On the Black-Box Complexity of Optimally-Fair Coin Tossing

Dana Dachman-Soled[1], Yehuda Lindell[2], Mohammad Mahmoody[3], and Tal Malkin[1]

[1] Columbia University
{dglasner, tal}@cs.columbia.edu
[2] Bar-Ilan University
lindell@macs.biu.ac.il
[3] Cornell University
mohammad@cs.cornell.edu

Abstract. A fair two-party coin tossing protocol is one in which both parties output the same bit that is almost uniformly distributed (i.e., it equals 0 and 1 with probability that is at most negligibly far from one half). It is well known that it is *impossible* to achieve fair coin tossing even in the presence of fail-stop adversaries (Cleve, FOCS 1986). In fact, Cleve showed that for every coin tossing protocol running for r rounds, an efficient fail-stop adversary can bias the output by $\Omega(1/r)$. Since this is the best possible, a protocol that limits the bias of any adversary to $O(1/r)$ is called *optimally-fair*. The only optimally-fair protocol that is known to exist relies on the existence of oblivious transfer, because it uses general secure computation (Moran, Naor and Segev, TCC 2009). However, it is possible to achieve a bias of $O(1/\sqrt{r})$ in r rounds relying only on the assumption that there exist one-way functions. In this paper we show that it is impossible to achieve optimally-fair coin tossing via a black-box construction from one-way functions for r that is less than $O(n/\log n)$, where n is the input/output length of the one-way function used. An important corollary of this is that it is impossible to construct an optimally-fair coin tossing protocol via a black-box construction from one-way functions whose round complexity is *independent* of the security parameter n determining the security of the one-way function being used. Informally speaking, the main ingredient of our proof is to eliminate the random-oracle from "secure" protocols with "low round-complexity" and simulate the protocol securely against semi-honest adversaries in the plain model. We believe our simulation lemma to be of broader interest.

Keywords: black-box separations, coin tossing, optimally-fair coin tossing, round-complexity, lower-bound.

1 Introduction

We study the fundamental problem of (two-party) coin tossing, where two mutually distrustful parties wish to generate a common random bit. Ideally, this bit should be almost completely unbiased (namely be equal to 1 with probability that is at most negligibly far from 1/2). Furthermore, by the definition of a secure coin tossing protocol, if the two parties follow the protocol then they must both output the same random bit. Unfortunately, however, as shown in a classic result by Cleve [C86], if one of the parties may deviate from the protocol (even if the deviation is only "fail-stop" meaning that the adversary merely aborts early), then secure coin tossing cannot be achieved. In fact, Cleve proved that for any coin tossing protocol running for r rounds there exists an efficient fail-stop adversary that can bias the resulting bit by at least $\Omega(1/r)$.

On the positive side, an early result by Blum [B82] uses one-way functions to construct a coin tossing protocol in a weaker model, where an unbiased output is achieved if both parties complete the protocol, but if a malicious party aborts early, the honest party does not output any bit. This protocol was used by Cleve [C86] to construct a coin tossing protocol that runs for r rounds and for which no efficient adversary can bias the output bit by more than $O(1/\sqrt{r})$ assuming that one-way functions exist.[4]

This gap between the lower and upper bounds in [C86] remained open for more than two decades. Recently, it was closed by Moran et al. [MNS09], who constructed a protocol for coin tossing that matches the lower-bound of [C86]. Specifically, they constructed an $O(r)$-round protocol with the property that no adversary can bias the output by more than $O(1/r)$. Thus, they demonstrated that the $\Omega(1/r)$ lower-bound is tight. We call such a protocol *optimally-fair* because no protocol can achieve lower bias.

Interestingly, the protocol of [MNS09] uses general secure computation and thus requires the assumption that oblivious transfer exists (or any assumption implying it, like enhanced trapdoor permutations). In contrast, the coin tossing protocol of Blum [B82] and the protocol of [C86] achieving bias of $O(1/\sqrt{r})$ can be constructed from any one-way function. This disparity was observed by [MNS09] who state: *"A challenging problem is to either achieve the optimal bias based on seemingly weaker assumptions (e.g., one-way functions), or to demonstrate that oblivious transfer is in fact essential."*

In this paper we take a step toward answering this question, and show that one-way functions are not sufficient for achieving optimally-fair coin tossing via *black-box reductions* when the number of rounds r is $o(n/\log n)$ for security parameter n (i.e., the input/output length of the one-way function). We note that the protocols mentioned above of [C86,MNS09] are indeed black-box

Theorem 1 (Main Theorem, Informal). *Let Π be a black-box construction for two-party optimally-fair coin tossing based on one-way functions with input and output length n. Then the number of rounds r of interaction in Π is at least $r = \Omega(n/\log n)$.*

[4] Essentially, this protocol works by running Blum's protocol r times sequentially and outputting the bit that appeared in most executions. (If one of the parties halts prematurely, then the other party takes locally chosen uniformly distributed bits as the output bits for the remaining Blum executions.)

In fact, we prove something even stronger: – **Stronger primitives.** The same result holds even if the primitive used in the construction is an exponentially-hard one-way function or an exponentially hard collision resistant hash function $h\colon \{0,1\}^n \mapsto \{0,1\}^{\theta(n)}$ (or in fact any primitive which can be derived in a black-box manner from a random oracle). The result holds also for more structured primitives such as one-way permutation. The latter extension is based on the simple observation that a random function and a random permutation can not be distinguished with "few" queries asked by the construction. We refer the reader for the full proof of these extensions to the full version of the paper. – **Optimality of the bias.** The same result holds even when Π achieves any $o(1/\sqrt{r})$ bias (not only for optimally-fair protocols with a bias of $O(1/r)$).

Our main technical lemma in order to prove Theorem 1 is to show how to remove random oracles from certain secure protocols in the random oracle models which we believe to be of independent interest.

Lemma 1 (Simulation Lemma, Informal). *Let Π be a two-party protocol in the random oracle model in which the parties query a (random) oracle of input/output length n, ask a total of $m = \operatorname{poly}(n)$ queries and communicate for $o(n/\log n)$ rounds. Then there are two protocols: Π_E (the extended protocol) and Π_T (the threshold-simulation protocol) such that the following holds.* **(a)** *In Π_E the parties act as Π but the ask up to $2^{o(n)}$ extra queries from the oracle.* **(b)** *Π_T is performed in the plain model without the random oracle.* **(c)** *The joint views of the parties in Π_E and Π_T are λ-close for an arbitrary parameter $\lambda = 1/\operatorname{poly}(n)$.*

The high level structure of the proof of Theorem 1 is to use the simulation lemma and the result of [CI93] which breaks any coin-tossing protocol in the plain model with "few" rounds. See Section 1.1 for more details.

