat{f}, \widehat{\mathcal{K}}, \widehat{\mathcal{E}})$, it holds that $C^{\widehat{f}}(m) = \beta$, $\widehat{K}(\mathsf{msk}, C) = \mathsf{sk}_C$, $\widehat{\mathcal{E}}(\mathsf{msk}, m, r) = c$ and so $\widehat{\mathcal{D}}(\mathsf{sk}_C, c) = \beta$ as well.
3. For every query $[\mathcal{D}(\mathsf{sk}_C, c) = \perp] \in \mathcal{D}' \cup \mathsf{AugQ}(\mathcal{AB})$ we show that $\widehat{\mathcal{D}}(\mathsf{sk}_C, c) = \perp$ as well. Specifically, it suffices to show that there do not exist msk,C,m and r for which $\widehat{\mathcal{K}}(\mathsf{msk}, C) = \mathsf{sk}_C$ and $\widehat{\mathcal{E}}(\mathsf{msk}, m, r) = c$. Assume towards a contradiction that there exist such msk,C,m and r, then there is inconsistency only if $\widehat{\mathcal{K}}(\mathsf{msk}, C) = \mathsf{sk}_C$ and $\widehat{\mathcal{E}}(\mathsf{msk}, m, r) = c$ but $[\mathcal{D}(\mathsf{sk}_C, c) =$

$\bot] \in \mathcal{D}' \cup \mathsf{AugQ}(\mathcal{AB})$. However, this cannot occur since $\mathsf{Partial}(\Psi')$ and $\mathsf{AugQ}(\mathcal{AB})$ do not contradict, and the oracles $\widehat{K}$ and $\widehat{\mathcal{E}}$ avoid the sets $\mathsf{avoid}\text{-}\mathcal{K}$ and $\mathsf{avoid}\text{-}\mathcal{E}$, respectively.

■

This conclues the proof of Theorem 4.15. ■

4.6 Extending the Result to Indistinguishability Obfuscation

In this section we show that Theorem 4.2 can be extended for separating indistinguishability obfuscation for oracle-aided circuits from private-key functional encryption for oracle-aided circuits. As discussed in Section 1.1, this does not necessarily imply a separation in the standard model, since it may be that there exists an indistinguishability obfuscator for all polynomial-size circuits, but there does not exist such an obfuscator for polynomial-size *oracle-aided* circuits. Nevertheless, this provides substantial evidence that private-key functional encryption is somewhat unlikely to imply indistinguishability obfuscation using standard techniques.

We now formally define the class of constructions considered in this section, tailoring our definitions to the specific primitives under consideration, and then formally state our result. We consider any implementation of a one-way permutation f and a private-key functional encryption scheme Π for the class of all polynomial-size oracle-aided circuits C^f. As in Section 3, we model these primitives as two independent building blocks.

Definition 4.20. *A fully black-box construction of an indistinguishability obfuscator for the class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of all polynomial-size oracle-aided circuits from a one-way permutation and a private-key functional encryption scheme for the class $\mathcal{C}$ consists of a probabilistic polynomial-time oracle-aided algorithm $i\mathcal{O}$, an oracle-aided algorithm M that runs in time $T_M(\cdot)$, and functions $\epsilon_{M,1}(\cdot)$ and $\epsilon_{M,2}(\cdot)$, such that the following two conditions hold:*

- **Correctness:** *For any $n \in \mathbb{N}$, for any permutation f, for any correct private-key functional encryption scheme Π, and for any oracle-aided circuit $C \in \mathcal{C}_n$ it holds that*

$$\Pr\left[C^f \equiv \widehat{C}^f \ : \ \widehat{C} \leftarrow i\mathcal{O}^{f,\Pi}(1^n, C)\right] = 1.$$

- **Black-box proof of security:** *For any permutation f, for any correct private-key functional encryption scheme Π, for any (not necessarily uniform) probabilistic oracle-aided algorithm D that runs in time $T_D = T_D(n)$, and for any function $\epsilon_D = \epsilon_D(n)$, if*

