# A Note on Black-Box Separations and Key Agreement from OWFs

Naman Kumar*
Oregon State University

May 14, 2024

**Abstract**

We provide a simple exposition of the black-box separation of [IR89], rephrasing the terminology in modern notions. We also attempt to simplify the proof wherever possible, though we hope to preserve the general spirit of the proof.

## 1 Introduction

The seminal proof of Impagliazzo and Rudich provides a black-box separation between one-way functions and key agreement protocols. It is the first in a long line of works that aims to characterize the construction of cryptographic primitives based on foundational principles. Several other works since then ([HR04],[RTV04],[Sim98]) have formalized these ideas and gained a variety of new results. In this note we focus only on the original paper [IR89].

We state here our main result:

**Theorem 1.1** (Main Theorem, Informal)**.** *There exists an oracle relative to which a strongly one-way permutation exists, but secure secret-key agreement is impossible.*

The oracle with respect to which the result holds is any oracle for which $\mathbf{P} = \mathbf{NP}$; in particular, a **PSPACE** oracle suffices.

### 1.1 Black-Box Reducibility and Separations

While the above theorem makes it clear that any construction of secret-key agreement using a one-way permutation does not relativize, it is not immediate how this ties into the notion of black-box separation. Such separations were formally analyzed by [RTV04] and [BBF13], who grouped black-box reductions into three distinct categories. We restate those notions here. Assume here that $\mathcal{P}$ and $\mathcal{Q}$ are primitives, and $f$ is any functionality that realizes $\mathcal{Q}$.

It may be instructive here to keep an example in mind: consider, for instance, a black-box construction of CPA-secure encryption from a PRF.

1. **Fully Black-Box Reductions**: Let $G^f$ be an implementation of $\mathcal{P}$, where $G$ can query $f$ as an oracle. A fully black-box reduction is an efficient algorithm that transforms any (even

*kumarnam@oregonstate.edu.

inefficient!) adversary $\mathcal{A}$ that breaks $G^f$ as an implementation of $\mathcal{P}$ into an algorithm $\mathcal{S}^{\mathcal{A},f}$ which breaks the instance $f$ of $\mathcal{Q}$. Both the adversary $\mathcal{A}$ and the primitive $f$ are treated as a black-box, and $G$ is the black-box construction of $\mathcal{P}$ from $f$.

In our example, given an adversary that breaks a CPA-secure encryption scheme, the security reduction *is* a general technique that constructs an adversary which breaks the underlying PRF given oracle access to the adversary as well as the PRF.

2. **Semi Black-Box Reductions:** In a semi black-box reduction, for any instance $G^f$ of $\mathcal{P}$, if an *efficient* adversary $\mathcal{A}^f$ breaks $G^f$, then there is an algorithm $\mathcal{S}^f$ that breaks the instance $f$ of $\mathcal{Q}$. Here, $\mathcal{S}^f$ is not generic: it is tailor-made for $\mathcal{A}$ and $f$. [RTV04] show that the black-box separation of Impagliazzo and Rudich is a semi black-box separation.
3. **Weakly Black-Box Reductions:** In a weakly black-box reduction, for any instance $G^f$ of $\mathcal{P}$, if an efficient adversary $\mathcal{A}$ (without oracle access to $f$) breaks $G^f$, then there is an algorithm $\mathcal{S}^f$ breaking the instance $f$ of $\mathcal{Q}$.

In our example, a semi black-box reduction claims that any efficient adversary which has black-box access to the PRF used in implementing a CPA-secure encryption scheme immediately implies an adversary that has access to the PRF that breaks the PRF. Intuitively, the difference here is that access to the adversary is non-black-box; the simulator can access the code of the adversary. A *weak* black-box separation provides the strongest guarantee. Note that this entire notion is inapplicable to the setting where the construction crucially depends on a *non-black-box* use of the PRF, since in such a case $G^f$ simply does not exist.

For now, it is instructive to see why Theorem 1.1 in fact proves the desired result, that there is no black-box construction of secret-key agreement from one-way permutations. In fact, it might not even be clear at first sight how the result even holds – if $\mathbf{P} = \mathbf{NP}$, then one way functions do not exist!