We also observe that our simulation lemma can be used to derive impossibility results in the context of secure two-party computation of non-trivial functions. Kushilevits [K92] classified the finite functions that have perfectly secure two-party protocols against semi-honest adversaries and called them "decomposable functions". Maji, Prabhakaran and Rosoulek [MPR09] extended this result to the regime of statistical security and showed that only decomposable functions can have (randomized) two-party protocols which are statistically secure against semi-honest parties. The latter result together with our simulation lemma imply that if a function is not decomposable, it can not have a black-box secure protocol based on one-way function (or based on the other primitives mentioned above) with $o(n/\log n)$ rounds of communication. The steps of the proof of this result are very similar to the case of coin-tossing described in Theorem 1. See Section 1.1 for more details and see the full version of the paper for the complete proof.

Discussion and Implications. Our lower-bound proves that either there is no black-box construction of optimally-fair coin tossing from any of the primitives mentioned in Theorem 2, or if there is any such construction it will suffer from an almost linear $\widetilde{\Omega}(n)$ lower-bound on its round-complexity (which arguably is the main efficiency measure) depending on the security parameter of the primitive used. Such a construction, where the number of rounds, and thus the bias, depends on the security parameter, seems counter-intuitive (yet see the comparison below with statistically hiding commitments which do have constructions with the number of rounds depending on the security parameter).

In particular, our negative result implies that the use of oblivious transfer (as an assumption stronger than one-way function) in the construction of [MNS09], achieving $O(1/r)$ bias for any r, is *inherent.* Moreover, the construction of [C86], using commitments (that can be constructed in a black-box way from one-way functions) and achieving $O(1/\sqrt{r})$ bias for any r, is actually *optimal* (as Theorem 2 holds for any $o(1/\sqrt{r})$ bias).

It is also interesting to contrast our lower bound with the original impossibility result of Cleve [C86]. One way to view the result of [C86] is as a proof that in order to achieve $O(1/r)$ bias any protocol must have at least $\Omega(r)$ rounds of interaction. Our lower bound then says that it is only possible to achieve $O(1/r)$ bias with r rounds when relying on one-way functions (or any of the primitives mentioned in Theorem 2) for $r = \Omega(n/\log n)$ which is very large. In particular, it is not possible to construct a protocol (using a black-box reduction) whose round efficiency depends *only* on the desired bias and is independent of the security parameter n used to determine the input length to the one-way function. This has the ramification that increasing the security parameter in order to obtain a stronger guarantee of invertibility of the one-way function (to get a more secure protocol) has an effect also on the round complexity of the protocol.

Black-Box Separations. One of the main goals of modern cryptography has been to identify the minimal assumptions necessary to construct secure cryptographic primitives. For example, [Y82,GM84,R90,HILL99,GGM86,LR88,IL89,NY89,N91] have shown that private key encryption, pseudorandom generators, pseudorandom functions and permutations, bit commitment, and digital signatures exist if and only if one-way functions exist. On the other hand, some cryptographic primitives such as public key encryption, oblivious transfer, and key agreement are not known to be equivalent to one way functions. Thus, it is natural to ask whether the existence of one-way functions implies these primitives. However, it seems unclear how to formalize such a question; since it is widely believed that both one-way functions and public key encryption exist, this would imply in a trivial logical sense that the existence of one-way functions implies the existence of public key encryption. Thus, we can only hope to rule out restricted types of constructions that are commonly used to prove implications in cryptography. Impagliazzo and Rudich [IR89] were the first to develop a technique to rule out the existence of an important class of reductions between primitives known as black-box reductions. Intuitively, this is a reduction where the primitive is treated as an oracle or a "black-box". There are actually several flavors of black-box reductions (fully black-box, semi black-box and weakly black-box [RTV04]). In our work, we only deal with fully black-box reduction, and so we will focus on this notion here. Informally, a fully black-box reduction from a primitive $\mathcal{Q}$ to a primitive $\mathcal{P}$ is a pair of *oracle* PPT Turing machines (G, S) such that the following two properties hold:

Correctness: For every implementation f of primitive $\mathcal{P}$, $g = G^f$ implements $\mathcal{Q}$.
Security: For every implementation f of primitive $\mathcal{P}$, and every adversary A, if A breaks G^f (as an implementation of $\mathcal{Q}$) then $S^{A,f}$ breaks f. (Thus, if f is "secure", then so is G^f.)

We remark that an *implementation* of a primitive is any specific scheme that meets the requirements of that primitive (e.g., an implementation of a public-key encryption scheme provides samplability of key pairs, encryption with the public-key, and decryption with the private key). Correctness thus states that when G is given oracle access to any valid implementation of $\mathcal{P}$, the result is a valid implementation of $\mathcal{Q}$. Furthermore, security states that any adversary breaking G^f yields an adversary breaking f. The reduction here is *fully* black-box in the sense that the adversary S breaking f uses A in a black-box manner.

Comparison to Similar Lower-Bounds on the Round-Complexity. The only similar lower-bound on the round-complexity of black-box constructions that we are aware of is the result of Haitner, Hoch, Reingold, and Segev [HHRS07] which deals with the round-efficiency of statistically hiding commitment schemes. Interestingly, our lower-bound is exactly the same as that of [HHRS07] which also is based on the security parameter of the one-way function used in the construction . It seems that the techniques used in [HHRS07] and our techniques explained below are quite different. This raises the question of whether there are more connections between the two results. For instance, is it possible to simplify any of these arguments using ideas from the other work? More importantly, this suggests the intriguing possibility that perhaps a positive solution for optimally-fair coin tossing from one-way functions can be achieved with $O(n/\log n)$ rounds, using the techniques which are used in constructing the positive results of $O(n/\log n)$-round statistically hiding commitments [NOVY98,HR07,HNO+09].