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{iO}}_{(f,\Pi),i\mathcal{O},D,\mathcal{C}}(n) = 1\right] - \frac{1}{2}\right| \geq \epsilon_D(n)$$

for infinitely many values of n (see Definition 2.2 for the description of the experiment $\mathsf{Exp}^{\mathsf{iO}}_{(f,\Pi),i\mathcal{O},D,\mathcal{C}}$), then either

$$\Pr_{x\leftarrow\{0,1\}^n}\left[M^{f,\Pi,D}(f(x)) = x\right] \geq \epsilon_{M,1}\left(T_D(n)\cdot\epsilon_D^{-1}(n)\right)\cdot\epsilon_{M,2}(n)$$

or

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{FE}}_{(f,\Pi),\Pi,M^D,\mathcal{C}}(n) = 1\right] - \frac{1}{2}\right| \geq \epsilon_{M,1}\left(T_D(n)\cdot\epsilon_D^{-1}(n)\right)\cdot\epsilon_{M,2}(n)$$

for infinitely many values of n (see Definition 2.4 for the description of the experiment $\mathsf{Exp}^{\mathsf{FE}}_{(f,\Pi),\Pi,M^E,\mathcal{C}}$).

Theorem 4.21. *Let $(i\mathcal{O}, M, T_M, \epsilon_{M,1}, \epsilon_{M,2})$ be a fully black-box construction of an indistinguishability obfuscator for the class $\mathcal{C}$ of all polynomial-size oracle-aided circuits C^f from a one-way permutation f and a private-key functional encryption scheme Π for the class $\mathcal{C}$ (see Definition 4.20). Then, at least one of the following propertied holds:*

1. *$T_M(n) \geq 2^{\zeta n}$ for some constant $\zeta > 0$ (i.e., the reduction runs in exponential time).*
2. *$\epsilon_{M,1}(n^c) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$ for some constant $c > 1$ (i.e., the security loss is exponential).*

In particular, the theorem implies that if the running time $T_M(\cdot)$ of the reduction is sub-exponential and the adversary-dependent security loss $\epsilon_{M,1}(\cdot)$ is polynomial as in the vast majority of constructions, then the adversary-independent security loss $\epsilon_{M,2}(\cdot)$ must be exponential (thus ruling out even constructions that rely on sub-exponential security assumptions).

Theorem 4.21 follows from Theorem 4.2 and from the construction of a (perfectly-correct) public-key encryption scheme from a one-way function and a general-purpose indistinguishability obfuscator by Sahai and Waters [SW14]. Their construction, however, relies on the underlying one-way function in a non-black-box manner, and therefore Theorem 4.21 does not immediately follow from Theorem 4.2. However, although their construction relies the underlying one-way function in a non-black-box manner, relative to the oracle Ψ it is in fact a fully black-box construction (with a polynomial security loss). Specifically, Sahai and Waters use the underlying indistinguishability obfuscator for obfuscating a circuit that invokes a puncturable pseudorandom function and a pseudorandom generator as sub-routines. Given that puncturable pseudorandom functions and pseudorandom generators can be based on any one-way function in a fully black-box manner, in our setting such a circuit is a polynomial-size oracle-aided circuit C^f (and we note that black-box proofs of security with a polynomial security loss clearly compose nicely). Equipped with this view we now prove Theorem 4.21.

