Limits on the Usefulness of Random Oracles

Iftach Haitner[1,⋆], Eran Omri[2,⋆,⋆⋆,⋆⋆⋆], and Hila Zarosim[3,⋆⋆,†]

[1] School of Computer Science, Tel Aviv University
iftachh@cs.tau.ac.il
[2] Dep. of Mathematics and Computer Science, Ariel University Center
omrier@gmail.com
[3] Dep. of Computer Science, Bar Ilan University
zarosih@cs.biu.ac.il

Abstract. In the *random oracle* model, parties are given oracle access to a random function (i.e., a uniformly chosen function from the set of all functions), and are assumed to have unbounded computational power (though they can only make a bounded number of oracle queries). This model provides powerful properties that allow proving the security of many protocols, even such that cannot be proved secure in the standard model (under any hardness assumptions). The random oracle model is also used for showing that a given cryptographic primitive cannot be used in a black-box way to construct another primitive; in their seminal work, Impagliazzo and Rudich [STOC '89] showed that no key-agreement protocol exists in the random oracle model, yielding that key-agreement cannot be black-box reduced to one-way functions. Their work has a long line of followup works (Simon [EC '98], Gertner et al. [STOC '00] and Gennaro et al. [SICOMP '05], to name a few), showing that given oracle access to a certain type of function family (e.g., the family that "implements" public-key encryption) is not sufficient for building a given cryptographic primitive (e.g., oblivious transfer). Yet, the following question remained open:

What is the exact power of the random oracle model?

We make progress towards answering this question, showing that essentially, any no private input, semi-honest two-party functionality that can be securely implemented in the random oracle model, can be securely implemented information theoretically (where parties are assumed to be all powerful, and no oracle is given). We further generalize the above result to function families that provide some natural combinatorial property.

Our result immediately yields that essentially the only no-input functionalities that can be securely realized in the random oracle model (in the sense of secure function evaluation), are the trivial ones (ones that can be securely realized information theoretically). In addition, we use the recent information theoretic impossibility result of McGregor et al. [FOCS '10], to show the existence of functionalities (e.g., inner product) that cannot

⋆ Supported by the Israeli Centers of Research Excellence program (Center No. 4/11).
⋆⋆ Supported by the Israel Science Foundation (ISF) (Grant No. 189/11).
⋆⋆⋆ Research was done while Eran Omri was at Bar Ilan University.
† Hila Zarosim is grateful to the Azrieli Foundation for the Azrieli Fellowship.

A. Sahai (Ed.): TCC 2013, LNCS 7785, pp. 437–456, 2013.

be computed both accurately and in a differentially private manner in the random oracle model; yielding that protocols for computing these functionalities cannot be black-box reduced to one-way functions.

Keywords: random oracles, black-box separations, one-way functions, differential privacy, key agreement.

1 Introduction

In the *random-oracle* model, the parties are given oracle access to a random function (i.e., a uniformly chosen function from the set of all functions — the all-function family), and are assumed to have unbounded computational power (though they can only make a bounded number of oracle queries). Many cryptographic primitives are known to exist in this model, such as (exponentially hard) collision resistant hash functions. More importantly, in this model it is possible to implement secure protocols for tasks that are hard to implement in the standard model, and sometimes even completely unachievable; a well known example is the work of Fiat and Shamir [6], showing how to convert three-message identification schemes to a highly efficient (non interactive) signature scheme. In the random-oracle model, their methodology preserves the security of the original scheme [20], but (for some schemes) does not do so in the standard model [10, 3].

On a different route, the random-oracle model was used to show that one cryptographic primitive *cannot* be used in a black-box way to construct another primitive. In their seminal work, Impagliazzo and Rudich [13] showed that no key-agreement protocol exists in the random oracle model, yielding that key-agreement cannot be black-box reduced to one-way functions. Their work has initiated a long line of follow up works (Simon [22], Gertner et al. [9], and Gennaro et al. [8], to name a few) showing that given oracle access to a certain type of function family (e.g., the family that "implements" public-key encryption) is not sufficient for building a given cryptographic primitive (e.g., oblivious transfer). Yet, the following question remained open:

What is the exact power of the random-oracle model?

Apart from being aesthetic mathematically, answers to this question are very likely to enrich our understanding of (the limitations of) black-box reductions in cryptography.

It is well known that for malicious adversaries, there exist functionalities that cannot be achieved in the *information-theoretic model*, i.e., where all entities are assumed to be unbounded (with no oracle access), yet can be securely computed in the random-oracle model (e.g., commitment schemes, coin-tossing protocols and, non-trivial zero-knowledge proofs). All of these functionalities, however, are blatantly trivial when considering semi-honest adversaries, which are the focus of this work.

1.1 Our Result

We make progress towards answering the above question, showing that, essentially, *any* no private input, semi-honest, two-party computation that can be

securely implemented in the random-oracle model, *can* be securely implemented in the *information-theoretic* model.

Theorem 1 (Main Theorem, Informal). *Let π be a no private-input, m-round, ℓ-query, oracle-aided two-party protocol. Then for any $\varepsilon > 0$ there exists an $O(\ell^2/\varepsilon^2)$-query oracle-aided function* Map*, and a stateless, no oracle, m-round protocol $\widetilde{\pi} = (\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$ such that:*

$$\mathrm{SD}\left(\left(\mathrm{out}_{\mathsf{A}}, \mathrm{out}_{\mathsf{B}}, \mathsf{Map}^f(\overline{t})\right)_{f \leftarrow \mathcal{F}_{\mathrm{AF}}, (\mathrm{out}_{\mathsf{A}}, \mathrm{out}_{\mathsf{B}}, \overline{t}) \leftarrow \langle \mathsf{A}^f, \mathsf{B}^f \rangle}, \left\langle \widetilde{\mathsf{A}}, \widetilde{\mathsf{B}} \right\rangle\right) \in O(\varepsilon),$$

where $\mathcal{F}_{\mathrm{AF}}$ is the all functions family, and $\langle X, Y \rangle$ stands for a random execution of the protocol (X, Y), resulting in the parties' private outputs and the common transcript.

Furthermore, the projections of the above distributions to their first and third coordinates, or to their second and third coordinates are identically *distributed (i.e., the transcripts concatenated with the outputs of one of the parties, are identically distributed).*

Namely, the distributions induced by a random execution of π^f (for a random $f \leftarrow \mathcal{F}_{\mathrm{AF}}$) on the parties' private outputs and the common transcript, is almost the same as that induced by a random execution of the (no oracle) protocol $\widetilde{\pi}$, where the only difference is that one needs to apply an efficient procedure Map to π's transcript. Theorem 1 generalizes to all function families with the property that answers for *distinct* queries, induced by drawing a random member from the family, are independent.

A major ingredient in the proof of Theorem 1 is the *dependency finder* algorithm presented by Barak and Mahmoody [1], refining a similar algorithm by Impagliazzo and Rudich [13] (see Section 1.2). While we could have based the proof of Theorem 1 on a combination of several results from [1] (or alternatively, to get a somewhat weaker variant of the theorem by basing the proof on a followup result of Dachman-Soled et al. [5, Lemma 5] or of Mahmoody et al. [16, Lemma A.1]), we chose to give a new proof also for this part (modulo clearly marked parts taken from [1]). The new proof (given as part of the proof of Lemma 2) holds with respect to a larger set of function families. More significantly, it is more modular and introduces several simplifications comparing to the previous proofs.

Applications. We demonstrate the usefulness of Theorem 1 via the following three examples. The first example (reproving [13, 1]) concerns the existence of key-agreement protocols in the random-oracle model. Recall that key-agreement protocols cannot be realized in the information-theoretic model. Namely, for any (no oracle) protocol π, there exists a passive (i.e., semi-honest) adversary that extracts the key from the protocol's transcript. Hence, Theorem 1 yields that key-agreement protocols cannot be realized in the random-oracle model, and thus key-agreement protocols cannot be black-box reduced to one-way functions. The actual parameters achieved by applying Theorem 1, match the optimal bound given in Barak and Mahmoody [1].

As a second more detailed example, we prove that in the random-oracle model, it is impossible for two parties to accurately approximate the inner-product function in a *differentially private* manner. Namely, in a way that very little information is leaked about *any* single bit of the input of each party to the other party. A recent result of McGregor et al. [17] shows that in the information-theoretic model, it is impossible to approximately compute the inner product function in a differentially private manner. Combining their result with Theorem 1, we obtain the following fact.[1]

Theorem 2 (Informal). *Any ℓ-query, (ℓ^2, α, γ)-differentially private protocol, errs (with constant probability) with magnitude at least $\frac{\sqrt{n}}{\log(n)\cdot e^{\alpha}}$, in computing the inner product of two n-bits strings.*

Very informally, a protocol is (k, α, γ)-differentially private, if no party, making at most k queries to the oracle, learns more then ε information about one of the other party's input bits, except with some small probability γ.