The way to mitigate this is to let a *random oracle* be an unconditional implementation of a one-way permutation (which is perfectly compatible with a world in which $\mathbf{P} = \mathbf{NP}$, provided all our TMs are polytime). The result constructs a particular adversary which, in this world, breaks any secret-key agreement protocol without breaking the one-way permutation property of the random oracle (which it cannot do with high probability, being a PPT machine). The resulting construction is an adversary which can break any implementation of secret-key agreement from black-box use of OWPs without breaking the underlying one-way permutation. As an immediate corollary we get that a black-box construction of SKA from OWPs implies $\mathbf{P} \neq \mathbf{NP}$.

## 2 Preliminaries

We will begin by defining *secret key agreement*. Let $\mathcal{A}(\cdot; r)$ and $\mathcal{B}(\cdot; r)$ denote PPT machines Alice and Bob respectively, equipped with random string $r$. We denote by $\mathcal{A}^{\mathcal{O}}(\cdot, \cdot)$ and $\mathcal{B}^{\mathcal{O}}(\cdot, \cdot)$ *oracle* PPT machines Alice and Bob respectively. Throughout the protocol, Alice and Bob will send messages to each other and output a string at the end of their interaction.

**Definition 2.1** (Secret Key Agreement Protocol)**.** *An $\alpha(\ell)$-secret key agreement protocol is defined as a pair $(\mathcal{A}, \mathcal{B})$ such that*

$$\Pr[\langle \mathcal{A}(1^\ell; r_A) \rangle_{\mathcal{B}} = \langle \mathcal{B}(1^\ell; r_B) \rangle_{\mathcal{A}}] \geq \alpha(\ell)$$

*where the probability is taken over the choice of Alice's and Bob's random bits, and where for any two interactive PPT machines $X$ and $Y$, $\langle X(x) \rangle_Y$ is defined as the output of $X$ on completing*

*interaction with $Y$ given input $x$.*

We note that we have not defined secret-key agreement with any kind of security property. Denote by $\text{TRANS}(\mathcal{A}, \mathcal{B})$ the transcript of Alice and Bob's communication, which consists of all the protocol messages sent to each other, and by $\langle \mathcal{A}, \mathcal{B} \rangle$ the (private) output of their interaction. We say that a PPT machine $\mathcal{E}$ can *break* an $\alpha(\ell)$-secure key agreement protocol if

$$\Pr[\mathcal{E}(\text{TRANS}(\mathcal{A}, \mathcal{B})) = \langle \mathcal{A}(1^\ell), \mathcal{B}(1^\ell) \rangle] \geq \frac{\alpha(\ell)}{\mathsf{poly}(\ell)}$$

for some polynomial $\mathsf{poly}(\cdot)$. A secret-key agreement protocol is *secure* if there exists no PPT machine $\mathcal{E}$ which can break it.

## 2.1 One-Way Functions

We will recall the concepts of one-way functions and one-way permutations here. While the 'core' idea of the separation is based on a separation using random oracles, a random oracle is an implementation of a one-way-permutation. We provide a full treatment of what random oracle properties are required in its own section. Here, we briefly recall the definitions of one-way functions and one-way permutations, paraphrased from [Gol01].

**Definition 2.2** (One-Way Function)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ is termed strongly one-way if*

1. *there exists a polynomial time algorithm $A$ such that on input $x$, $A$ outputs $f(x)$, ie. $A(x) = f(x)$, and*
2. *for every PPT algorithm $\mathbf{Adv}$, and all sufficiently large $n$*

$$\Pr[\mathbf{Adv}(f(U_n), 1^n) \in f^{-1}(f(U_n))] \leq \mathsf{negl}(n).$$

*A one-way permutation is an OWF which is* 1-1 *and onto for $n$-bit strings.*

## 2.2 Uniform Generation

We will make heavy use of the concept of *uniform generation*. Intuitively, the uniform generation paradigm allows an oracle TM to efficiently generate a random computation of a machine that is compatible with a certain set of inputs and outputs, or with any communication transcript. This property is essential to our proof: our adversary $\mathcal{E}$ will have to sample random possible computations of Alice and Bob wherever required.

**Definition 2.3** (Uniform Generation)**.** *Consider any polynomial-time decidable relation $R$. We say a TM $M$ uniformly generates $R$ if given $x$, $M$ can output a uniformly chosen $y$ such that $xRy$ with high probability.*

We can allow the probability to be exactly $1/2$; if such a $y$ does note exist, or $M$ fails to output such a $y$, it outputs $\perp$ instead. It is easily observed that the probability can be quite easily boosted by performing polynomially many repetitions. The following theorem was proved in [JVV86].