Proof of Theorem 4.21. Let $(i\mathcal{O}, M, T_M, \epsilon_{M,1}, \epsilon_{M,2})$ be a fully black-box construction of a general-purpose indistinguishability obfuscator for the class $\mathcal{C}$ of all polynomial-size oracle-aided circuits C^f from a one-way permutation f and a private-key functional encryption scheme Π for the class $\mathcal{C}$ (recall Definition 4.20). Note that in our setting, relative to the oracle Ψ, this means we allow the algorithm $i\mathcal{O}$ to access f, $\mathcal{K}$, $\mathcal{E}$ and $\mathcal{D}$ (i.e., to access Ψ). We now construct a perfectly-complete bit-agreement protocol by relying on the following two building blocks:

- A length-doubling pseudorandom generator that is constructed from the permutation f in a fully black-box manner [HIL+99] with a polynomial security loss (it in fact suffices for f to be a one-way function). This means that: (1) the pseudorandom generator is of the form G^f where $G \in \mathcal{C}$ is a polynomial-size oracle-aided circuit, and (2) any oracle-aided distinguisher that runs in time $T = T(n)$ and has an advantage $\epsilon = \epsilon(n)$ in breaking the pseudorandom generator, can be used in a black-box manner for inverting f in time that is polynomially related to T and with probability that is polynomially related to ϵ.
- A puncturable pseudorandom function that is constructed from the permutation f in a fully black-box manner [KPT+13, BW13, SW14, BGI14] with a polynomial security loss (it again suffices for f to be a one-way function). This means that: (1) the evaluation and puncturing algorithms of the family are of the form $\mathsf{PRF.Eval}^f$ and $\mathsf{PRF.Punc}^f$ where $\mathsf{PRF.Eval}, \mathsf{PRF.Punc} \in \mathcal{C}$ are polynomial-size oracle-aided circuits[14], and (2) any oracle-aided distinguisher that runs in time $T = T(n)$ and has an advantage $\epsilon = \epsilon(n)$ in breaking the puncturable pseudorandom

[14] For simplicity we assume that the key-generation algorithm of the pseudorandom family on input 1^n outputs a uniform n-bit key k.

function, can be used in a black-box manner for inverting f in time that is polynomially related to T and with probability that is polynomially related to ϵ.

Consider now the following perfectly-complete bit-agreement protocol $(\mathcal{A}, \mathcal{B})$ relative to the oracle Ψ:

- The algorithm $\mathcal{A}$, on input 1^n, samples $\mathsf{k} \leftarrow \{0,1\}^n$, computes $\widehat{C_{\mathsf{k}}} \leftarrow i\mathcal{O}^{\Psi}(C_{\mathsf{k}})$ and sends $\widehat{C_{\mathsf{k}}}$ to $\mathcal{B}$, where $C_{\mathsf{k}} \in \mathcal{C}$ is a polynomial-size oracle-aided circuit that is defined as follows: On input a value $r \in \{0,1\}^n$ and oracle access to f, it outputs $(G^f(r), \mathsf{PRF.Eval}^f(\mathsf{k}, G^f(r)))$.
- The algorithm $\mathcal{B}$, on input 1^n and $\widehat{C_{\mathsf{k}}}$, first samples $b \leftarrow \{0,1\}$ and $r \leftarrow \{0,1\}^n$. Then, it sends to $\mathcal{A}$ the value $(c_1, c_2 \oplus b)$ where $(c_1, c_2) = \widehat{C_{\mathsf{k}}}^f(r)$, and outputs $k_{\mathcal{B}} = b$.
- The algorithm $\mathcal{A}$, on input (c_1, c_2'), outputs $k_{\mathcal{A}} = c_2' \oplus \mathsf{PRF.Eval}^f(\mathsf{k}, c_1)$.

The protocol is directly based on the public-key encryption scheme of Sahai and Waters [SW14] by having $\mathcal{A}$ send $\mathcal{B}$ a public key, and then $\mathcal{B}$ samples a uniform bit b (which will serve as their output) and replies with its encryption. The protocol is clearly perfectly complete (based on the fact that $i\mathcal{O}^{\Psi}$ preserves functionality) and results is a uniformly-distributed key.