1.1 Our Technique

We recall a result of Cleve and Impagliazzo [CI93] which shows that for any coin tossing protocol with r rounds, there exists a *computationally unbounded* adversary who can achieve bias of at least $\Omega(1/\sqrt{r})$. Moreover, this adversary follows the protocol as specified, except that it may abort prematurely; as such the adversary is fail-stop. We show that a black-box construction of an $o(n/\log n)$-round coin tossing from own-way functions with input/output length n (or in fact any primitive which is implied by a random-function in a black-box way) will essentially suffer from the same attack of [CI93] and thus cannot guarantee any bias below $\Omega(1/\sqrt{r})$ through a black-box proof of security.

We start by assuming that there is a black-box construction Π of optimally-fair coin tossing from one-way function with $r = o(n/\log n)$ rounds. A random function is one-way with overwhelming probability, so informally speaking, if we feed the construction Π with a random function it should still be an optimally-fair coin tossing protocol. In fact, something stronger happens when a construction based on one-way function is fed with a random function: Such a construction will now be secure even against computationally *unbounded* adversaries who are allowed to ask $2^{o(n)}$ oracle queries to the random oracle. The reason for this is that if there were such an adversary, then the security reduction will imply that there is an adversary inverting a random function with $2^{o(n)}$ number of queries (see the proof of Theorem 2 for more details) which is not possible. We will take advantage of this stronger property to derive the contradiction by presenting a $2^{o(n)}$-query attack whenever the round complexity is $o(n/\log n)$. The idea of feeding a black-box construction with a random-function and enhancing its security, and then deriving contradiction by a simple counting argument (rather than refuting the relativizing reductions [IR89]—which is a much harder task) is also employed in previous works such as [GGKT05,BM07].

Our main technical step will be to show that the round-complexity of $o(n/\log n)$ for the black-box construction of coin tossing implies the existence of a $2^{o(n)}$-query adversary who is able to bias the output bit by $\omega(1/r)$. In fact we show how to achieve bias $\Omega(1/\sqrt{r}) = \omega(1/r)$. The existence of such an attack implies the result because by the security reduction the ability to bias the protocol yields an adversary inverting the one-way function. Our $2^{o(n)}$-query attacker runs the protocol (of the corresponding party) honestly except that it gathers more information about the random oracle along the execution of the protocol by asking $\mathrm{poly}(n,r)^r$ (which is $2^{o(n)}$ for $r = o(n/\log n)$) more queries and achieves bias of $\Omega(1/\sqrt{r})$ by deciding to stop at some point during the protocol.

We shall emphasize that the reason that we can *not* directly use the attack of [CI93] in the presence of a random oracle is that, even conditioned on the transcript of the interaction, the random oracle builds *dependencies* between the views of Alice and Bob. However the attack of [CI93] essentially uses the fact that conditioned on the transcript the views of Alice and Bob are independent in a plain protocol (where no random oracle is used). Thus we need to find a way to "kill" this dependency to be able to use their attack.

Our $2^{o(n)}$-query attacker uses special properties of an attack given by Barak and Mahmoody [BM09] to break any key-agreement protocol with an optimal number of queries to the random oracle. The attacker of [BM09]—which here we call the "independence learning algorithm", or the simply the learning algorithm for short—gets as input a threshold parameter ε which controls its efficiency and accuracy at the same time. Roughly speaking if Alice and Bob ask m oracle queries in their execution, it will lead to $O(m/\varepsilon)$ queries asked by the learner and the error of $m\varepsilon$. This learning algorithm can be described more naturally as an *online* algorithm which learns certain oracle queries during the interaction between Alice and Bob (despite the fact that passive adversaries can always wait till the end of the interaction). Our attacker uses this learning algorithm internally and feeds it with *different* values for the threshold parameter ε for each round; the parameter ε taken grows exponentially with the round numbers. Due to the heavy use of the threshold parameter of the learning algorithm in our attack, we call it the "threshold attacker" TA. Note that since the learning algorithm only requires the knowledge of the public transcripts, both Alice and Bob can run the learning algorithm in any two-party protocol (e.g., a coin tossing protocol rather than a key-agreement protocol). Thus our threshold attacker

TA, which is in fact executed by either Alice or Bob, can also run the learning algorithm during the coin tossing protocol.

The Threshold Attacker—More Details. For an arbitrary two-party protocol Π in the random oracle model (or any other oracle model) we can think of "curious" parties who run the protocol honestly but will ask more oracle queries along their execution of the protocol[5]. We use the terminology of [GIMS10] and call such a game a *curious extension* of the original protocol Π. To get the threshold attacker, Alice or Bob (whoever is performing the attack) will need to play a curious extension of the original protocol by asking up to $2^{o(n)}$ oracle queries. Here we will only deal with an extension based on the learning algorithm of [BM09]. That is, the attacking party runs the learning algorithm along the honest execution of the original coin-tossing protocol and decides to abort prematurely. We let the parties take turn in simulating the learning algorithm in the following way: Whenever Alice (or Bob) is sending a message w_i, they attach to it the set of query/answer pairs that the learning algorithm would learn after w_i is sent across the channel. For brevity we call this specific curious extension in which both Alice and Bob run the learning algorithm along the original game (and attach the learner's view of each round to their messages) simply "the extended execution" of the original protocol (without referring to the learning algorithm explicitly). We show how our threshold attacker can perform their attack in the extended execution.

We prove that the extended protocol has the interesting property that now Alice and Bob can in fact "simulate" the random oracle on their own (using their private randomness) in a way that their views are statistically close to those in the execution of the original extended game in the random oracle model. To perform the simulation, Alice and Bob will answer their queries to the random oracle using fresh randomness unless they have asked this query at some point before (and thus chose the answer already) or that they are told by the other party what the answer to this query should be (through the extra messages simulating the learner's view).

To prove that the above simple simulation is indeed a statistically-close simulation of the extension game we need to show that (unless with small probability) there is no inconsistencies between the oracle answers chosen by Alice and Bob for their oracle queries. Here we crucially use the fact that the learning algorithm provides enough information along the game so that Alice and Bob will always choose consistent oracle answers for their queries. Suppose that Alice is sending a message w_i and is also attaching a list of $k \approx m/\varepsilon_i$ simulated learning queries to the message w_i where ε_i is the learner's threshold used in round i by Alice and m is the total number of queries in the original protocol. For any query q among these k queries which are being asked by Alice from the random oracle (and thus being simulated) for the first time, we want that q is *not* among Bob's "private"

[5] This is slightly different from the semi-honest parties who run the protocol honestly without asking more oracle queries and only later analyze their view of the interaction.

queries which was simulated at some point before (yet is not announced through the learner's simulation). The learner's algorithm has the property that if Bob uses threshold ε_{i-1} to simulate the learner in the previous round $i-1$ then any such query q has chance of at most ε_{i-1} to be a "private" query of Bob. Therefore, by a union bound, the probability that any of these k queries cause an inconsistency is at most $\approx k\varepsilon_{i-1} = m\varepsilon_{i-1}/\varepsilon_i$. By taking $\varepsilon_{i-1} \ll \varepsilon_i/m$, we can control the probability of such event to be arbitrary small. This clarifies why we end up using exponentially smaller thresholds for smaller rounds.