**Theorem 2.1** (Jerrum, Valiant and Vazirani)**.** *For any polynomial-time relation, there exists a PPT machine equipped with a $\Sigma_2^p$ oracle that can uniformly generate it.*

**Theorem 2.2.** *If $\mathbf{P} = \mathbf{NP}$ then there exist PPTMs such that*

1. *it is possible to generate random potential computations of a PPTM $M$, given its output and input in expected polytime.*

2. *given a transcript* $\textsc{trans}(\mathcal{A}, \mathcal{B})$*, it is possible to uniformly generate a possible computation of both* $\mathcal{A}$ *and* $\mathcal{B}$ *in expected polytime.*

*Proof.* If $\mathbf{P} = \mathbf{NP}$ then the polynomial hierarchy collapses, and so a $\Sigma_2^p$ oracle can be simulated in polytime. Note that both verifying that a particular computation is consistent with a particular input/output pair (just simulate the underlying machine) and verifying that a computation is consistent with a transcript (simulate the execution of one machine directly, and for the other just use the messages sent by it in the transcript) is doable in polytime. Thus, both of these are polytime relations, and the sets can be uniformly generated. □

### 2.3 Probability

We finally state a few basic results on probability which will be useful in our analysis.

**Theorem 2.3** (Counting in a Binary Matrix)**.** *Suppose that we have a binary matrix where at least* $1 - \alpha$ *portion of the elements are* 1*s. Then for* $ab = \alpha$*, there exist* $(1 - a)$ *columns such that a* $(1 - b)$ *fraction of their elements are* 1*s.*

*Proof.* Suppose that this is not the case, ie. there is no group of $1 - a$ columns for which this is true. Then there exist $> a$ columns which have a greater than $b$ fraction of 0s. This means the number of 0s is greater than an $ab = \alpha$ portion, a contradiction. □

**Theorem 2.4** (Markov's Inequality)**.** *Let* $X$ *be a non-negative random variable and* $a > 0$*. Then*

$$\Pr[X \geq a] \leq \frac{\mathbb{E}(X)}{a}.$$

**Theorem 2.5** (Borel-Cantelli Lemma)**.** *Let* $\{E_n\}$ *be a sequence of events. Then if the sum of the probability of the events is finite, ie*

$$\sum_{n=1}^{\infty} \Pr[E_n] < \infty$$

*the probability that infinitely many of them occur is* 0*.*

## 3 Random Oracles

We will now provide a simple review of certain facts about random oracles. The following propositions are folklore and can be safely skipped, though the proofs are nontrivial and worthy of being repeated. In general we will consider $R : \{0,1\}^* \to \{0,1\}^*$.

**Lemma 3.1.** *Consider any oracle PPTM* $T$*. Then there is a polynomial* $\mathsf{poly}$ *such that for any* $n$ *and input* $x \in \{0,1\}^n$

$$\Pr_{R,r}[R(T^R(x;r)) = x] \leq \frac{\mathsf{poly}(n)}{2^n}$$

*where* $R$ *is a randomly chosen function.*

*Proof.* Assume that $r$ is fixed. Then if $T$ never queries $R$ on some $y$ for which $R(y) = x$ the output of $T^R(x)$ is independent of $R$, and hence the probability that $R(T^R(x;r)) = x$ is uniform and exactly $1/2^n$. Furthermore, the probability that *any* queried input is a $y$ such that the condition holds is $1/2^n$. $T$ is a PPT and so it can query at most $\mathsf{poly}(n)$ inputs and so the probability that the output is some input for which the relation holds is $(\mathsf{poly}(n) + 1)/2^n$. □

**Lemma 3.2.** *Consider any oracle PPTM $T$. Then there is a polynomial* poly *such that for nearly all oracles $R$, you can invert with probability better than* $\mathsf{poly}(n)/2^n$ *only for finitely many $n$.*