The above result yields the impossibility of fully black-box reducing differentially private protocols for (well) approximating two-party inner-product to the existence of one-way functions. Roughly speaking, such a fully black-box reduction is a pair of efficient oracle-aided algorithms (Q, R) such that the following hold: (1) Q^f is a good approximation protocol of the inner-product for any function f, and (2) $\mathsf{R}^{f,\mathcal{A}}$ inverts f, for any adversary $\mathcal{A}$ that learns too much about the input of one of the parties in Q^f. Since a random sample from the all-function family is hard to invert (cf., [13, 8]), the existence of such a reduction yields that Q^f is differentially-private with respect to poly-query adversaries, when f is chosen at random from the set of all functions.[2] Hence, Theorem 2 yields the following result.

Corollary 1 (Informal). *There exists* no *fully black-box reduction from (α, γ)-differentially private protocol computing the inner product of two n-bit strings with error magnitude less than $\frac{\sqrt{n}}{\log(n)\cdot e^{\alpha}}$, to one-way functions.*

We mention that, following an observation made by McGregor et al. [17], Theorem 2 and Corollary 1 imply similar results for two-party differentially private protocols for the Hamming distance functionality.[3]

The third (and last) example is with respect to no-input secure function evaluation. Let $G = (G_{\mathsf{A}}, G_{\mathsf{B}})$ be a distribution over $\mathcal{A} \times \mathcal{B}$, where G_{A} and G_{B} denote

[1] We mention that the result of [17] is stated for protocol with inputs, where Theorem 1 is only applicable to no-input protocols. Indeed, a fair amount of work was needed to derive an impossibility result for no-input protocols, from the work of [17].

[2] Assume towards a contradiction the existence of a poly-query adversary $\mathcal{A}$ for Q^f, then the poly-query $\mathsf{R}^{f,\mathcal{A}}$ would successfully invert a random f.

[3] The inner product between two bit strings x, y can be expressed as $\mathsf{IP}(x, y) = w(x) + w(y) + H_d(x, y)$, where the weight $w(z)$ is number of 1-bits in z. Thus, a differentially private protocol for estimating the Hamming distance $H_d(x, y)$ can be turned into one for the inner product by having the parties send differentially private approximations of the weights of their inputs.

its marginal distributions over $\mathcal{A}$ and $\mathcal{B}$ respectively. A protocol $\pi = (\mathsf{A}, \mathsf{B})$ is an *information-theoretically* δ-secure implementation of G, if it is a δ-correct (no-oracle) implementation of G (i.e., the local outputs of the parties induced by a random execution of π are δ-close to G), and is δ-private according to the simulation paradigm (against all-powerful distinguishers). Specifically, there exists an algorithm (a simulator) that on input $x \in \mathrm{Supp}(G_{\mathsf{A}})$, outputs a view for A that is δ-close to the distribution of A's view, conditioned on A's local output being x. Similarly, there exists a simulator for B's view. A protocol π is a (T, δ)-secure *random-oracle* implementation of G, if it is a correct random-model implementation of G, and it is δ-private according to the simulation paradigm, against T-query, all-powerful distinguishers. Finally, G is δ-*trivial*, if it has an information-theoretically δ-secure implementation. Theorem 1 yields the following result.

Theorem 3 (Informal). *Let π be an ℓ-query oracle-aided protocol that is an $(O(\ell^2/\delta^2), \delta)$-secure implementation of a distribution G in the random-oracle model. Then, G is $O(\delta)$-trivial.*

Applying Theorem 3 to a distribution G that has a $(\mathrm{poly}(n), 1/\mathrm{poly}(n))$-secure random-oracle model implementation, it follows that G has a $1/\mathrm{poly}(n)$-secure no-oracle implementation. We note that Theorem 3 does not seem to imply the previous two examples. Since, for instance, the notion of differential privacy cannot be realized via the real/ideal paradigm.

1.2 Our Technique

When using a no-oracle protocol to emulate an oracle-aided protocol π, having oracle access to a random member of the all-function family, the crucial issue is to find all *common* information the parties share at a given point. The clear obstacle are the oracle calls: the parties might share information without explicitly communicating it, say by making the same oracle call.

Here comes into play the *Dependency Finder* of Impagliazzo and Rudich [13], and Barak and Mahmoody [1] (algorithm *Eve*, in their terminology). This oracle-aided algorithm (Finder, hereafter) gets as input a communication transcript $\overline{t}$ of a random execution of π, and an oracle access to f, the "oracle" used by the parties in this execution. Algorithm Finder outputs a list of query/answer pairs to f that with high probability contains *all* oracle queries that are *common* to both parties (and possibly also additional ones). Moreover, with high probability Finder is guaranteed not to make "too many" oracle queries.

Equipped with Finder, we give the following definition for the mapping procedure Map and the stateless (no-oracle) protocol $\widetilde{\pi} = (\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$: on a communication transcript $\overline{t}$, the oracle-aided algorithm Map^f outputs $\Big((\overline{t}_1, \mathcal{I}_1 = \mathsf{Finder}^f(\overline{t}_1)), (\overline{t}_{1,2}, \mathcal{I}_2 = \mathsf{Finder}^f(\overline{t}_{1,2})) \ldots, (\overline{t}, \mathcal{I}_m = \mathsf{Finder}^f(\overline{t}))\Big)$. Namely, Map invokes Finder on each prefix of the transcript, and outputs the result. The no-oracle protocol $\widetilde{\pi} = (\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$ is defined as follows: assume that $\widetilde{\mathsf{A}}$ speaks in round $(i+1)$, and that the i'th message is $((\overline{t}_1, \mathcal{I}_1), \ldots, (\overline{t}_{1,\ldots,i}, \mathcal{I}_i))$.

The stateless, no-oracle $\widetilde{\mathsf{A}}$ samples random values for $f \in \mathcal{F}_{\mathrm{AF}}$ and the random coins of A, conditioned on $(\overline{t}_{1,\ldots,i}, \mathcal{I}_i)$ being the protocol's transcript. It then lets t_{i+1} be the next message of A induced by the above choice of f and random coins, and sends $\left(\overline{t}' = (\overline{t}_{1,\ldots,i}, t_{i+1}), \mathsf{Finder}^f(\overline{t}')\right)$ back to $\widetilde{\mathsf{B}}$. In case this is the last round of interaction, $\widetilde{\mathsf{A}}$ locally outputs the (local) output of A induced by this choice of f and random coins. In other words, $\widetilde{\mathsf{A}}$ selects a random view (including the oracle itself) for A that is consistent with the information contained in the no-oracle protocol augmented transcript (i.e., the transcript of the oracle protocol and the oracle calls), and then acts as A would.

The fact that $\widetilde{\mathsf{A}}$ perfectly emulates A (and that $\widetilde{\mathsf{B}}$ perfectly emulates B) trivially holds for information theoretic reasons. For the same reason, it also holds that the transcript generated by applying Map^f to a random transcript of π^f, where $f \leftarrow \mathcal{F}_{\mathrm{AF}}$, generates *exactly* the same transcript as a random execution of $\widetilde{\pi}$ does (actually, the above facts hold for any reasonable definition of Finder[4] and for any function family). The interesting part is arguing that the *joint output* of the no-oracle protocol has similar distribution to that of the oracle-aided protocol. To see that this is not trivial, assume that in the last round both oracle parties make the *same* oracle query q and output the query/answer pair $(q, f(q))$. If it happens that $(q, \cdot) \notin \mathcal{I}$, where $\mathcal{I} = \mathsf{Finder}(\overline{t})$ is the query/answer pairs made by the final call to Finder on transcript $\overline{t}$, then the answer that each of the no-oracle parties compute for the query q might be different. In this case, the joint output of the no-oracle protocol does not look like the joint output of the oracle protocol. Luckily, the above scenario is unlikely to happen due to the guarantee of Finder; with high probability $\mathcal{I}$ contains *all* common queries that the two parties made, yielding that the joint output of the no-oracle protocol has similar distribution to that of the oracle protocol. It turns out that the above example generalizes to any possible protocol, yielding that the above mapping and no-oracle protocol are indeed the desired ones.

1.3 Related Work

In their seminal work, Impagliazzo and Rudich [13] showed that there are no key-agreement protocols in the random-oracle model, and deduce that key-agreement protocols cannot be black-box reduced to one-way functions. This result was later improved by Barak and Mahmoody [1], showing there are no ℓ-query key-agreement protocols in the random-oracle model, secure against adversary making $O(\ell^2)$ queries. Thus, matching the upper bound of Merkle [18].