Theorem 4.5 guarantees the existence of an oracle-aided algorithm E that makes a polynomial number $T_E(n)$ of queries to the oracle Ψ, such that

$$\left| \Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{KA}}_{\Psi,(\mathcal{A},\mathcal{B}),E}(n) = 1\right] - \frac{1}{2} \right| \geq \epsilon_E(n), \tag{4.1}$$

where $\epsilon_E(n) = 1/4$ for all values of $n \in \mathbb{N}$. Given that the protocol $(\mathcal{A}, \mathcal{B})$ directly corresponds to the public-key encryption scheme of Sahai and Waters, their proof of security states that there are two possible cases to consider:

Case 1. E can be used in a black-box manner by an oracle-aided algorithm E' for inverting the one-way permutation f. Since their construction guarantees a polynomial security loss, the algorithm E' makes $T_{E'}(n) = \mathsf{poly}(T_E(n)) = \mathsf{poly}(n)$ oracle queries and succeeds with probability $\epsilon_{E'}(n) = 1/\mathsf{poly}(T_E(n) \cdot 4) = 1/\mathsf{poly}(n)$. This range of parameters for $T_{E'}(n)$ and $\epsilon_{E'}(n)$ contradicts Theorem 4.3, stating that any algorithm that makes at most $2^{n/4}$ queries to Ψ inverts f with probability at most $2^{-n/4}$, and therefore this case is not possible.

Case 2. E can be used in a black-box manner by an oracle-aided algorithm E' for breaking the indistinguishability obfuscator $i\mathcal{O}$. As in the previous case, since their construction guarantees a polynomial security loss, the algorithm E' makes $T_{E'}(n) = \mathsf{poly}(T_E(n)) = \mathsf{poly}(n)$ oracle queries and succeeds with probability $\epsilon_{E'}(n) = 1/\mathsf{poly}(T_E(n) \cdot 4) = 1/\mathsf{poly}(n)$. That is, in this case we have an oracle-aided algorithm E' making $T_{E'}(n) = \mathsf{poly}(n)$ oracle queries such that

$$\left| \Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{iO}}_{\Psi,i\mathcal{O},E',\mathcal{C}}(n) = 1\right] - \frac{1}{2} \right| \geq \epsilon_{E'}(n)$$

for infinitely many values of n, where $\epsilon_{E'}(n) = 1/\mathsf{poly}(n)$.

The analysis of this case proceeds as in the proof of Theorem 4.2. Specifically, Definition 4.20 states that there are two possible sub-cases to consider: E' can be used in a black-box manner by M either for breaking the functional encryption scheme Π, or for inverting the one-way permutation f. In the first sub-case, we obtain from Definition 4.20 that

$$\left| \Pr_{\Psi}\left[\mathsf{Exp}^{\mathsf{FE}}_{\Psi,\Pi,M^{E'},\mathcal{C}}(n) = 1\right] - \frac{1}{2} \right| \geq \epsilon_{M,1}\left(T_{E'}(n) \cdot \epsilon^{-1}_{E'}(n)\right) \cdot \epsilon_{M,2}(n),$$

where M runs in time $T_M(n)$. The algorithm M may invoke E' on various security parameters (i.e., in general M is not restricted to invoking E' only on security parameter n), and we denote by $\ell(n)$ the maximal security parameter on which M invokes E' (when M itself is invoked on security parameter n). Thus, viewing $M^{E'}$ as a single algorithm, its number of queries $T_{M^{E'}}(n)$ to the oracle Ψ satisfies $T_{M^{E'}}(n) \leq T_M(n) \cdot T_{E'}(\ell(n))$ (this follows since M runs in time $T_M(n)$ and in each step of its execution it may query Ψ directly at most once or invoke E' at most once where each such invocation results in at most $T_{E'}(\ell(n))$ queries to Ψ). Theorem 4.4 then implies that either $2^{n/4} \leq T_{M^{E'}}(n)$ or $\epsilon_{M,1}\left(T_{E'}(n) \cdot \epsilon_{E'}^{-1}(n)\right) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$. Now, since $T_{E'}(n)$ and $\epsilon_{E'}^{-1}(n)$ are some fixed polynomials in n, and since $\ell(n) \leq T_M(n)$, as in the proof of Theorem 4.2 we obtain that either $T_M(n) \geq 2^{\zeta n}$ for some constant $\zeta > 0$ or $\epsilon_{M,1}\left(n^c\right) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$ for some constant $c > 1$.