*Proof.* First, we will show that the fraction of random oracles for which it is possible to invert with probability better than $\mathsf{poly}(n)/2^n$ is low. Consider any fixed $n$. We know that there exists some polynomial poly such that

$$\Pr_{R,r}[R(T^R(x;r)) = x] \leq \frac{\mathsf{poly}(n)}{2^n}$$

by the previous proposition. Define this probability to be $P_R$ for some fixed $R$. Then the expectation of $P_R$ over random choices of $R$ is at most $\mathsf{poly}(n)/2^n$, which is merely the result of the previous question. Using Markov's inequality, we can bound

$$\Pr[P_R \geq \frac{n^2\mathsf{poly}(n)}{2^n}] \leq \frac{\mathsf{poly}(n)}{2^n} \cdot \frac{2^n}{n^2\mathsf{poly}(n)} = \frac{1}{n^2}.$$

Absorb the $n^2$ term into $\mathsf{poly}(n)$. Then at most $1/n^2$ fraction of the random oracles do not satisfy the property for any given $n$. Since $\sum_{n=0}^{\infty} 1/n^2$ converges, the Borel-Cantelli lemma concludes that with probability 1 a random oracle cannot invert better than $\mathsf{poly}(n)/2^n$. □

**Theorem 3.1.** *With probability* 1*, the function associated with a random oracle is one-way.*

*Proof.* Proof follows from the previous proposition; the function is unconditionally one-way. □

We will now prove a samplability lemma which will prove very useful in our analysis; informally, it states that given the state of $\mathcal{A}^{\mathcal{O}}$ at any moment in the protocol, it is possible to uniformly sample a computation that brought her there.

For the purposes of this lemma, we will assume that the random oracle has not been 'pre-fixed', ie. it is being generated on the fly during the computation. While not strictly necessary, it allows us to be more lenient with the proof.

**Lemma 3.3** (Uniform Generation of RO Transcripts)**.** *Assume* $\mathbf{P} = \mathbf{NP}$*. Let* TRANS *be the transcript of any conversation between $\mathcal{A}^{\mathcal{O}}$ and $\mathcal{B}^{\mathcal{O}}$, and let $E$ be any set of oracle query-answer pairs. Denote by $\mathcal{A}_E^{\mathcal{O},\mathrm{TRANS}}$ the probability space of possible computations of $\mathcal{A}^{\mathcal{O}}$ which are consistent with* TRANS *and $E$. Then it is possible to pick a uniform element of $\mathcal{A}_E^{\mathcal{O},\mathrm{TRANS}}$ in expected polynomial time.*

*Proof.* Note that the outputs to an oracle being generated on the fly are no different than purely random elements feeded as input. In particular, consider the machine $\mathcal{A}(\cdot; r')$ which first copies down a polynomial length string to its tape directly from some part of $r'$ and then uses that string as outputs to oracle queries. This machine is indistinguishable from an $\mathcal{A}^{\mathcal{O}}$ in which the oracle outputs are being generated on the fly. Then reducing $\mathcal{A}^{\mathcal{O}}$ to a PPT $\mathcal{A}$, it is possible to check membership in the set $\mathcal{A}_E^{\mathcal{O},\mathrm{TRANS}}$ simply by checking whether $\mathcal{A}$ is consistent with the transcript and with the hard-coded oracle queries $E$. It follows from Theorem 2.2 that this set can be uniformly generated in expected polynomial time. □

It is worth providing a bit of exposition at this point. Note that the theorem of [JVV86] is quite far-reaching; in particular it applies to PPT machines. In some sense this machine is merely sampling from the random strings whose computations are consistent with any specified input/output, since the only variance in a computation of a PPTM with a specified input/output pair is over its input randomness.

# 4 Separating OWPs and Secret-Key Agreement

In this section we will dive into the details of the proof of Theorem 1.1. The precise statement we will show is that a black-box reduction from SKA to OWF implies $\mathbf{P} = \mathbf{NP}$. We first note that we can assume any secret-key agreement protocol to be in the following normal form.