In an independent work, Mahmoody et al. [16] show that the all-function family (and thus one-way functions) are useless for *secure function evaluation* of deterministic, polynomial input-domain, two-party functionalities. In other words, deterministic, bounded input domain functionalities that *can* be securely computed in the random-oracle model, are the *trivial* ones — functionalities that can be securely computed unconditionally. The comparison to the result stated here is that [16] handle *with input* functionalities, but *only deterministic with*

[4] Whose output contains all queries it made to the oracle.

polynomial input domain, where here we handle *input-less* functionalities, but *including randomized ones*. Putting the two results together, gives a partial characterization of the power of the random-oracle model for (semi-honest) two-party computation. It is still open, however, whether the random-oracle model is useful for securely computing randomized functionalities *with* inputs, or functionalities of super-polynomial input domain.

Additional Black-Box Separations. Following [13], the method of black-box separation was subsequently used in many other works: [21] shows that there does not exist a black-box reduction from a k-pass secret key agreements to $(k-1)$-pass secret key agreements; [22] shows that there exist no black-box reductions from collision-free hash functions to one-way permutations; [14] shows that there exists no construction of one-way permutations based on one-way functions. Other works using this paradigm contain [4, 7, 8, 9, 11, 15, 23], to name a few.

Differential Privacy. Distributed differential privacy was considered by Beimel et al. [2], who studied the setting of multiparty differentially private computation (where an n-bit database is shared between n parties). They gave a separation between information theoretic and computational differential privacy in the distributed setting. The notion of computational differential privacy was considered in Mironov et al. [19]. They presented several definitions of computational differential privacy, studied the relationships between these definitions, and constructed efficient two-party computational differentially private protocols for approximating the Hamming-distance between two vectors. Two-party differential privacy (where an n-bit database is shared between two parties) was considered by McGregor et al. [17]. They prove a lower-bound on the accuracy of two party differentially private protocols, in the information theoretic model, for computing the inner-product between two n-bit strings (and, consequently for protocols for computing the Hamming distance). Hence, proving a separation between information theoretic and computational two-party differentially private computation. In this paper, we extend the lower-bound of [17] to the random-oracle model.

1.4 Open Problems

As mentioned above, the main open problem is the full characterization of the power of the random-oracle model with respect to semi-honest adversaries. Specifically, is it possible to come up with a similar mapping from any (also with inputs) oracle-aided protocol to an equivalent one in the no-oracle model? Another interesting problem is to use our mapping (or a variant of it) to show that the random-oracle model is also useless for protocols (say, input-less) that are secure against fail-stop adversaries. An immediate implication of such a result would be that optimally-fair coin tossing are impossible to achieve in the random function model.[5]

[5] We mention that Dachman-Soled et al. [5] showed such an impossibility result for $O(n/\log n)$-round protocols, where n being the random function input length.

Paper Organization

Formal definitions are given in Section 2. We state our main result in Section 3 where different applications of our main result are given in Section 4. For lack of space we omit most of the proofs, and they can be found in the full version of this paper [12].

2 Preliminaries

2.1 Interactive Protocols

The communication transcript (i.e., the "transcript") of a given execution of the protocol $\pi = (\mathsf{A}, \mathsf{B})$, is the list of messages $\overline{t}$ exchanged between the parties in an execution of the protocol, where $\overline{t}_{1,\ldots,j}$ denotes the first j messages in $\overline{t}$. A view of a party contains its input, its random tape and the messages exchanged by the parties during the execution. Specifically, A's view is a tuple $v_{\mathsf{A}} = (i_{\mathsf{A}}, r_{\mathsf{A}}, \overline{t})$, where i_{A} is A's input, r_{A} are A's coins, and $\overline{t}$ is the transcript of the execution. We let $(v_{\mathsf{A}})_j$ denote the partial view of A in the first j rounds of the execution described by v_{A}, namely, $(v_{\mathsf{A}})_j = (i_{\mathsf{A}}, r_{\mathsf{A}}, \overline{t}_{1,\ldots,j})$; we define v_{B} analogously. We call $v = (v_{\mathsf{A}}, v_{\mathsf{B}})$ the *joint view* of A and B, and let $v_j = ((v_{\mathsf{A}})_j, (v_{\mathsf{B}})_j)$. Given a distribution (or a set) $\mathcal{D}$ on the joint views of A and B, we let $\mathcal{D}_{\mathsf{A}}$ be the projection of $\mathcal{D}$ on A's view (i.e., $\Pr_{\mathcal{D}_{\mathsf{A}}}[v_{\mathsf{A}}] = \Pr_{(v_{\mathsf{A}},\cdot)\leftarrow\mathcal{D}}[v_{\mathsf{A}}]$), and define $\mathcal{D}_{\mathsf{B}}$ analogously. Finally, we sometimes refer to a well structured tuple v as a "view" of π, even though v happens with zero probability. When we wish to stress that we consider a view that has non-zero probability, we call it a *valid* view.

We call π an m-round protocol, if for *every* possible random tapes for the parties, the number of rounds is *exactly* m. Given a joint view v (containing the views of both parties) of an execution of (A, B) and $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$, let v_{P} denote P's part in v and let $\mathsf{trans}(v)$ denote the communication transcript in v. For $j \in [m]$, let $\mathrm{out}_j^{\mathsf{P}}(v) = \mathrm{out}_j^{\mathsf{P}}(v_{\mathsf{P}})$ denote the output of party P at the end of the j'th round of v (i.e., the string written on P's output tape), where $\mathrm{out}_j^{\mathsf{P}}(v) = \mathrm{out}_{j-1}^{\mathsf{P}}(v)$, in case P is inactive in the j'th round of v.

We sometimes consider *stateless* protocols – the parties hold no state, and in each round act on the message received in the previous round with freshly sampled random coins. Throughout this paper we almost solely consider *no-private input* protocols – the parties' only input is the common input (the only exception to that is in Section 4.2, additional required notations introduced therein). Given a no-input two-party protocol π, let $\langle \pi \rangle$ be the distribution over the joint views of the parties in a random execution of π.

Oracle-Aided Protocols. An oracle-aided, two-party protocol $\pi = (\mathsf{A}, \mathsf{B})$ is a pair of interactive Turing machines, where each party has an additional tape called the *oracle tape*; the Turing machine can make a query to the oracle by writing a string q on its tape. It then receives a string ans (denoting the answer for this query) on the oracle tape.

For simplicity, we only consider function families whose inputs and outputs are binary strings. For an oracle-aided, no-input, two-party protocol $\pi = (\mathsf{A}, \mathsf{B})$ and a function family $\mathcal{F}$, we let $\Omega^{\mathcal{F},\pi}$ be the set of all triplets $(r_{\mathsf{A}}, r_{\mathsf{B}}, f)$, where r_{A} and r_{B} are possible random coins for A and B, and $f \in \mathcal{F}$ (henceforth, we typically omit the superscript $(\mathcal{F}, \pi)$ from the notation, whenever their values are clear from the context). For $f \in \mathcal{F}$, the distribution $\langle \pi^f = (\mathsf{A}^f, \mathsf{B}^f) \rangle$, is defined analogously to $\langle \pi \rangle = \langle \mathsf{A}, \mathsf{B} \rangle$, i.e., it is the distribution over the joint views of parties in a random execution of π with access to f. Given some information inf about some element of Ω (e.g., a set of query/answer pairs, or a view), let $\Pr_{\Omega}[\text{inf}] = \Pr_{\omega \leftarrow \Omega}[\omega \text{ is consistent with inf}]$, and let $\Pr_{\Omega|\text{inf}'}[\text{inf}]$ be this probability conditioned that ω is consistent with inf' (set to zero in case $\Pr_{\Omega}[\text{inf}'] = 0$).

Given a (possibly partial) execution of π^f, the views of the parties contain additional lists of query/answer pairs made to the oracle throughout the execution of the protocol. Specifically, A's view is a tuple $v_{\mathsf{A}} = (r_{\mathsf{A}}, \overline{t}, f_{\mathsf{A}})$, where r_{A} are A's coins, $\overline{t}$ is the transcript of the execution, and f_{A} are the oracle answers to A's queries. By convention, the active party in round j first makes all its queries to the oracle for this round, and then writes a value to its output tape and send a message to the other party. We denote by $(f_{\mathsf{P}})_j$ the oracle answers to the queries that party P makes during the first j rounds. As above, we let $(v_{\mathsf{A}})_j$ denote the partial view of A in the first j rounds of the execution described by v_{A}, namely, $(v_{\mathsf{A}})_j = (r_{\mathsf{A}}, \overline{t}_{1,\ldots,j}, (f_{\mathsf{A}})_j)$. We define v_{B} analogously.