In the second sub-case, we obtain from Definition 4.20 that

$$\Pr_{\substack{\Psi \\ x \leftarrow \{0,1\}^n}}\left[\left(M^{E'}\right)^{\Psi}(f(x)) = x\right] \geq \epsilon_{M,1}\left(T_{E'}(n) \cdot \epsilon_{E'}^{-1}(n)\right) \cdot \epsilon_{M,2}(n),$$

where M runs in time $T_M(n)$. As in the first sub-case, viewing $M^{E'}$ as a single algorithm, its number of queries $T_{M^{E'}}$ to the oracle Ψ satisfies $T_{M^{E'}}(n) \leq T_M(n) \cdot T_{E'}(\ell(n))$. Theorem 4.3 then implies that either $2^{n/4} \leq T_{M^{E'}}(n)$ or $\epsilon_{M,1}\left(T_{E'}(n) \cdot \epsilon_{E'}^{-1}(n)\right) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$. Now, since $T_{E'}(n)$ and $\epsilon_{E'}^{-1}(n)$ are some fixed polynomials in n, and since $\ell(n) \leq T_M(n)$, then again as in the proof of Theorem 4.2 we obtain that either $T_M(n) \geq 2^{\zeta n}$ for some constant $\zeta > 0$ or $\epsilon_{M,1}\left(n^c\right) \cdot \epsilon_{M,2}(n) \leq 2^{-n/4}$ for some constant $c > 1$. ■

Acknowledgments

We thank Benny Applebaum, Nir Bitansky, Zvika Brakerski, Ran Canetti, Ilan Komargodski and Alon Rosen for very insightful discussions in various stages of this work.

References

[AAB$^+$13] S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. Function private functional encryption and property preserving encryption: New definitions and positive results. Cryptology ePrint Archive, Report 2013/744, 2013.

[ABG$^+$13] P. Ananth, D. Boneh, S. Garg, A. Sahai, and M. Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689, 2013.

[ABS$^+$15] P. Ananth, Z. Brakerski, G. Segev, and V. Vaikuntanathan. From selective to adaptive security in functional encryption. To appear in *Advances in Cryptology – CRYPTO '15* (available at `https://eprint.iacr.org/2014/917.pdf`), 2015.

[AJ15] P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. To appear in *Advances in Cryptology – CRYPTO '15* (available at `https://eprint.iacr.org/2015/173.pdf`), 2015.

[BCC$^+$14] N. Bitansky, R. Canetti, H. Cohn, S. Goldwasser, Y. Tauman Kalai, O. Paneth, and A. Rosen. The impossibility of obfuscation with auxiliary input or a universal simulator. In *Advances in Cryptology – CRYPTO '14*, pages 71–89, 2014.

[BCP14] E. Boyle, K. Chung, and R. Pass. On extractability obfuscation. In *Proceedings of the 11th Theory of Cryptography Conference*, pages 52–73, 2014.

[BGI$^+$01] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology – CRYPTO '01*, pages 1–18, 2001.

[BGI$^+$12] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *Journal of the ACM*, 59(2):6, 2012.

[BGI14] E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *Proceedings of the 17th International Conference on Practice and Theory in Public-Key Cryptography*, pages 501–519, 2014.

[BKS$^+$11] Z. Brakerski, J. Katz, G. Segev, and A. Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In *Proceedings of the 8th Theory of Cryptography Conference*, pages 559–578, 2011.

[BKS15] Z. Brakerski, I. Komargodski, and G. Segev. From single-input to multi-input functional encryption in the private-key setting. Cryptology ePrint Archive, Report 2015/158, 2015.

[BM07] B. Barak and M. Mahmoody-Ghidary. Lower bounds on signatures from symmetric primitives. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 680–688, 2007.