Finally, since we could simulate the extended execution through a plain protocol, we can use the *inefficient* attack of [CI93], which can be applied to any plain protocol and apply it to the simulation of the extension game. Since the extended execution and its simulation are statistically close experiments, we conclude that almost the same bias would be achieved by the attacker in the extension execution with only $2^{o(n)}$ queries and so we are done.

A Parallel Work. The threshold simulation technique was discovered independently in a parallel work by Maji and Prabhakaran [MP10] in the context of using random oracle for the aim of achieving statistically secure protocols.

2 Definitions and Useful Lemmas

Definition 1 (coin tossing from one-way function). *For (interactive) oracle algorithms A, B we call $\Pi = (A, B)$ a black-box construction of coin tossing with bias at most δ based on exponentially-hard one-way functions with security parameter n, if the following properties hold:*

– *A and B have their own private randomness R_A, R_B. They take as input 1^n and run in time $\mathrm{poly}(n)$ and interact for $r(n) = \mathrm{poly}(n)$ number of rounds.*
– **Completeness:** *For any function $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, when A and B are given oracle access to f, then at the end of the protocol A's output a and B's output b are such that $a = b$ and b is considered the output of the protocol. Also if during the protocol A (resp., B) receives the special message $\perp$ (denoting that the other party has stopped playing in the protocol) then A (resp., B) outputs a bit a (resp b) on their own which is considered as the output of the protocol.*
– **Security (against bias δ):** *There is an oracle algorithm S running in time $2^{o(n)}$ with the following property. For any $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ given as oracle, if $\widehat{A}$ (resp., $\widehat{B}$) is a malicious interactive algorithm interacting with B (resp., A) which makes the output bit b to be $\delta(n)$-biased, then $S^{f,\widehat{A}}$ (given oracle access to f and $\widehat{A}$) breaks the security of f (as an exponentially-hard one-way function).*

We denote by $(a|b) \leftarrow \langle \widehat{A}, B\rangle$ (resp. $(a|b) \leftarrow \langle A, \widehat{B}\rangle$) the joint output of $\widehat{A}$ and B (resp. A and $\widehat{B}$) generated by an interaction of $\widehat{A}$ and B (resp. A and $\widehat{B}$).

The proof of the following two lemmas can be verified by inspection.

Lemma 2 (Inverting Random Functions). *Let A be a* computationally unbounded *oracle algorithm given oracle access to a random function $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ (the randomness of f is chosen after A is fixed). Then if A asks at most $2^{\alpha n}$ queries from f, the probability that A can successfully invert a given input $y = f(U_n)$ (to any preimage of y) is at most $2 \cdot 2^{(\alpha-1)n} + 2^{-n}$ which is negligible for any constant $\alpha < 1$.*

Lemma 3 (Inverting Random Functions with a Fixed Subdomain). *Let $S \subset \{0,1\}^n$ be of size $|S| \leq 2^{\beta n}$ for $\beta < 1$, and let $f_S\colon S \mapsto \{0,1\}^n$ be a fixed function. Let F be the set of all functions $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ which are equal to f_S over S. Now, let A be a* computationally unbounded *oracle algorithm which can depend on f_S and is given oracle access to a random function $f \overset{R}{\leftarrow} F$ (the randomness of f is chosen after A is fixed). Then if A asks at most $2^{\alpha n}$ queries from f, the probability that A can successfully invert a given input $y = f(U_n)$ (to any preimage of y) is at most $2 \cdot (2^{(\alpha-1)n} + 2^{(\beta-1)n}) + 2^{-n}$ which is negligible for any constants $\alpha < 1$ and $\beta < 1$.*

3 Simulation Lemma

In this section, we present a general lemma that holds for any two-party protocol in the random oracle model. This lemma will be useful for proving our result on coin tossing, but also has applications to general two-party computation as we describe below.

Lemma 4 (Simulation Lemma). *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and interact for r rounds. Then there exist protocols Π_T and Π_E called the λ-threshold simulation and λ-extended execution of Π such that the views of Alice and Bob (as a jointly distributed random variable) in Π_T and Π_E are λ-close. Moreover, the following properties hold:*

- *Π_T makes no oracle queries.*
- *For $\lambda = 1/\operatorname{poly}(n)$, $r = o(n/\log n)$ and $m = \operatorname{poly}(n)$, Π_E makes at most $2^{o(n)}$ queries.*
- *Let $W^{\Pi} = [w_1^{\Pi}, \ldots, w_i^{\Pi}]$ be the sequence of messages sent between Alice and Bob so far in an execution of protocol Π relative to oracle f with random tapes R_A, R_B respectively. For $\lambda = 1/\operatorname{poly}(n)$, $r = o(n/\log n)$ and $m = \operatorname{poly}(n)$, both Alice and/or Bob can make at most $2^{o(n)}$ queries and produce the transcript $W^{\Pi_E} = [w_1^{\Pi_E}, \ldots, w_i^{\Pi_E}]$ that is generated by an execution of the protocol Π_E relative to oracle f with random tapes R_A, R_B.*

The above lemma implies the following corollary:

Corollary 1. *Let $p = 1/\operatorname{poly}(n)$ and let Q be some two-party cryptographic task such that for every implementation Π_{plain} in the plain model with $r = o(n/\log n)$ rounds, there is a computationally-unbounded, semi-honest adversary* *which breaks the security of Π_{plain} with probability p. Let Π be a black-box construction of Q with r rounds based on exponentially-hard one-way functions with security parameter n (i.e. the input/output length of f). Then $r = \Omega(n/\log n)$.*