**Lemma 4.1** (Normal Form of Secret-Key Agreement)**.** *Consider any secret-key agreement protocol that runs in $r(\lambda)$ rounds with total runtime $t(\lambda)$ and a maximum of $q(\lambda)$ queries made by each party can be converted to to a time $O(r(\lambda)q(\lambda) + t(\lambda))$ protocol that has the following normal form.*

*In any given round, the speaking party:*

1. *makes a single oracle query,*
2. *performs its computation,*
3. *sends a message,*
4. *waits for a response*

*in that order.*

*Proof.* We can make new machines $\mathcal{A}'$ and $\mathcal{B}'$ which simulate $\mathcal{A}$ and $\mathcal{B}$ but with slight modifications. WLOG consider $\mathcal{A}$. If there are multiple queries made in a round, then $\mathcal{A}'$ makes one oracle query, does whatever computation it needs to do, then sends a message stating $\mathsf{wait}$. $\mathcal{B}'$ then makes an arbitrary query and replies $\mathsf{ok}$. $\mathcal{A}'$ then makes its second oracle query and they proceed until it finally sends the message that $\mathcal{A}$ would have sent. It is easy to see that the probability of success is identical. □

## 4.1 Overview of the Proof

We now provide a brief description of how the proof will proceed.

Let Alice and Bob partake in any key agreement protocol, and let Eve be an adversary. We will show that in a world where $\mathbf{P} = \mathbf{NP}$ and the parties possess a random oracle $\mathcal{O}$, one-way permutations exist – this is merely what the oracle is providing – and yet no secure key agreement is possible. In particular, Eve will be able to (with high enough probability) determine whatever key was agreed upon simply by monitoring the transcript of Alice and Bob's communication, regardless of what the actual protocol is.

Eve will proceed via an algorithm that breaks any secret agreement. This algorithm is PPT if $\mathbf{P} = \mathbf{NP}$. Let $\lambda = 1/\mathsf{poly}(l)$ be the security parameter and let it be Eve's probability of failure. At a high level, Eve's algorithm performs $\lceil 3(n/\lambda)\log(2n/\lambda)\rceil$ invocations of the following steps at every round of the protocol:

- First, Eve samples some possible computation of the speaker (WLOG let it be Alice) consistent with the transcript thus far. This simulated computation will have certain oracle queries; they will be made up, and not necessarily consistent with $\mathcal{O}$, because Lemma 3.3 only allows sampling over some space of *determined* oracle queries: since Eve is not privy to the actual query-answer pairs (which are internal states of Alice, and cannot necessarily be determined from the protocol transcript) it can only guess them.
- Eve then takes the 'guessed' oracle queries and inputs them into the *actual* oracle. She now has a potential computation of Alice that is consistent with both the Oracle and with the transcript thus far.

Without loss of generality, we can assume that the output of the computation is an Oracle query; in particular, we can add an extra oracle query at the end so that whatever 'secret' has been agreed upon by Alice and Bob is then queried at any the oracle and the output is the new secret (and the final output of the computation). This query is an *intersection* query: it is queried by both Alice and Bob.

We will show that the Eve's algorithm with very high probability is able to determine all the intersection queries. It immediately follows that Eve has a polynomial-size list that contains the output of the protocol. Since this works for arbitrary key agreement protocols, we can conclude that key agreement does not exist in this world.

## 4.2 Notation and Definitions

It is worth going over this section carefully; there is quite a bit of notation that is introduced. The purpose of the notation is to carefully understand all moving parts at any given step during the protocol. From here on out we implicitly assume any TM is an oracle TM.

First, we set the $\mathsf{state} = \langle l, r_A, r_B, r_E, \mathcal{O} \rangle$ of the protocol to be a complete determination of the protocol execution; it is determined by the length of the secret $l$, the private randomness and the (finite) random oracle. Denote by $\mathcal{S}_l$ to be the sample space of all possible states of a protocol with length input $l$. Note that $\mathcal{S}_l$ is a probability space.

### 4.2.1 Round Variables

Let $\text{TRANS}_r$ be the transcript up to and including round $r$ and $q_r$ be the query asked in round $r$. Let $A_r$ be the query-answer pairs Alice knows upto and including round $r$ and let $B_r$ be the query-answer pairs Bob knows.

We say that $\mathsf{state} \vdash \text{TRANS}_r$ if the two are consistent.

### 4.2.2 Sample Spaces

We say that $A_r \vdash \langle \text{TRANS}_r, \mathcal{O} \rangle$ if the set of oracle queries/answers known by Alice is consistent with the transcript up to round $r$ and with the provided oracle.