[BM09] B. Barak and M. Mahmoody-Ghidary. Merkle puzzles are optimal - An $O(n^2)$-query attack on any key exchange from a random oracle. In *Advances in Cryptology – CRYPTO '09*, pages 374–390, 2009.

[BP15] N. Bitansky and O. Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 401–427, 2015.

[BPR15] N. Bitansky, O. Paneth, and A. Rosen. On the cryptographic hardness of finding a nash equilibrium. To appear in *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science* (available at `https://eprint.iacr.org/2014/1029.pdf`), 2015.

[BPW15] N. Bitansky, O. Paneth, and D. Wichs. Perfect structure on the edge of chaos. Cryptology ePrint Archive, Report 2015/126, 2015.

[BS15] Z. Brakerski and G. Segev. Function-private functional encryption in the private-key setting. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 306–324, 2015.

[BST14] M. Bellare, I. Stepanovs, and S. Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In *Advances in Cryptology – ASIACRYPT '14*, pages 102–121, 2014.

[BSW11] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *Proceedings of the 8th Theory of Cryptography Conference*, pages 253–273, 2011.

[BV15] N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. To appear in *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science* (available at `https://eprint.iacr.org/2014/163.pdf`), 2015.

[BW13] D. Boneh and B. Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology - ASIACRYPT '13*, pages 280–300, 2013.

[BZ14] D. Boneh and M. Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *Advances in Cryptology – CRYPTO '14*, pages 480–499, 2014.

[Can97] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Advances in Cryptology – CRYPTO '97*, pages 455–469, 1997.

[CDN$^+$97] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *Advances in Cryptology – CRYPTO '97*, pages 90–104, 1997.

[CGP15] R. Canetti, S. Goldwasser, and O. Poburinnaya. Adaptively secure two-party computation from indistinguishability obfuscation. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 557–585, 2015.

[CLM$^+$13] K. Chung, H. Lin, M. Mahmoody, and R. Pass. On the power of nonuniformity in proofs of security. In *Proceedings of the 4th Innovations in Theoretical Computer Science Conference*, pages 389–400, 2013.

[CLP14] K. Chung, H. Lin, and R. Pass. Constant-round concurrent zero-knowledge from indistinguishability obfuscation. Cryptology ePrint Archive, Report 2014/991, 2014.

[CLT$^+$15] R. Canetti, H. Lin, S. Tessaro, and V. Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 468–497, 2015.

[CTP15] R. Canetti, Y. Tauman Kalai, and O. Paneth. On obfuscation with random oracles. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 456–467, 2015.

[DKR15] D. Dachman-Soled, J. Katz, and V. Rao. Adaptively secure, universally composable, multiparty computation in constant rounds. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 586–613, 2015.

[DLM$^+$11] D. Dachman-Soled, Y. Lindell, M. Mahmoody, and T. Malkin. On the black-box complexity of optimally-fair coin tossing. In *Proceedings of the 8th Theory of Cryptography Conference*, pages 450–467, 2011.

[DMM14] D. Dachman-Soled, M. Mahmoody, and T. Malkin. Can optimally-fair coin tossing be based on one-way functions? In *Proceedings of the 11th Theory of Cryptography Conference*, pages 217–239, 2014.

[GGG$^+$14] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou. Multi-input functional encryption. In *Advances in Cryptology – EUROCRYPT '14*, pages 578–602, 2014.

[GGH$^+$13] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 40–49, 2013.

[GGH$^+$14a] S. Garg, C. Gentry, S. Halevi, and M. Raykova. Two-round secure MPC from indistinguishability obfuscation. In *Proceedings of the 11th Theory of Cryptography Conference*, pages 74–94, 2014.

[GGH$^+$14b] S. Garg, C. Gentry, S. Halevi, and D. Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In *Advances in Cryptology – CRYPTO '14*, pages 518–535, 2014.

[GGH$^+$14c] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, 2014.

[GGK$^+$05] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005.

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.

[GJK$^+$15] V. Goyal, A. Jain, V. Koppula, and A. Sahai. Functional encryption for randomized functionalities. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 325–351, 2015.