The corollary follows from Lemma 4 due to the following: Assume such a construction Π exists with $r = o(n/\log n)$ rounds. Now consider Π_T, the λ-threshold simulation of Π. Since Π_T also has $r = o(n/\log n)$ rounds and does not make calls to the oracle, we have by hypothesis that there is an unbounded attacker $\widehat{A}$ (resp. $\widehat{B}$) which breaks the security of Π_T with probability $p = 1/\operatorname{poly}(n)$. Now, for $\lambda \leq p/2 = 1/\operatorname{poly}(n)$, we have that the views of Alice and Bob (as a jointly distributed random variable) in Π_T and in the λ-extended exection, Π_E, are λ-close. Moreover, given the transcript generated by Π, Alice (resp. Bob) can make at most $2^{o(n)}$ queries and produce the corresponding transcript of Π_E. Thus, there is a threshold attacker TA which plays the part of Alice (resp. Bob) in Π, makes at most $2^{o(n)}$ queries to compute the messages of Π_E, runs $\widehat{A}$ (resp. $\widehat{B}$) internally while simulating the view of $\widehat{A}$ (resp. $\widehat{B}$) using the λ-close view produced by Π_E and finally outputs whatever $\widehat{A}$ (resp. $\widehat{B}$) outputs. So TA breaks the security of Π_E (and thus of Π) with probability $p/2$, where the probability is computed over the randomness of f. Having the threshold attacker TA the proof can be concluded as follows:

(a) Since the attacker TA breaks security with probability $p/2 = 1/\operatorname{poly}(n)$, by an averaging argument, for at least $p/4$ fraction of the functions $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, the attacker TA^f breaks security with probability $p/4$. We call such function f, a good function. **(b)** Using the security reduction S, for all good functions f, S^{f,TA^f} inverts $y = f(U_n)$ with probability at least $2^{-o(n)}$. **(c)** We can combine the algorithms S and TA to get a single oracle algorithm T^f which inverts $f(U_n)$ with probability $2^{-o(n)}$ when f is a good function by asking only $2^{o(n)}$ queries to f. Which means that in this case T asks only $2^{o(n)}$ oracle queries and inverts a *random* f with probability at least $p/4 \cdot 2^{-o(n)} = 2^{-o(n)}$ (because f is a good function with probability at least $p/4$). The latter contradicts Lemma 2.

Before we prove Lemma 4, we review relevant previous work.

The Independence Learner of [BM09]. Here we describe the properties of the attacker of Barak and Mahmoody [BM09] presented in the context of breaking any key agreement protocol with optimal number of queries to the random oracle. Since the main property of the learning algorithm is that conditioned on the learner's information Alice and Bob's views are almost independent, we call this attack the independence learning algorithm.

Lemma 5 (The Independence Learner of [BM09]). *Let Σ be any two-party protocol in the random oracle model (with arbitrary number of rounds) between Alice and Bob in which Alice and Bob ask at most m queries from the random oracle H. Then there is a universal constant c and a (computationally unbounded)* independence learning *algorithm which is given a parameter ε (called the* threshold*) as input and has the following properties. For brevity we denote the independence learning algorithm by Eve.*

– *Eve only has access the public messages sent between Alice and Bob and can ask queries from the random oracle.*
– (cm/ε)**-Efficiency:** *Eve is deterministic and, over the randomness of the oracle and Alice and Bob's private randomness, the expected number of Eve queries from the oracle H is at most cm/ε.*
– *Eve asks its queries* along *the game. Namely, although Eve can wait till the end and then ask all of her queries, her description defines which queries to be asked right after each message is sent across the public channel. So the learning algorithm is divided into the same number of rounds as the protocol.*
– $(c\sqrt{m\varepsilon})$**-Security:** *Let $W = [w_1, \ldots, w_i]$ be the sequence of messages sent between Alice and Bob so far, and let I be the list of oracle query/answer pairs that Eve has asked till the* end *of the i'th round, and let $\mathbf{AB} = (\mathbf{A}, \mathbf{B})$ be the* joint *distribution over the views of Alice and Bob* only *conditioned on (W, I). By $\mathbf{A}$ and $\mathbf{B}$ we refer to the projections of $\mathbf{AB}$ over its first or second components (referring to the view of either Alice or Bob only) as random variables. For a specific view $A \leftarrow \mathbf{A}$ for Alice, by $Q(A)$ we refer to the set of oracle queries that A contains. We also use the notation $Q(I)$ to refer to the queries denoted in I.*
With probability at least $1-c\sqrt{m\varepsilon}$ over the randomness of Alice, Bob, and the random oracle H the following holds at all *moments during the protocol when Eve is done with her learning phase in that round: There are* independent *distributions $\widehat{\mathbf{A}}, \widehat{\mathbf{B}}$ such that:*
  1. *The statistical distance between $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ and $\mathbf{AB}$ is at most $\Delta(\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}, \mathbf{AB}) \leq c\sqrt{m\varepsilon}$.*
  2. *For every oracle query $q \notin Q(I)$, it holds that $\Pr[q \in Q(\widehat{\mathbf{A}}) \cup Q(\widehat{\mathbf{B}})] \leq \varepsilon$.*

– **Robustness.** *The learning algorithm is robust to the input parameter ε in the following sense. If the parameter ε changes in the interval $\varepsilon \in [\varepsilon_1, \varepsilon_2]$ arbitrarily during the learner's execution (even inside a learning phase of a specific round), it still preserves $O(cm/\varepsilon_1)$-efficiency and $(c\sqrt{m\varepsilon_2})$-security.*

Lemma 5 is implicit in [BM09], and we show how to derive it from the explicit results of [BM09] in the full version of the paper.

Given a protocol Π, we now describe the λ-extended execution, Π_E, and the λ-threshold simulation, Π_T, of Π that were mentioned in Lemma 4.