For $\omega \in \Omega$, we let $\mathsf{view}(\omega)$ be the full view of the parties determined by ω. We say that a "view" v is *consistent* with $(\mathcal{F}, \pi)$, if $\Pr_{\Omega^{\mathcal{F},\pi}}[v] > 0$.

We consider the following distributions.

Definition 1 ($\Omega(\overline{t}, \mathcal{I})$ and $\mathcal{VIEW}(\overline{t}, \mathcal{I})$). *Given a partial transcript $\overline{t}$ and a set of query/answer pairs $\mathcal{I}$, let $\Omega(\overline{t}, \mathcal{I}) = \Omega^{\mathcal{F},\pi}(\overline{t}, \mathcal{I})$ be the set of all tuples $(r_{\mathsf{A}}, r_{\mathsf{B}}, f) \in \Omega = \Omega^{\mathcal{F},\pi}$, in which f is consistent with $\mathcal{I}$, and $\overline{t}$ is a prefix of the transcript induced by $\langle \mathsf{A}^f(r_{\mathsf{A}}), \mathsf{B}^f(r_{\mathsf{B}}) \rangle$. Given a set $\mathcal{P} \subseteq \Omega$, let $\Omega_{\mathcal{P}}(\overline{t}, \mathcal{I}) = \Omega(\overline{t}, \mathcal{I}) \cap \mathcal{P}$.*

Let $\mathcal{VIEW}(\overline{t}, \mathcal{I}) = \mathcal{VIEW}^{\mathcal{F},\pi}(\overline{t}, \mathcal{I})$ be the value of $\mathsf{view}(\omega)_{|\overline{t}|}$ for $\omega \leftarrow \Omega(\overline{t}, \mathcal{I})$, and define $\mathcal{VIEW}^{\mathcal{F},\pi}_{\mathcal{P}}(\overline{t}, \mathcal{I})$ analogously.

We note that since we consider the uniform distribution over Ω, we have that for any partial transcript $\overline{t}$, set of query/answer pairs $\mathcal{I}$, set $\mathcal{P} \subseteq \Omega$, and information inf about some element of Ω it holds that $\Pr_{\Omega_{\mathcal{P}}(\overline{t},\mathcal{I})}[\text{inf}] = \Pr_{\Omega|\overline{t},\mathcal{I},\mathcal{P}}[\text{inf}]$.

3 Mapping Oracle-Aided Protocols to No-Oracle Protocols

In this section we prove our main result, a mapping from protocols in the random-oracle model to (inefficient) no-oracle protocols.

3.1 Dependent Views

In the following we fix an m-round oracle-aided protocol π and a function family $\mathcal{F}$. We would like to restrict $\mathcal{VIEW}(\overline{t}, \mathcal{I})$ to those views for which $\mathcal{I}$ contains all the joint information of the parties about f. We start by formally defining what it means for $\mathcal{I}$ to contain all the joint information.

Definition 2. *Let v_{A} be a j_{A}-round view for A and v_{B} be a j_{B}-round view for B, for some $j_{\mathsf{A}}, j_{\mathsf{B}} \in [m]$. For $i \in [j_{\mathsf{A}}]$, let ${\mathcal{I}_{\mathsf{A}}}^i$ be the set of query/answer pairs that A makes in the i'th round of v_{A} (where ${\mathcal{I}_{\mathsf{A}}}^i = \emptyset$, if A is idle in round i), and define ${\mathcal{I}_{\mathsf{B}}}^i$ analogously. Given a set $\mathcal{I}$ of query/answer pairs, we define*

1. $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}} = \prod_{i\in[j_{\mathsf{A}}]} \Pr_{\Omega \mid \mathcal{I}, {\mathcal{I}_{\mathsf{A}}}^1, \ldots, {\mathcal{I}_{\mathsf{A}}}^{i-1}} \left[{\mathcal{I}_{\mathsf{A}}}^i\right]$ *and*
2. $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}|v_{\mathsf{B}}} = \prod_{i\in[j_{\mathsf{A}}]} \Pr_{\Omega \mid \mathcal{I}, {\mathcal{I}_{\mathsf{A}}}^1, {\mathcal{I}_{\mathsf{B}}}^1, \ldots, {\mathcal{I}_{\mathsf{A}}}^{i-1}, {\mathcal{I}_{\mathsf{B}}}^{i-1}} \left[{\mathcal{I}_{\mathsf{A}}}^i\right]$,

and define $\alpha^{\mathcal{I}}_{v_{\mathsf{B}}|v_{\mathsf{A}}}$ and $\alpha^{\mathcal{I}}_{v_{\mathsf{B}}}$ analogously.

Intuitively, $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}}$ is the probability of A's view of f given $\mathcal{I}$, and $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}|v_{\mathsf{B}}}$ is this probability when conditioning also on B's view. We will focus on those views with $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}} = \alpha^{\mathcal{I}}_{v_{\mathsf{A}}|v_{\mathsf{B}}}$ and $\alpha^{\mathcal{I}}_{v_{\mathsf{B}}} = \alpha^{\mathcal{I}}_{v_{\mathsf{B}}|v_{\mathsf{A}}}$.

Definition 3 (Dependent Views). *Let $v = (v_{\mathsf{A}}, v_{\mathsf{B}})$ be a pair of (possibly partial) valid views.*[6] *We say that v_{A} and v_{B} are* dependent *with respect to a set of query/answer pairs $\mathcal{I}$ and a function family $\mathcal{F}$, denoted* $\text{Dependent}^{\mathcal{F}}_{\mathcal{I}}(v) = 1$, *if $\alpha^{\mathcal{I}}_{v_{\mathsf{A}}} \neq \alpha^{\mathcal{I}}_{v_{\mathsf{A}}|v_{\mathsf{B}}}$ or $\alpha^{\mathcal{I}}_{v_{\mathsf{B}}} \neq \alpha^{\mathcal{I}}_{v_{\mathsf{B}}|v_{\mathsf{A}}}$.*[7]

A pair of views $v = (v_{\mathsf{A}}, v_{\mathsf{B}})$ with $\text{Dependent}_{\mathcal{I}}(v) = 0$ *is called* independent. *We let* $\text{Ind}^{\mathcal{F},\pi}(\overline{t}, \mathcal{I}) = \{\omega \in \Omega(\overline{t}, \mathcal{I})\colon \text{Dependent}^{\mathcal{F}}_{\mathcal{I}}(\mathsf{view}(\omega)_{|\overline{t}|}) = 0\}$ *and let $\mathcal{VIEW}^{\mathcal{F},\pi}_{\text{Ind}}(y)$ stand for $\mathcal{VIEW}^{\mathcal{F},\pi}_{\text{Ind}^{\mathcal{F},\pi}(y)}(y)$.*

3.2 Intersecting Views

A special case of dependent views is when the two paries share a *common* oracle query not in $\mathcal{I}$.

Definition 4 (Intersecting Views). *A (possibly partial) pair of views $v = (v_{\mathsf{A}}, v_{\mathsf{B}})$ are* intersecting *with respect to a set of query/answer pairs $\mathcal{I}$, denoted* $\text{Intersect}_{\mathcal{I}}(v) = 1$, *if v_{A} and v_{B} share a common query q not in $\mathcal{I}$ (i.e., $(q, \cdot) \notin \mathcal{I}$).*

For most function families, an intersection implies being dependent (with respect to the same list of query/answer pairs). In this paper we limit our attention to "simple" function families for which also the other direction holds, namely dependency implies intersection.

[6] While properly defined for any pair of views $(v_{\mathsf{A}}, v_{\mathsf{B}})$, we will typically only consider the following notions for pairs with $\mathsf{trans}(v_{\mathsf{A}}) = \mathsf{trans}(v_{\mathsf{B}})$ (i.e., both views induce the same transcript).

[7] One can verify that $\text{Dependent}^{\mathcal{F}}_{\mathcal{I}}(v) = 1$, in case v is inconsistent with $\mathcal{F}$, namely, $\Pr_{\mathcal{F}}[\mathcal{I}_{\mathsf{A}}, \mathcal{I}_{\mathsf{B}}] = 0$, where $\mathcal{I}_{\mathsf{A}}$ and $\mathcal{I}_{\mathsf{B}}$, are the lists of query/answer pairs appear in v_{A} and v_{B} respectively.