We let $E$ be any set of query-answer pairs; it need not be consistent with the oracle, just any set of string pairs of the correct length. We denote by $AS_E^{\text{TRANS}_r}$ the set of $\langle \mathcal{O}, r_A \rangle$ pairs which are consistent with $E$ and $\text{TRANS}_r$. This is different from $\mathcal{A}_E^{\text{TRANS}_r}$, which is the set of *computations* of Alice consistent with $E$ and $\text{TRANS}_r$. There is functionally no difference between the two: $\mathcal{A}_E^{\text{TRANS}_r}$ is induced from $AS_E^{\text{TRANS}_r}$, in particular any element of the latter uniquely determines an element of the former, although the latter has an *exponential-size* oracle while the former only has a polysize segment of an oracle.

## 4.3 Algorithm

We now state a few terms relevant to Eve's algorithm. Recall that the algorithm performs $m = \lceil 3(n/\lambda)\log(2n/\lambda) \rceil$ invocations per round.

Suppose Alice is the speaker in round $r$. Then denote with $E_{r,i-1}$ the query-answer pairs that Eve knows about the oracle, ie. $\langle q, a \rangle \in E_{r,i-1}$ iff sometime before the $i$th invocation during round $r$, Eve queried $\mathcal{O}(q) = a$. Finally, denote by $BQ_{r,i}$ the query-answer pairs that Bob knows but Eve does not after the $i$th invocation of round $r$.

More formally, we can state the algorithm as below:

**Eve's Algorithm**

In round $r$, invocation $i$,

1. Eve samples $\mathsf{comp} \xleftarrow{\$} \mathcal{A}^{\text{TRANS}_r}_{E_{r,i-1}}$.
2. Eve asks $\mathcal{O}$ all the queries in $\mathsf{comp}$ that are not covered by $E_{r,i-1}$ and adds their outputs to the set. Name this new set to be $E_{r,i}$.

Continue with $m$ invocations at each round until all $n$ rounds are complete. Return $E_{n,m}$.

## 4.4 Proof

**Theorem 4.1.** *Suppose Alice and Bob carry out the protocol on a length $l$ secret. Then the probability that Eve finds all the intersection queries, ie.*

$$\Pr_{\mathsf{state} \in \mathcal{S}_l} [A_n \cap B_n \subseteq E_{n,m}] \geq 1 - \lambda.$$

*Proof.* Suppose this does not happen, ie. Eve does not find all the intersection queries. Then since there is one oracle query per each round, as per Lemma 4.1, there is a first round $r$ at which Eve fails to anticipate the oracle queries. If this is the case, three things must be true:

- Eve has determined all previous intersection queries.
- $q_{r+1}$ is an intersection query.
- $q_{r+1} \notin E_{r,m}$.

We will show this has probability at most $\lambda/n$, ie. for each round there is no more than a $\lambda/n$ probability that all three of the above are true. Then the statement follows from the union bound.

We now *fix* some $\langle \text{TRANS}_r, E_{r,0}, r_B, BQ_{r,0} \rangle$. We show that for *each* such choice the probability that the *complement* is true is at most $1 - \lambda/n$. Then taking the average over all such states the probability is still at most $1 - \lambda/n$.

Precisely, the complement states that one of the three situations occur:

1. **Situation 1**. Eve does not know all the intersection queries before round $r$. Concretely, this means that
$$A_r \cap BQ_{r,0} \neq \emptyset.$$
Recall that $A_r$ is all of Alice's queries and $BQ_{r,0} = B_r \setminus E_{r,0}$ is all of Bob's private queries before Eve started working on round $r$. The nontrivial intersection means that there is some private query known only to Bob which Alice has asked before.
2. **Situation 2**. $q_{r+1} \notin A_r$. This implies $q_r + 1$ is not an intersection query. The reasoning is simple: $q_{r+1} \in B_{r+1}$ by definition, so it cannot be in $A_{r+1} = A_r$.
3. **Situation 3**. $q_{r+1} \in E_{r,m}$.

Now take the set of states $S := \{\mathsf{state} \in \mathcal{S}_l : \mathsf{state} \vdash \langle \text{TRANS}_r, E_{r,0}, r_B, BQ_{r,0} \rangle\}$. Partition $S$ into $S_1, S_2, S_3$, where $|S_1| + |S_2| + |S_3| = |S|$. The proof will conclude once we have successfully bounded the size of $S$, ie. if the set of states that satisfy the three conditions has size at least $(1 - \lambda/n)|S|$. Note that $q_{r+1}$ is uniquely determined within $S$.