Definition 2 (Extended Execution). *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and interact for r rounds. The extended execution Π_E of Π gets as input a parameter λ and simulates the original protocol Π in the random oracle model as follows.*

– *Let $\varepsilon_r = \frac{1}{m} \cdot \left(\frac{\lambda}{9rc}\right)^2$ and for $j \in \{r, r-1, \ldots, 2\}$ define $\varepsilon_{j-1} = \varepsilon_j \cdot \frac{\lambda^2}{90r^2m}$. Note that if r, λ, m are $\leq \operatorname{poly}(n)$, then $\varepsilon_r = 1/\operatorname{poly}(n)$ and $\varepsilon_1 = \operatorname{poly}(n)^{-r}$.*
– *Now imagine an Eve who runs the independence learner of Lemma 5 and uses ε_i as its learning parameter in the learning phase after the i'th round.*

– *In round i, the party who is sending the message w_i, also runs the i'ih round of the learning phase of Eve and attaches to w_i the list of all the query/answer pairs that are the result of this learning algorithm. Note that since Eve's algorithm is only depending on the messages being sent and her previous knowledge about the oracle, the parties are able to do this job.*

Definition 3 (Threshold Simulation). *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and interact for r rounds. A threshold simulation Π_T of Π gets as input a parameter λ and simulates the original protocol Π* plainly *as follows.*

– *The parameters ε_i for $i \in [r]$ are defined similar to the extended execution.*
– *In the i'th round the party who sends the i'th message tries to simulate the i'th round of the extended execution but* without *using a random oracle. The way the simulation is done is as follows: To compute the message w_i, suppose q is a query to be asked from the oracle. Now if q is in the set of queries learned by Eve so far or if q was asked previously by the same party, the same answer will be returned which was used before. But, if the query q is new, a fresh random answer will be used. The same is also done to answer any query that the learning algorithm Eve tries to learn.*

The following lemma explains why a threshold simulation is indeed a good simulation of the extended execution.

Lemma 6 (Properties of the Threshold Simulation). *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and let Π_T and Π_E be in order its λ-threshold simulation and λ-extended execution. Then the views of Alice and Bob (as a jointly distributed random variable) in Π_T and Π_E are λ-close.*

Proof. It is easy to see that the extended execution and the threshold simulation will be exactly the same games until the following happens: A party, say Alice sends a message w_i along with the simulation of Eve's i'th round, but one of these queries (which are asked in this round either for her own protocol or to simulate Eve) will hit one of Bob's "private" queries which are *not* announced through Eve's previous simulated query/answers. We show that this "bad" event happens with probability at most λ.

Note that by the robustness of the independence learner Eve and by the choice of the (largest) parameter $\varepsilon_r = \frac{1}{m} \cdot \left(\frac{\lambda}{9rc}\right)^2$, Eve's algorithm remains at least $c\sqrt{m\varepsilon} = \lambda/(9r)$ secure in round i. So, except with probability at most $r \cdot \lambda/(9r) = \lambda/9$ we can pretend (as a mental experiment) that at all moments the security requirement of the learning algorithm holds with probability 1 rather than $1 - c\sqrt{m\varepsilon}$. In the following we show that (up to the bad event mentioned above which happens with probability at most $\lambda/9$) the probability that an "inconsistency" happens in round i is at most $\lambda/(3r)$, and thus we will be done by a union bound. By inconsistency we mean that Alice announces (a different) answer for an oracle query that is privately asked by Bob already (or vice versa).

Suppose Alice is sending the message in the i'th round and suppose no inconsistency has happened so far. Let fix $W = [w_1, \ldots, w_{i-1}]$ to be the sequence of the messages sent till this moment and let I be the union Eve's simulated queries till the end of the $(i-1)$'th round. An inconsistency in round i can happen as follows: one of the queries asked by Alice (either to run her own protocol or to simulate Eve) hits one of Bob's private queries. We bound this probability conditioned on any fixed (W, I) over which the security property of the learner holds (as we said this property will hold with probability at least $1 - \lambda/9$).

As a mental experiment we can continue the game (after fixing (W, I)) by sampling from the random variable $(A, B) \leftarrow \mathbf{AB}$ for the views of Alice and Bob so far conditioned on (W, I) and then continue Alice's simulation. Let assume for a moment that we sample $(A, B) \leftarrow \widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ rather than from $\mathbf{AB}$. We bound the probability of any inconsistency in the former case to be $2\lambda/(9r)$, and since the distributions $\mathbf{AB}$ and $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ are $\lambda/(9r)$ close, it follows that the probability of any inconsistency in this round is bounded by $2 \cdot \lambda/(9r) + \lambda/(9r) = \lambda/(3r)$ which is small enough for us.

But now we use the security property of the independence learner. Note that when we get the sample $(A, B) \leftarrow \widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$, A and B are sampled *independently*. So, we can sample A first, continue Alice's computation, and then sample $B \leftarrow \widehat{\mathbf{B}}$ at the end (and we will abort if the private queries collide). The number of queries that Alice will ask to run her own protocol is at most m. By the efficiency property of the learning algorithm applied to round i, the number of Eve's simulated queries in this round are, on average, at most cm/ε_i. By a Markov bound, this number is at most $\frac{cm}{\varepsilon_i} \cdot \frac{9r}{\lambda}$ with probability at least $1 - \lambda/(9r)$. So except with probability $\lambda/(9r)$ the total number of queries asked by Alice in this round is at most $m + 9cmr/(\varepsilon_j \lambda) < 10cmr/(\varepsilon_j \lambda)$. Note that the probability that any of these $10cmr/(\varepsilon_j \lambda)$ queries are among the private queries of a sample from $\widehat{\mathbf{B}}$ (sampled as Bob's view) is at most ε_{j-1}. So, by a union bound, the probability that at least one of these queries hits $\widehat{\mathbf{B}}$'s private queries is at most $\frac{10cmr}{\varepsilon_j \lambda} \cdot \varepsilon_{j-1} = \lambda/(9r)$ and this finishes the proof.

So, all left to do is to count how many queries are asked by our λ-extended execution Π_E and show that it is (say on average) at most $2^{o(n)}$. This is indeed the case because of the robustness and the efficiency properties of the learning algorithm. The smallest threshold used in our attack is $\varepsilon_1 = \mathrm{poly}(n)^{-r}$ because $\lambda = 1/r$ and $r = \mathrm{poly}(n), m = \mathrm{poly}(n)$. Therefore our attacker asks at most $O(m/\varepsilon_1)$ number of queries on average which for $r = o(n/\log n)$ is at most $O(m/\varepsilon_1) = \mathrm{poly}(n)^r = 2^{o(n)}$.

4 Proof of the Main Theorem

In this section we first prove our main theorem for the case of exponentially-hard one-way function as the primitive used. Extending the proof to stronger primitives implied by a random oracle is discussed at the end.