Definition 5 (Simple Function Families). *A function family $\mathcal{F}$ is* simple*, if for any oracle-aided protocol π, list $\mathcal{I}$ of query/answer pairs that is consistent with some $f \in \mathcal{F}$, and a (possibly partial) pair of views $v = (v_{\mathsf{A}}, v_{\mathsf{B}})$ consistent with $\mathcal{I}$, it holds that* $\text{Dependent}^{\mathcal{F}}_{\mathcal{I}}(v) = 1$ *iff* $\text{Intersect}_{\mathcal{I}}(v) = 1$.

It is not hard to verify that the all-function family is simple (see proof in [12]).

Definition 6 (the all-function family). *For $n \in \mathbb{N}$, let $\mathcal{F}_{\text{AF}n}$ be the family of all functions from n-bit strings to n-bit strings.*

Lemma 1. *For every $n \in \mathbb{N}$, the family $\mathcal{F}_{\text{AF}n}$ is simple.*

3.3 Oracle-Aided to No-Oracle Protocol Mapping

The following theorem shows that an execution of an oracle-aided protocol with oracle access to a random $f \in \mathcal{F}$, where $\mathcal{F}$ is a simple function family, can be mapped to an execution of a related protocol with no oracle access. In Section 4 we use this result to prove limitations on the power of oracle-aided protocols in achieving specific cryptographic tasks.

Definition 7 (Oracle-Aided to No-Oracle Mapping). *A pair of a function family $\mathcal{F}$ and a no-input, m-round, oracle-aided protocol $\pi = (\mathsf{A}, \mathsf{B})$, has a (T, ε)-*mapping*, if there exists a deterministic, oracle-aided T-query algorithm* Map *and a stateless, m-round, no-input (and no-oracle) protocol $(\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$, such that the following hold:*

1. $\text{SD}(\mathcal{D}_{\mathcal{F}}, \mathcal{D}_P) \leq \varepsilon$ *for every $j \in [m]$, where*

$$\mathcal{D}_{\mathcal{F}} = \left(\text{out}^{\mathsf{A}}_j(v), \text{out}^{\mathsf{B}}_j(v), \mathsf{Map}^f(\mathsf{trans}(v)_{1,\ldots,j})\right)_{f \leftarrow \mathcal{F}, v \leftarrow \langle \mathsf{A}^f, \mathsf{B}^f \rangle} \quad \textit{and,}$$

$$\mathcal{D}_P = \left(\text{out}^{\widetilde{\mathsf{A}}}_j(v), \text{out}^{\widetilde{\mathsf{B}}}_j(v), \mathsf{trans}(v)_{1,\ldots,j}\right)_{v \leftarrow \langle \widetilde{\mathsf{A}}, \widetilde{\mathsf{B}} \rangle}.$$ [8]

   Furthermore, $\mathcal{D}_P[1,3] \equiv \mathcal{D}_{\mathcal{F}}[1,3]$ and $\mathcal{D}_P[2,3] \equiv \mathcal{D}_{\mathcal{F}}[2,3]$.[9]
2. *For every $f \in \mathcal{F}$, an m-round transcript $\overline{t}$ and $j \in [m]$, it holds that $\mathsf{Map}^f(\overline{t}_{1,\ldots,j}) = \mathsf{Map}^f(\overline{t})_{1,\ldots,j}$. Furthermore, the oracle calls made in $\mathsf{Map}^f(\overline{t}_{1,\ldots,j})$ are a subset of those made in $\mathsf{Map}^f(\overline{t})$.*

Theorem 4. *Let $\mathcal{F}$ be a simple function family and let $\pi = (\mathsf{A}, \mathsf{B})$ be an ℓ-query, oracle-aided, no-input protocol, then $(\mathcal{F}, \pi)$ has an $(256 \cdot \ell^2/\varepsilon^2, \varepsilon)$-*mapping *for any $0 < \varepsilon \leq 1$.*

Remark 1 (Round complexity of the no-oracle protocol). The proof of Theorem 4 can be easily modified to yield a one-message no-oracle protocol (in this case, $\mathcal{D}_{\mathcal{F}}$ and $\mathcal{D}_P$ should be modified to reflect the transcript and outputs at the end of the executions). The roles of $\widetilde{\mathsf{A}}$ and $\widetilde{\mathsf{B}}$ in the resulting protocol however, cannot reflect as closely the roles of A and B, as done in the many-round, no-oracle protocol stated above.

[9] I.e., the projections of $\mathcal{D}_P$ and $\mathcal{D}_{\mathcal{F}}$ to their transcript part and the output of one of the parties, are identically distributed.

The heart of the proof is the following lemma, proof given in [12] .[10]

Definition 8 (DependencyFinder). *Let $\mathcal{F}$ be a function family and let $\pi = (\mathsf{A}, \mathsf{B})$ be an m-round oracle-aided protocol. A deterministic oracle-aided algorithm* Finder *is a (T, ε)-*DependencyFinder *for $(\mathcal{F}, \pi)$ if the following holds for any $j \in [m]$: consider the following random process $\mathsf{CF} = \mathsf{CF}(\mathcal{F}, \pi, \mathsf{Finder})$:*

1. *Choose $(r_{\mathsf{A}}, r_{\mathsf{B}}, f) \leftarrow \Omega^{\mathcal{F},\pi}$ and let $\overline{t}$ be the j-round transcript of π induced by $(r_{\mathsf{A}}, r_{\mathsf{B}}, f)$.*
2. *For $i = 1$ to j set $\mathcal{I}_i = \mathcal{I}_{i-1} \cup \mathsf{Finder}^f(\overline{t}_{1,\ldots,i}, \mathcal{I}_{i-1})$ (letting $\mathcal{I}_0 = \emptyset$), where $\mathsf{Finder}^f(x)$ is the set of queries/answers made by $\mathsf{Finder}^f(x)$ to f.*
3. *Output $\left(\overline{t}, \mathcal{I}_j\right)$.*

Then

1. $\mathrm{E}_{d \leftarrow \mathsf{CF}}\left[\mathrm{SD}\left(\mathcal{VIEW}^{\mathcal{F},\pi}(d), (\mathcal{VIEW}^{\mathcal{F},\pi}(d)_{\mathsf{A}}, \mathcal{VIEW}^{\mathcal{F},\pi}(d)_{\mathsf{B}})\right)\right] \leq \varepsilon$, *and*
2. $\Pr[\#$ *of f-calls made in* $\mathsf{CF} > T] \leq \varepsilon$.

That is, conditioned on a random transcript of $\pi^{\mathcal{F}}$ and the oracle queries made by a (T, δ)-DependencyFinder, the parties' views are close to being in a product distribution.

Lemma 2. *Let $\mathcal{F}$ be a simple function family and let $\pi = (\mathsf{A}, \mathsf{B})$ be an ℓ-query oracle-aided protocol, then $(\mathcal{F}, \pi)$ has a $(64/\delta^2, \ell\delta)$-*DependencyFinder *for any $0 < \delta \leq 1/\ell$.*

We now use Lemma 2 to prove Theorem 4.

Proving Theorem 4. Fix a simple function family $\mathcal{F}$ and a no-input, m-round, ℓ-query oracle-aided protocol π. Fix $0 < \varepsilon \leq 1$ and let Finder be the $(T = 256 \cdot \ell^2/\varepsilon^2, \varepsilon/2)$-DependencyFinder guaranteed by Lemma 2 for $(\mathcal{F}, \pi)$ (taking $\delta = \varepsilon/2\ell$). We start by defining the mapping algorithm and then we define a protocol with no oracle access.

Algorithm 5 (Map)
Oracle: $f \in \mathcal{F}$.
Input: j-round transcript $\overline{t}$ of π.
Operation:

1. *For $i = 1$ to j set $\mathcal{I}_i = \mathcal{I}_{i-1} \cup \mathsf{Finder}^f(\overline{t}_{1,\ldots,i}, \mathcal{I}_{i-1})$ (letting $\mathcal{I}_0 = \emptyset$).*
2. *If in some round i^* the* overall *number of f calls (made by* Finder*) is T, halt the above procedure and set $\mathcal{I}_{i^*}$ to be the set of T query/answer pairs obtained so far,[11] and set $\mathcal{I}_i = \mathcal{I}_{i^*}$ for all $i^* < i \leq j$.*
3. *Output $\left(\overline{t}_1, \mathcal{I}_1\right), \left(\overline{t}_{1,2}, \mathcal{I}_2\right), \ldots, \left(\overline{t}, \mathcal{I}_j\right)$.*

[10] As mentioned in the introduction, the proof of Lemma 2 could be essentially derived by combining several statements in [1]. Alternatively, a somewhat weaker variant of the lemma can be directly proved using the followup result of Dachman-Soled et al. [5, Lemma 5] or of Mahmoody et al. [16, Lemma A.1].