[Gol00] O. Goldreich. On security preserving reductions – revised terminology. Cryptology ePrint Archive, Report 2000/001, 2000.

[GP15] S. Garg and A. Polychroniadou. Two-round adaptively secure MPC from indistinguishability obfuscation. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 614–637, 2015.

[GR14] S. Goldwasser and G. N. Rothblum. On best-possible obfuscation. *Journal of Cryptology*, 27(3):480–505, 2014.

[HHR$^+$15] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols – Tight lower bounds on the round and communication complexities of statistically hiding commitments. *SIAM Journal on Computing*, 44(1):193–242, 2015.

[HHS08] I. Haitner, J. J. Hoch, and G. Segev. A linear lower bound on the communication complexity of single-server private information retrieval. In *Proceedings of the 5th Theory of Cryptography Conference*, pages 445–464, 2008.

[HIL$^+$99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[HSW14] S. Hohenberger, A. Sahai, and B. Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In *Advances in Cryptology – EUROCRYPT '14*, pages 201–220, 2014.

[HW15] P. Hubacek and D. Wichs. On the communication complexity of secure function evaluation with long output. In *Proceedings of the 6th Conference on Innovations in Theoretical Computer Science*, pages 163–172, 2015.

[IKO05] Y. Ishai, E. Kushilevitz, and R. Ostrovsky. Sufficient conditions for collision-resistant hashing. In *Proceedings of the 2nd Theory of Cryptography Conference*, pages 445–456, 2005.

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 44–61, 1989.

[KMN$^+$14] I. Komargodski, T. Moran, M. Naor, R. Pass, A. Rosen, and E. Yogev. One-way functions and (im)perfect obfuscation. In *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science*, pages 374–383, 2014.

[KPT$^+$13] A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In *Proceedings of the 20th Annual ACM Conference on Computer and Communications Security*, pages 669–684, 2013.

[KSY15] I. Komargodski, G. Segev, and E. Yogev. Functional encryption for randomized functionalities in the private-key setting from minimal assumptions. In *Proceedings of the 12th Theory of Cryptography Conference*, pages 352–377, 2015.

[LPS04] B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology – EUROCRYPT '04*, pages 20–39, 2004.

[Lub96] M. Luby. Pseudorandomness and Cryptographic Applications. Princeton University Press, 1996.

[MMP14] M. Mahmoody, H. K. Maji, and M. Prabhakaran. On the power of public-key encryption in secure computation. In *Proceedings of the 11th Theory of Cryptography Conference*, pages 240–264, 2014.

[MP12] M. Mahmoody and R. Pass. The curious case of non-interactive commitments – On the power of black-box vs. non-black-box use of primitives. In *Advances in Cryptology – CRYPTO '12*, pages 701–718, 2012.

[Rot11] R. Rothblum. Homomorphic encryption: From private-key to public-key. In *Proceedings of the 8th Theory of Cryptography Conference*, pages 219–234, 2011.

[RTV04] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of reducibility between cryptographic primitives. In *Proceedings of the 1st Theory of Cryptography Conference*, pages 1–20, 2004.

[Sim98] D. R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology – EUROCRYPT '98*, pages 334–345, 1998.

[SW08] A. Sahai and B. Waters. Slides on functional encryption. Available at `http://www.cs.utexas.edu/~bwaters/presentations/files/functional.ppt`, 2008.

[SW14] A. Sahai and B. Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 475–484, 2014.

[Wat15] B. Waters. A punctured programming approach to adaptively secure functional encryption. To appear in *Advances in Cryptology – CRYPTO '15* (available at `https://eprint.iacr.org/2014/588.pdf`), 2015.

[Wee05] H. Wee. On obfuscating point functions. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 523–532, 2005.

[Wee07] H. Wee. One-way permutations, interactive hashing and statistically hiding commitments. In *Proceedings of the 4th Theory of Cryptography Conference*, pages 419–433, 2007.