Let us see what the sets are:

- The set $S_1$:
$$S_1 = \{\mathsf{state} \in S : \mathsf{state} \vdash (\text{either } \textbf{Situation } 1 \text{ or } 2)\}.$$
We want this set to be large, since we want to show that most states satisfy the above situations: for any state that Eve satisfies the above situations, either Eve made an error before this round or the round is not an intersection query. In particular, it means that Eve succeeding or failing in round $r$ does not matter.

- The set $S_2$: Formally,
$$S_2 = \{\mathsf{state} \in S : \mathsf{state} \nvdash (\textbf{Situations } 1 \text{ and } 2) \text{ but } \exists \text{ an } i \text{ such that } \Pr_{x \in S}[x \vdash (\textbf{Situations } 1 \text{ or } 2) | x \vdash E_{r,i}^{\mathsf{state}}] > 1 - \lambda/2n\}$$
This is a complicated statement, so let's break it down. Suppose we have a state in $S$ which does not satisfy the first two situations. $S_2$ consists of all such states which also have an additional property: at some $i$th invocation, the set of Q/A pairs that Eve has at the end of the invocation that are consistent with $\mathsf{state}$ are consistent with a very large number of states that satisfy the first two conditions. Note that this is bad; states which satisfy this Q/A set are overwhelmingly likely to satisfy Situations 1 and 2, which means that $S_2$ being large causes problems (this is because $S_2$ explicitly does not satisfy the two situations). Fortunately, we will show the size of $S_2$ is small.

- The set $S_3$: Formally,
$$S_2 = \{\mathsf{state} \in S : \mathsf{state} \nvdash (\textbf{Situations } 1 \text{ and } 2) \text{ but } \exists \text{ an } i \text{ such that } \Pr_{x \in S}[x \vdash (\textbf{Situations } 1 \text{ or } 2) | x \vdash E_{r,i}^{\mathsf{state}}] \leq 1 - \lambda/2n\}$$
These are the same as the states in the previous case except it is not overwhelmingly likely that states consistent with such Q/A sets satisfy Situations 1 and 2. We will show that these sets satisfy Situation 3 with high likelihood, however.

Lemmas 4.2 and 4.3 taken together conclude the proof. The full proofs of the lemma are in [IR89]. Here we provide brief proof sketches.

**Lemma 4.2.** *$P_2$ is small. That is, $|P_2| \leq \lambda|S|/2n$.*

*Proof.* Note that this set is in some sense self-defeating; if other states which do satisfy Situations 1 and 2 are consistent with the Q/A queries at round $i$, then by definition they must be more likely. The proof takes advantage of this. □

**Lemma 4.3.** *$P_3$ satisfies situation 3 with high probability. That is, a $1 - \lambda/2n$ portion of $P_3$ satisfies situation 3.*

*Proof.* This is the main complex claim in the proof. While the actual probabilities are hard to bound, the strategy is that when Eve samples a potential computation of Alice whose queries do not intersect $BQ_{r,i}$, then it is quite likely she finds $q_{r+1}$. However, if she samples one which does intersect $BQ_{r,i}$, then she 'whittles down' the intersection queries, ie. it is quite likely that $BQ_{r,i} \cap \mathcal{A}_{E_{r,i-1}}^{\text{TRANS}_r}$ becomes smaller. Since this is repeated $m$ times, which is quite large, the probability can be bounded successfully. □

The two Lemmas taken together show that the probability of at least one of the situations being satisfied is quite large, ie. at least $(1 - \lambda/n)$ per round. This proves the theorem. □

## References


[BBF13] Paul Baecher, Christina Brzuska, and Marc Fischlin. Notions of black-box reductions, revisited. In *Advances in Cryptology-ASIACRYPT 2013: 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I 19*, pages 296–315. Springer, 2013.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Volume 1*, volume 1. Cambridge university press, 2001.

[HR04] Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology–CRYPTO 2004: 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004. Proceedings 24*, pages 92–105. Springer, 2004.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 44–61, 1989.

[JVV86] Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986.

[RTV04] Omer Reingold, Luca Trevisan, and Salil Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography Conference*, pages 1–20. Springer, 2004.

[Sim98] Daniel R Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in CryptologyEUROCRYPT'98: International Conference on the Theory and Application of Cryptographic Techniques Espoo, Finland, May 31–June 4, 1998 Proceedings 17*, pages 334–345. Springer, 1998.