[11] I.e., augmenting $\mathcal{I}_{i^*-1}$ with the queries/answers made in round i^* before halting.

The no-oracle protocol. Our stateless, no-oracle protocol $\widetilde{\pi} = (\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$, emulates the oracle-aided protocol π by keeping the "important" oracle queries as part of the transcript, and selecting the rest of the oracle at random (independently in each round). In particular, $\widetilde{\mathsf{A}}$ is active in $\widetilde{\pi}$ in the same rounds that A is in π (same for $\widetilde{\mathsf{B}}$ and B). The definition of $\widetilde{\mathsf{A}}$ is given below ($\widetilde{\mathsf{B}}$ is analogously defined).

Algorithm 6 ($\widetilde{\mathsf{A}}$)
Input: A pair $(\overline{t}, \mathcal{I})$*, where* $\overline{t}$ *is a transcript of length* j *and* $\mathcal{I}$ *is a set of query/answer pairs.*
Operation:

1. *Sample* $(r_{\mathsf{A}}, r_{\mathsf{B}}, f) \leftarrow \Omega(\overline{t}, \mathcal{I})$*, and let* out_{j+1} *and* t_{j+1} *denote* A*'s output and message respectively, in the* $(j+1)$ *round of* $\langle \mathsf{A}^f(r_{\mathsf{A}}), \mathsf{B}^f(r_{\mathsf{B}}) \rangle$*.*
2. *Output* out_{j+1}*.*
3. *Compute the value of* $\mathcal{I}_{j+1}$ *output by* $\mathsf{Map}^f(\overline{t_{j+1}})$ *for* $\overline{t_{j+1}} = (\overline{t}, t_{j+1})$*.*
4. *Send* $(\overline{t_{j+1}}, \mathcal{I}_{j+1})$ *to* $\widetilde{\mathsf{B}}$*.*

Using Lemma 2, one can prove that above mapping function and no-oracle protocol, indeed establish mapping and protocol guaranteed in Theorem 4. For the formal proof, see the full version.

4 Applications

In this section we use our main result (i.e., the oracle-aided to no-oracle protocol mapping for simple function families) from Section 3 to derive the impossibility of realizing three cryptographic tasks, with respect to simple function families (implying the same result with respect to the all function family, which is simple). In Section 4.1 we re-establish the result of [13], showing that key-agreement protocols cannot be realized with respect to simple function families. Then, in Section 4.2, we extend the lower-bound of [17] on the accuracy of two-party differentially private no-oracle protocols, to show it also holds (with a slight loss in parameters) for oracle-aided differentially private protocols (with respect to this class of function families). Finally, in Section 4.3, we show that no-input functionalities that cannot be securely evaluated in the no-oracle model (even when allowing some small loss of security), cannot be securely evaluated (again, even with some small loss of security) by oracle-aided protocols that are given access to a random member of a simple function family.

Remark 2 (definitions for no-oracle primitives). Throughout this section we only give formal definitions (of the security and correctness) of primitives with respect to oracle-aided protocols. Deriving formal definitions for their no-oracle counterparts can be easily done by considering the trivial function family (i.e., a singleton family, whose only member returns $\perp$ on any query).

4.1 Key Agreement Protocols

In a key-agreement protocol two parties wish to agree on a common secret in a secure way — an observer (adversary) seeing the communication transcript, cannot find the secret. Below we prove that with respect to a certain class of function families, non-trivial key-agreement cannot be achieved. We start by formally defining the notion of key agreement. We then recall the known fact that in the no-oracle model, an adversary can reveal any secret agreement between two parties in the strongest possible sense (i.e., with the same probability that the parties themselves agree). Combining this fact with the mapping from oracle-aided to no-oracle protocols, described in Section 3, yields a similar result for oracle-aided protocols.

We remark that the results presented in this section yield very little conceptual added-value to what was already shown by [13, 1]. We do, however, present them here to demonstrate how they are easily derived from our main result (Theorem 4), and as a warm-up before moving on to the other applications of our main result, described in Sections 4.2 and 4.3.

Standard Definitions and Known Facts. Recall (see Section 2.1) that for a joint view $v \in \mathrm{Supp}\left(\langle \pi^f \rangle\right)$, we let $\mathsf{trans}(v)$ denote the communication transcript in v, and $\mathrm{out}_i^{\mathsf{P}}(v)$ denote the output of the party P at the i'th round. In the following we let $\mathrm{out}^{\mathsf{P}}(v) = \mathrm{out}_m^{\mathsf{P}}(v)$, where m is the last round in v.

Definition 9 (Key Agreement Protocol). *Let $0 \leq \gamma,\ \alpha \leq 1$ and $k \in \mathbb{N}$. A two-party, oracle-aided protocol $\pi = (\mathsf{A}, \mathsf{B})$ is a (k, α, γ)-key-agreement protocol with respect to a function family $\mathcal{F}$, if the following hold:*

Consistency: *π is* $(1-\alpha)$-consistent *with respect to $\mathcal{F}$. Namely for every $f \in \mathcal{F}$,*

$$\Pr_{v \leftarrow \langle \pi^f \rangle}\left[\mathrm{out}^{\mathsf{A}}(v) = \mathrm{out}^{\mathsf{B}}(v)\right] \geq 1 - \alpha. \tag{1}$$

Security: *For every $P \in \{\mathsf{A}, \mathsf{B}\}$ and any k-query adversary Eve,*

$$\Pr_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f \rangle}\left[\mathsf{Eve}^f\left(\mathsf{trans}(v)\right) = \mathrm{out}^P(v)\right] \leq \gamma. \tag{2}$$

A protocol π is an (α, γ)-key-agreement protocol, if it is a $(\cdot, \alpha, \gamma)$-key-agreement protocol with respect to the trivial function family.[12]

In the no-oracle model, all correlation between the parties is implied by the transcript. Hence, an adversary that on a given transcript $\overline{t}$ samples a random view for A that is consistent with $\overline{t}$ and outputs whatever A would upon this view, agrees with B with the same probability as does A. This simple argument yields the following fact.

[12] We remark that our impossibility result (as well the results of [13, 1]) would also hold with respesct to a weaker definition, requiring consistency to hold for a random f, rather than for every $f \in \mathcal{F}$.

Fact 7. *Let $0 \leq \alpha \leq 1$ and let $\pi = (\mathsf{A}, \mathsf{B})$ be a no-oracle, two-party, no-input protocol. Assume that the probability that in a random execution of π both parties output the same value is $1 - \alpha$. Then there exists an adversary that, given the transcript of a random execution of π, outputs the same value as does B with probability $1 - \alpha$.*

An immediate implication of Fact 7 is that there does not exist any no-oracle, two-party, (α, γ)-key-agreement protocol for any $0 \leq \gamma < 1 - \alpha$. We next use our main result from Section 3 to prove a similar result for oracle-aided protocols.

Our Result. In the language of the above definition, our main result is stated as follows.

Theorem 8. *Let $\mathcal{F}$ be a function family and let π be an oracle-aided protocol. Assume that the pair $(\mathcal{F}, \pi)$ has a* (T, ε)-mapping, *then π is* not *a (T, α, γ)-key-agreement with respect to $\mathcal{F}$ for any $0 \leq \gamma < 1 - (\alpha + \varepsilon)$.*

Proof. Assume to the contrary that π is a (T, α, γ)-key-agreement with respect to $\mathcal{F}$ for some $0 \leq \gamma < 1 - (\alpha + \varepsilon)$. Let $\widetilde{\pi} = (\widetilde{\mathsf{A}}, \widetilde{\mathsf{B}})$ and Map be the no-input no-oracle protocol and oracle-aided algorithm, guaranteed by the assumption of the theorem. The first item in Definition 7 yields that

$$\mathrm{SD}\left(\left(\mathrm{out}^{\widetilde{\mathsf{A}}}(v), \mathrm{out}^{\widetilde{\mathsf{B}}}(v)\right)_{v \leftarrow \langle \widetilde{\pi} \rangle}, \left(\mathrm{out}^{\mathsf{A}}(v), \mathrm{out}^{\mathsf{B}}(v)\right)_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f \rangle}\right) \leq \varepsilon \tag{3}$$

Hence, the $(1 - \alpha)$-consistency of π yields that

$$\tau := \Pr_{v \leftarrow \langle \widetilde{\pi} \rangle}\left[\mathrm{out}^{\widetilde{\mathsf{A}}}(v) = \mathrm{out}^{\widetilde{\mathsf{B}}}(v)\right] \geq 1 - (\alpha + \varepsilon). \tag{4}$$