Theorem 2 (Main Theorem, Formal). *Let Π be a black-box construction for two-party coin tossing (between Alice and Bob) with bias at most $o(1/\sqrt{r})$ (where r is the number of rounds in Π) based on exponentially-hard one-way functions with security parameter n (i.e., the input/output length of f). Then $r = \Omega(n/\log n)$.*

Proof. For sake of contradiction let assume that such construction exists with $r = o(n/\log n)$ round complexity. The proof goes through the following steps. We first feed Alice and Bob's protocols in the construction Π with a *random* function $f\colon \{0,1\}^n \mapsto \{0,1\}^n$. We show that in that setting at least one of the parties can ask $n^{O(r)}$ queries to f and bias the output by at least $\Omega(1/\sqrt{r})$ by a fail-stop attack. The probability over which the bias is computed also includes the randomness of f. As in Section 3, we call this attacker the threshold attacker, TA. Having the threshold attacker TA the proof can be concluded as follows.

(a) Since the attacker TA achieves bias $\delta = \Omega(1/\sqrt{r})$ and since the bias is always $\delta < 1$, therefore by an averaging argument, for at least $\delta/2$ fraction of the functions $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, the attacker TA^f achieves bias at least $\delta/2 = \Omega(1/\sqrt{r})$. We call such function f, a good function. **(b)** Using the security reduction S, for all good functions f, S^{f,TA^f} inverts $y = f(U_n)$ with probability at least $2^{-o(n)}$. **(c)** We can combine the algorithms S and TA to get a single oracle algorithm T^f which inverts $f(U_n)$ with probability $2^{-o(n)}$ when f is a good function by asking only $2^{o(n)}\operatorname{poly}(n)^r$ queries to f. For $r = o(n/\log n)$, it holds that $\operatorname{poly}(n)^r = 2^{o(n)}$, which means that in this case T asks only $2^{o(n)} \cdot 2^{o(n)} = 2^{o(n)}$ oracle queries and inverts a *random* f with probability at least $\frac{\delta}{2} \cdot 2^{-o(n)} = 2^{-o(n)}$ (because f is a good function with probability at least $\delta/2$). The latter contradicts Lemma 2.

In the following we first describe the results that we borrow or derive from previous work needed for our threshold attacker TA, and then will describe and prove the properties of TA.

The Fail Stop Attacker of [CI93]*.* Cleve and Impagliazzo [CI93] showed that when computationally unbounded parties participate in any coin tossing protocol, at least one of them can bias the output bit by following the protocol honestly and aborting at some point based on the information provided to them by their view.

Lemma 7 (The Attacker of [CI93]). *Let Σ be any two-party protocol for coin tossing between Alice and Bob with r rounds of interaction. Then either Alice or Bob can bias the output bit by $\Omega(1/\sqrt{r})$ in the fail-stop model through a computationally* unbounded *attack.*

4.1 Our Threshold Attacker

In this section we use the attack of Lemma 7 as well as the results of Section 3 to finish the proof of Theorem 2 by presenting our threshold attacker. We will do so first in a special case where the protocol Π is of a special form which we call *instant*. The case of instant constructions carries the main ideas of the proof. Later we prove Theorem 2 for constructions which are not necessarily instant.

Definition 4 (Instant Constructions). *A black-box construction of coin tossing is an* instant *construction if whenever a party aborts the protocol, the other party decides on the output bit* without *asking any additional queries to its oracle.*

We note that the protocol of Cleve [C86] which achieves bias at most $O(1/\sqrt{r})$ based on one-way function is in fact an instant construction.

Given an instant coin-tossing protocol Π, we apply Lemma 4 to obtain the λ-threshold simulation and λ-extended execution of Π, Π_T, Π_E. Since the threshold simulation, Π_T, is a plain protocol we can apply Lemma 7 to get an attack of bias $\Omega(1/\sqrt{r})$ by either Alice or Bob. Now if we take the simulation parameter λ to be at most $1/r = o(1/\sqrt{r})$, then the same exact attack will also give a bias of $\Omega(1/\sqrt{r}) - o(1/\sqrt{r}) = \Omega(1/\sqrt{r})$ in the extended execution. Here we crucially rely on the instant property because of the following: As soon as Alice or Bob (who is the attacker) stops continuing the game, the other party in the threshold simulation will decide on the final output bit by looking at their current view. But this last step will not be statistically close between the extended execution and the threshold execution if in the extended execution the deciding party chooses the output after asking more queries. In other words, if the party who announces the output bit (not the attacker) wants to ask more oracle queries to compute the output bit, there should be some simulated random answers chosen by the corresponding party in the threshold simulation to on behalf of these queries, but that step is not taken care of by Lemma 6 (because the aborted party is *not* given the learning algorithm's queries for the aborted round). By Lemma 4, our attacker asks at most $2^{o(n)}$ queries.

Before going over how to handle the non-instant constructions we clarify that extending Theorem 2 to stronger primitives such as exponentially-hard collision resistant hash function is immediate. All one has to do is to substitute the collision resistant hash functions $h\colon \{0,1\}^n \mapsto \{0,1\}^{n/2}$ used in the construction by a random function $f\colon \{0,1\}^n \mapsto \{0,1\}^{n/2}$ (which is in fact a $2^{\Omega(n)}$-secure hash function). To provide access to a family of hash functions one can use the random oracle over larger domains of input/output length $3n$ and use the first n bits of the input as the index to the hash family and simply throw away the last $\frac{5n}{2}$ bits of the output. The rest of the proof remains the same.

Handling Non-instant Constructions It is instructing to recall that given a random oracle there is indeed a one-round protocol which is optimally-fair: Alice asks $H(0)$ (assuming that the random oracle is Boolean) and then sends $H(0)$ to Bob which is the final output bit. If Alice aborts and does not send $H(0)$, Bob will go ahead and ask $H(0)$ himself and takes that as the final output bid. It is clear that this trivial protocol is completely fair because $H(0)$ is an unbiased bit. Also note that the proof of the previous section handing the instant constructions works just as well for protocols which use a truly random oracle (rather than a one-way function) as their primitive used. So it should be of no surprise that the proof of the instant case does not immediately generalize to cover all the black-box constructions (the trivial coin-tossing protocol based on random oracle is clearly a non-instant protocol). To handle the non-instant constructions we inherently need to use the fact that the constructions we deal with are optimally-fair protocols given any *one-way* function as the primitive used. In the following we show how this stronger requirement of the construction gives us what we need in Theorem 2.