Fact 7 yields an adversary $\widetilde{\mathsf{Eve}}$ that given the transcript of a random execution of $\widetilde{\pi}$, outputs the same value as does B with probability τ. Let Eve be an adversary for π that upon a transcript $\overline{t}$ (of an execution of π with access to f) applies $\widetilde{\mathsf{Eve}}$ to $\mathsf{Map}^f(\overline{t})$ and outputs whatever $\widetilde{\mathsf{Eve}}$ does. Note that by Definition 7, Eve makes at most T oracle calls. The definition of Eve yields that

$$\begin{aligned}
&\Pr_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f \rangle}\left[\mathsf{Eve}^f(\mathsf{trans}(v)) = \mathrm{out}^{\mathsf{B}}(v)\right] \\
&= \Pr_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f \rangle}\left[\widetilde{\mathsf{Eve}}\left(\mathsf{Map}^f(\mathsf{trans}(v))\right) = \mathrm{out}^{\mathsf{B}}(v)\right] \\
&= \Pr_{\tilde{v} \leftarrow \langle \widetilde{\pi} \rangle}\left[\widetilde{\mathsf{Eve}}(\mathsf{trans}(\tilde{v})) = \mathrm{out}^{\widetilde{\mathsf{B}}}(\tilde{v})\right] = \tau \geq 1 - (\alpha + \varepsilon),
\end{aligned} \tag{5}$$

where the second equality follows from the furthermore statement of the first item in Definition 7, stating that $(\mathsf{Map}^f(\mathsf{trans}(v)), \mathrm{out}^{\mathsf{B}}(v))$ is identically distributed as $(\mathsf{trans}(\tilde{v}), \mathrm{out}^{\widetilde{\mathsf{B}}}(\tilde{v}))$, where f, v, and $\tilde{v}$ are sampled as in Equation (5). □

Combining Theorems 4 and 8 yields the following result.

Theorem 9. *Let $\mathcal{F}$ be a simple function family. For parameters $k, \ell \in \mathbb{N}$ and $\alpha, \gamma \in \mathbb{R}$ with $k \geq 2^{10} \cdot \left(\frac{\ell}{1-\alpha-\gamma}\right)^2$ and $1 - \alpha > \gamma \geq 0$, there exists no ℓ-query oracle-aided protocol, that is (k, α, γ)-key-agreement with respect to $\mathcal{F}$.*

4.2 Differentially Private Two-Party Computation

In this section we apply our main result to extend the lower-bound of McGregor et al. [17] to oracle-aided protocols equipped with simple function families. Specifically, we show that when given access to a random member of a simple function family (e.g., the all-function family), any two-party, differentially private, oracle-aided protocol computing the inner product of two s-bit strings, exhibits error magnitude of roughly $\Omega\left(\sqrt{s}/\log s\right)$.

Standard Definitions. For strings $x, x' \in \Sigma^s$, let $H_d(x, x') = |\{i \in [s] \colon x_i \neq x'_i\}|$ denote the Hamming distance between x and x'.

Definition 10 (Differential Privacy for Oracle-Aided Protocols). *Let $\mathcal{F}$ be a function family and let $\pi = (\mathsf{A}, \mathsf{B})$ be an s-bit input, oracle-aided protocol. The protocol π is (k, α, γ)-*differentially private with respect to $\mathcal{F}$ and A*, if for every k-query, oracle-aided distinguisher D and every $x, x', y \in \{0,1\}^s$ with $H_d(x, x') = 1$, it holds that*

$$\Pr_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f(x,y) \rangle} \left[\mathsf{D}^f(\mathsf{trans}(v)) = 1\right] \leq e^{\alpha} \cdot \Pr_{f \leftarrow \mathcal{F}, v \leftarrow \langle \pi^f(x',y) \rangle} \left[\mathsf{D}^f(\mathsf{trans}(v)) = 1\right] + \gamma.$$

*Being (k, α, γ)-differentially private with respect to $\mathcal{F}$ and B, is analogously defined. If π is (k, α, γ)-differentially private with respect to $\mathcal{F}$ and both parties, then it is (k, α, γ)-*differentially private with respect to $\mathcal{F}$*.*
Finally, π is (α, γ)-differentially private, if it is $(\cdot, \alpha, \gamma)$-differentially private with respect to the trivial function family.

Note that for no-oracle protocols, the above definition of (α, γ)-differentially private matches the standard (no-oracle) definition (slightly relaxed, as we only require the transcript to preserve the privacy of the parties). Our impossibility results, given below, apply to privacy parameter α being smaller than some constant.

Since differentially private mechanisms cannot be deterministic, for any deterministic (non-constant) function g of the input, one can only hope for the output of the mechanism being a good approximation for g. We next define a notion of accuracy for differentially private protocols.

Definition 11 (Good Approximations). *Let $g : \{0,1\}^s \times \{0,1\}^s \mapsto \mathbb{R}$ be a deterministic function and let $\pi = (\mathsf{A}, \mathsf{B})$ be an s-bit input, oracle-aided protocol. The protocol π is a (β, d)-approximation for g with respect to a function family $\mathcal{F}$, if for very $f \in \mathcal{F}$, for every $x, y \in \{0,1\}^s$ and $\mathsf{P} \in \{\mathsf{A}, \mathsf{B}\}$, it holds that*

$$\Pr_{v \leftarrow \langle \pi^f(x,y) \rangle} \left[\left|g(x,y) - \mathrm{out}^{\mathsf{P}}(v)\right| > d\right] < \beta. \tag{6}$$

Namely, we require that the output of both parties is within distance d from $g(x,y)$ with probability at least β.

For two s-bit strings x and y, let $\mathsf{IP}(x,y)$ denote the inner product of x and y; that is $\mathsf{IP}(x,y) = \sum_{i\in[s]} x_i \cdot y_i$.

Our Result. Combining Theorem 4 and the lower bounds of McGregor et al. [17] we get the following result (see proof in [12]) .

Definition 12 (The Sampled-Input Variant $\mu(\pi)$). *Given an s-bit input, (possibly, oracle-aided) protocol $\pi = (\mathsf{A},\mathsf{B})$, let $\mu(\pi) = (\mu(\mathsf{A}), \mu(\mathsf{B}))$ denote the following s-bit sampled-input protocol:*
The parties $\mu(\mathsf{A})$ and $\mu(\mathsf{B})$ interact in an execution of $(\mathsf{A}(x_\mathsf{A}; r_\mathsf{A}), \mathsf{B}(x_\mathsf{B}; r_\mathsf{B}))$, taking the roles of A and B respectively, where x_A [resp., x_B] is the first s bits of $\mu(\mathsf{A})$'s [resp., $\mu(\mathsf{B})$'s] coins, and r_A [resp., r_B] is the rest of $\mu(\mathsf{A})$'s [resp., $\mu(\mathsf{B})$'s] coins. Let a and b be the outputs of A and B, respectively, in this execution, then the outputs of $\mu(\mathsf{A})$ and $\mu(\mathsf{B})$ will be (x_A, a) and (x_B, b), respectively.

Theorem 10. *For numbers $\nu > 0$ and $\alpha \geq 0$, there exist numbers $\lambda > 0$ and $z \in \mathbb{N}$ such that the following holds. Let $\mathcal{F}$ be a function family and let $\pi = (\mathsf{A},\mathsf{B})$ be an oracle-aided, s-bit input protocol.*

*Assume that π is (T,α,γ)-differentially private with respect to $\mathcal{F}$, that the pair $(\mathcal{F}, \mu(\pi))$ has a (T,ε)-*mapping *(where $\mu(\pi)$ is sampled-input variant of π) and that $s \geq z$, then for some $f \in \mathcal{F}$,[13] and every $\mathsf{P} \in \{\mathsf{A},\mathsf{B}\}$, there exist $x, y \in \{0,1\}^s$ such that*

$$\Pr_{v \leftarrow \langle \pi^f(x,y)\rangle}\left[\left|\mathrm{out}^\mathsf{P}(v) - \mathsf{IP}(x,y)\right| \leq \Delta := \lambda \cdot \frac{\sqrt{s}}{\log s} \cdot (\tau - \varepsilon)\right] \leq \tau \qquad (7)$$

for every $\tau \leq 1$ with $\tau - \varepsilon \geq \max\{48s\gamma, \nu\}$.[14]

Combining Theorems 4 and 10 yields the following result.

Theorem 11. *Let $\mathcal{F}$ be a simple function family. For numbers $0 < \nu < 1$ and $\alpha \geq 0$, there exist numbers $\lambda > 0$ and $z \in \mathbb{N}$ such that, for $s \geq z$, the following holds. Assume that π is an s-bit input, ℓ-query oracle-aided protocol that is (k,α,γ)-differentially private with respect to $\mathcal{F}$, with $k > 2^{10} \cdot \left(\frac{\ell}{1-\nu}\right)^2$ and $\gamma \leq \frac{\nu}{48\cdot s}$. Then, π is* not *a (β,d)-approximation with respect to $\mathcal{F}$ for the inner-product function, with $\beta < \frac{1-\nu}{2}$ and $d \leq \lambda \cdot \nu \cdot \frac{\sqrt{s}}{\log s}$.*

4.3 Secure Function Evaluation

In this section we apply our main result to show that when given access to a random member of a simple function family (e.g., the all-function family),

[13] Actually, the following holds for *most* elements of $\mathcal{F}$.

[14] This constraint implies that γ should be smaller than the inverse of some polynomial in s, however, this is how we typically think of γ.

no oracle-aided protocol can securely compute any no-input functionality that cannot be (almost) securely computed by a no-oracle protocol.

In semi-honest no-input secure function evaluation, two parties A and B wish to compute some (randomized) functionality privately and correctly. Let $G = (G_{\mathsf{A}}, G_{\mathsf{B}})$ be a distribution over $\mathcal{A} \times \mathcal{B}$, where G_{A} and G_{B} denote its marginal distributions over $\mathcal{A}$ and $\mathcal{B}$ respectively. The parties wish to perform a computation, where party A learns $g_{\mathcal{A}}$ and party B learns $g_{\mathcal{A}}$ for $g = (g_{\mathcal{A}}, g_{\mathcal{B}}) \leftarrow G$, but nothing else. Since the parties are semi-honest, they will always follow the prescribed protocol. A corrupted party, however, may try to use its view in the computation to infer additional information after the computation terminates.

Standard Definitions

Definition 13 (No-Input Secure Function Evaluation). *Let $G = (G_{\mathsf{A}}, G_{\mathsf{B}})$ be a distribution over $\mathcal{A} \times \mathcal{B}$, where G_{A} and G_{B} denote its marginal distributions $\mathcal{A}$ and $\mathcal{B}$ respectively. A two-party, oracle-aided protocol $\pi = (\mathsf{A}, \mathsf{B})$ is a (m, k, δ)-secure protocol for G with respect to a function family $\mathcal{F}$, for $\delta \in [0, 1]$ and $m, k \in \mathbb{N}$, if the following hold:*

Correctness: *π is a δ-correct implementation of G with respect to $\mathcal{F}$:*

$$\mathrm{SD}\left(\left(\mathrm{out}^{\mathsf{A}}(v), \mathrm{out}^{\mathsf{B}}(v)\right)_{v \leftarrow \langle \pi^f \rangle}, G\right) \leq \delta$$

for every $f \in \mathcal{F}$.

Privacy: *π is an (m, k, δ)-private implementation of G with respect to $\mathcal{F}$: for every $P \in \{\mathsf{A}, \mathsf{B}\}$ there exists an m-query algorithm (simulator) $\mathrm{Sim}_{\mathsf{P}}$ such that*

$$\mathop{\mathrm{E}}_{f \leftarrow \mathcal{F}} \left| \left[\Pr\left[\mathsf{D}\left(\left(\mathrm{Sim}_{\mathsf{P}}^f(g), g \right)_{g \leftarrow G_{\mathsf{P}}} \right) = 1 \right] - \Pr\left[\mathsf{D}\left(\left(v_{\mathsf{P}}, \mathrm{out}^{\mathsf{P}}(v) \right)_{v \leftarrow \langle \pi^f \rangle} \right) = 1 \right] \right] \right| \leq \delta$$

for any k-query distinguisher D.

A protocol π is a δ-secure (no-oracle) implementation of G if it is a $(\cdot, \cdot, \delta)$-secure implementation of G with respect to the trivial (i.e., the empty) function family. A distribution G is δ-trivial, if G has a δ-secure no-oracle implementation.

Our Result. In the language of the above definitions, our main result is stated as follows (see proof in [12]) .

Theorem 12. *Let $\mathcal{F}$ be a function family, and let π be an oracle-aided protocol that is a $(\cdot, T, \delta)$-secure oracle-aided implementation of a distribution G with respect to $\mathcal{F}$. Assume that the pair $(\mathcal{F}, \pi)$ has a (T, δ)-*mapping*. Then, G is 2δ-trivial.*

Combining Theorems 4 and 12 yields the following result.

Theorem 13. *Let $\mathcal{F}$ be a simple function family. For parameters $k, \ell \in \mathbb{N}$ and $\delta \in \mathbb{R}$ with $k \geq 256 \cdot \left(\frac{\ell}{\delta}\right)^2$, and for a distribution G that is not 2δ-trivial, there exists no ℓ-query oracle-aided protocol that is a $(\cdot, k, \delta)$-secure oracle-aided implementation of G with respect to $\mathcal{F}$.*

References

1. Barak, B., Mahmoody-Ghidary, M.: Merkle Puzzles Are Optimal — An $O(n^2)$-Query Attack on Any Key Exchange from a Random Oracle. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 374–390. Springer, Heidelberg (2009)
2. Beimel, A., Nissim, K., Omri, E.: Distributed private data analysis: On simultaneously solving how and what. CoRR, abs/1103.2626 (2011)
3. Canetti, R., Goldreich, O., Halevi, S.: On the Random-Oracle Methodology as Applied to Length-Restricted Signature Schemes. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 40–57. Springer, Heidelberg (2004)
4. Chang, Y.-C., Hsiao, C.-Y., Lu, C.-J.: On the Impossibilities of Basing One-Way Permutations on Central Cryptographic Primitives. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 110–124. Springer, Heidelberg (2002)
5. Dachman-Soled, D., Lindell, Y., Mahmoody, M., Malkin, T.: On the Black-Box Complexity of Optimally-Fair Coin Tossing. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 450–467. Springer, Heidelberg (2011)
6. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
7. Gennaro, R., Trevisan, L.: Lower bounds on the efficiency of generic cryptographic constructions. In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, pp. 305–313 (2000)
8. Gennaro, R., Gertner, Y., Katz, J., Trevisan, L.: Bounds on the efficiency of generic cryptographic constructions. SIAM Journal on Computing 35(1), 217–246 (2005)
9. Gertner, Y., Kannan, S., Malkin, T., Reingold, O., Viswanathan, M.: The relationship between public key encryption and oblivious transfer. In: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, STOC (2000)
10. Goldwasser, S., Tauman-Kalai, Y.: On the (in)security of the fiat-shamir paradigm. In: Proceedings of the 44th Annual Symposium on Foundations of Computer Science, FOCS (2003)
11. Haitner, I., Hoch, J.J., Reingold, O., Segev, G.: Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In: Proceedings of the 48th Annual Symposium on Foundations of Computer Science, FOCS (2007)
12. Haitner, I., Omri, E., Zarosim, H.: Limits on the usefulness of random oracles. Technical Report 2012/573, Cryptology ePrint Archive (2012), `http://eprint.iacr.org/2012/573`
13. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC), pp. 44–61. ACM Press (1989)
14. Kahn, J., Saks, M., Smyth, C.: A dual version of reimer's inequality and a proof of rudich's conjecture. In: Proceedings of the 15th Annual IEEE Conference on Computational Complexity, 2000, pp. 98–103 (2000)
15. Kim, J.H., Simon, D., Tetali, P.: Limits on the efficiency of one-way permutation-based hash functions. In: 40th Annual Symposium on Foundations of Computer Science, 1999, pp. 535–542 (1999)
16. Mahmoody, M., Maji, H.K., Prabhakaran, M.: Limits of random oracles in secure computation. Technical Report, arXiv:1205.3554v1 (2012)

17. McGregor, A., Mironov, I., Pitassi, T., Reingold, O., Talwar, K., Vadhan, S.P.: The limits of two-party differential privacy. In: Electronic Colloquium on Computational Complexity (ECCC), p. 106 (2011); Preliminary version in FOCS 2010 (2010)
18. Merkle, R.C.: Secure communications over insecure channels. In: SIMMONS: Secure Communications and Asymmetric Cryptosystems (1982)
19. Mironov, I., Pandey, O., Reingold, O., Vadhan, S.: Computational Differential Privacy. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 126–142. Springer, Heidelberg (2009)
20. Pointcheval, D., Stern, J.: Security Proofs for Signature Schemes. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 387–398. Springer, Heidelberg (1996)
21. Rudich, S.: The Use of Interaction in Public Cryptosystems. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 242–251. Springer, Heidelberg (1992)
22. Simon, D.R.: Findings Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998)
23. Wee, H.: One-Way Permutations, Interactive Hashing and Statistically Hiding Commitments. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 419–433. Springer, Heidelberg (2007)