Making constructions almost instant. It is easy to see that any construction for coin tossing can be changed into an equivalent protocol which is "almost" an instant one. Namely, whenever a party A is sending a message m, it can also consider the possibility that the other party B will abort the game right after A sends his message. So, during the computation of m, A can go ahead and ask whatever query from the oracle which is needed to compute the final bit in case B aborts. This way, A will not need to ask any oracle queries in case B aborts in this round. By doing this change (which clearly does not affect the security of the protocol) the construction becomes "almost" instant. The point is that the receiver of the *first* message can not follow the change suggested here because they do not send any message before the first round. Therefore, in the following we only consider constructions which are "almost-instant" (i.e., the only moment that a party might violate the instant property is when the sender of the first message aborts the protocol, and the receiver might still need to ask oracle queries before deciding on the output.)

Handling almost-instant constructions. Suppose Π is an almost-instant construction. Suppose Π_E and Π_T be in order Π's extended execution and the threshold simulation games. The proof of Lemma 6 shows that if no party aborts the experiments Π_E and Π_T are λ-close. The discussion following the proof of Lemma 6 shows that if one of the parties runs the same fail-stop attack in Π_E and Π_T the experiments are *still* λ-close conditioned on the assumption that the round in which the abort happens is any round other than the first one. So, all we need to handle is the case in which the sender of the first message (which we assume to be Alice) aborts the game in the first round (after asking some oracle queries). In the following we focus on this specific cease.

Note that when aborted in the first round Bob can *not* simply simulate the extended execution by using fresh randomness to answer his oracle queries in order to decide the output bit. If he does so it might not be consistent with Alice's queries asked before aborting and thus it will not be a good simulation.[6] Despite this issues, if we are somehow magically guaranteed that when aborted in the first round, none of Bob's queries to compute the output bit collides with Alice's queries asked before, then we can still use fresh random answers to answer Bob's queries to compute the output bit.

Suppose after Alice computes her message but *right before* she sends this message we run the independence learning algorithm with parameter $\lambda/(10m)$.

[6] This will be more clear if one consider the trivial protocol mentioned above which uses a truly random oracle. If Alice aborts whenever $H(0) = 0$, and if Bob uses a fresh random answer whenever he gets aborted by Alice, then the final output will be equal to 1 with probability 3/4 which is clearly a huge bias!

This learning algorithm will announce a set of $O(10m^2/\lambda)$ queries and answers conditioned on which any other query has a chance of at most $\lambda/(10m)$ of being asked by Alice in her computation of the first message. Let the set S be the set of all these $O(10m^2/\lambda)$ queries and let $f(S)$ be their answers. By the security property of the learning algorithm, conditioned on S and $f(S)$, an aborted Bob will not ask any query out of S which collides with Alice's private queries out of S before aborting (unless with probability at most $O(\lambda)$).

The idea is to sample the set S and $f(S)$ once for all, and hardwire them into the random oracle and Alice and Bob's algorithms. This way, simulating Bob's queries with random answers after being aborted will not lead to any inconsistency with Alice's queries unless with probability at most $O(\lambda)$. But if we *fix* the answer of such queries that might hurt the protocol's fairness. At this point we use the fact that the construction is supposed to be fair given any one-way function (and not necessarily a random function). Any random oracle is one-way with overwhelming probability even if we fix a subdomain $S \subseteq \{0,1\}^n$, $|S| \leq \text{poly}(n)$ of its domain and this idea is formalized in Lemma 3. Namely, if we hardwire the random function over a subdomain $S \subseteq \{0,1\}^n$, $|S| \leq \text{poly}(n)$ we can still use the same exact proof as the case of instant constructions for Theorem 2 with the only difference that now we will use Lemma 3 rather than Lemma 2.

Acknowledgement.

We thank the anonymous reviewers for numerous valuable comments.

References

[B82] M. Blum. Coin flipping by telephone - a protocol for solving impossible problems. In *COMPCON*, pages 133–137, 1982.

[BM07] B. Barak and M. Mahmoody. Lower bounds on signatures from symmetric primitives. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2007.

[BM09] B. Barak and M. Mahmoody. Merkle puzzles are optimal - an O(n^2)-query attack on any key exchange from a random oracle. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 374–390. Springer, 2009.

[C86] R. Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *STOC*, pages 364–369, 1986.

[CI93] R. Cleve and R. Impagliazzo. Martingales, collective coin flipping and discrete control processes. Unpublished, 1993.

[GGKT05] Gennaro, Gertner, Katz, and Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SICOMP: SIAM Journal on Computing*, 35, 2005.

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

[GIMS10] V. Goyal, Y. Ishai, M. Mahmoody, and A. Sahai. Interactive locking, zero-knowledge PCPs, and unconditional cryptography. In T. Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 173–190. Springer, 2010.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[HHRS07] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *FOCS*, pages 669–679, 2007.

[HILL99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HNO$^+$09] I. Haitner, M.-H. Nguyen, S. J. Ong, O. Reingold, and S. Vadhan. Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. *SIAM Journal on Computing*, 39(3):1153–1218, 2009.

[HR07] I. Haitner and O. Reingold. A new interactive hashing theorem. In *IEEE Conference on Computational Complexity (CCC)*, 2007.

[IL89] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *FOCS*, pages 230–235, 1989.

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *STOC*, pages 44–61, 1989.

[K92] E. Kushilevitz. Privacy and communication complexity. *SIAM J. Discrete Math*, 5(2):273–284, 1992.

[LR88] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, 1988.

[MNS09] T. Moran, M. Naor, and G. Segev. An optimally fair coin toss. In *TCC*, pages 1–18, 2009.

[MP10] H. Maji and M. Prabhakaran. Personal communication. 2010.

[MPR09] H. K. Maji, M. Prabhakaran, and M. Rosulek. Complexity of multi-party computation problems: The case of 2-party symmetric secure function evaluation. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2009.

[N91] M. Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[NOVY98] Naor, Ostrovsky, Venkatesan, and Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *JCRYPTOL: Journal of Cryptology*, 11, 1998.

[NY89] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.

[R90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[RTV04] O. Reingold, L. Trevisan, and S. Vadhan. Notions of reducibility between cryptographic primitives. In *TCC*, pages 1–20, 2004.

[Y82] A. C.-C. Yao. Theory and applications of trapdoor functions. In *FOCS*, pages 80–91, 1982.