The Random Oracle Methodology, Revisited
(preliminary version)

Ran Canetti* Oded Goldreich† Shai Halevi‡

March 31, 1998

Abstract

We take a formal look at the relationship between the security of cryptographic schemes in the Random Oracle Model, and the security of the schemes which result from implementing the random oracle by so called "cryptographic hash functions".

The main result of this paper is a negative one: There exist signature and encryption schemes which are secure in the Random Oracle Model, but for which *any implementation* of the random oracle results in insecure schemes.

In the process of devising the above schemes, we consider possible definitions for the notion of a "good implementation" of a random oracle, pointing out limitations and challenges.

Keywords: Cryptography (Encryption and Signature Schemes) and Complexity Theory (use of CS-Proofs).

*IBM Watson, P.O. Box 704, Yorktown Height, NY 10598, USA. E-mail: `canetti@watson.ibm.com`

†Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel. E-mail: `oded@wisdom.weizmann.ac.il`. Work done while visiting LCS, MIT. Partially supported by DARPA grant DABT63-96-C-0018.

‡IBM Watson, P.O. Box 704, Yorktown Height, NY 10598, USA. E-mail: `shaih@watson.ibm.com`

1 Introduction

A popular methodology for designing cryptographic protocols consists of the following two steps. One first designs an *ideal* system in which all parties (including the adversary) have oracle access to a truly random function, and proves the security of this ideal system. Next, one replaces the random oracle by a "good cryptographic hashing function" (such as MD5 or SHA), providing all parties (including the adversary) with the succinct description of this function. Thus, one obtains an *implementation* of the ideal system in a "real-world" where random oracles do not exist. This methodology, explicitly formulated by Bellare and Rogaway [1] and hereafter referred to as the *random oracle methodology*, has been used in many works (see, for example, [7, 20, 11, 17, 1, 15, 2, 18]).

Although the random oracle methodology seems to be useful in practice, it is unclear how to put this methodology on firm grounds. One can indeed make clear statements regarding the operation of the ideal system, but it is not clear what happens when one replaces the random oracle by a function which has a succinct description available to all parties. What one would have liked is (at least a definition of) a class of functions which, when used to replace the random oracle, maintains the security of the ideal scheme. The purpose of this work is to point out fundamental difficulties in proceeding towards this goal. Specifically, we argue that the traditional approach of providing a *single* robust definition which supports a wide range of applications is bound to fail. That is, one should not expect to see definitions such as of pseudorandom generators [3, 21] (or functions [8]), and general results of the type saying that these can be used in any application in which parties are restricted merely by computing resources.

Instead, we suggest that one should proceed by identifying useful (special-purpose) properties of a random oracle, which can be also provided by a fully specified function (or function ensemble), and so yield implementations of certain useful ideal systems. Alas, the results in this paper imply that such properties cannot cover all that is doable in the Random Oracle Model. Specifically, we consider a property which we call "correlation intractability" (which seems to underline heuristics such as the Fiat–Shamir transformation of a three-round identification scheme into a signature scheme [7]), and show that even a minimalistic formulation of this property cannot be obtained by any fully specified function (or function ensemble).

To demonstrate the implications of the above to the security of cryptographic systems, we show that systems whose security relies on the "correlation intractability" of their oracle may be secure in the Random Oracle Model, and yet be insecure when implemented using any fully specified function (or function ensemble). Specifically, we describe schemes for digital signatures and public-key encryption which are secure in the Random Oracle Model, but for which any implementation yields insecure schemes. This refutes the belief that a security proof in the Random Oracle Model means that there are no "structural flaws" in the scheme.

1.1 The Setting

1.1.1 The Random Oracle Model

In a scheme which operates in the Random Oracle Model, all parties (including the adversary) are modeled by probabilistic polynomial-time interactive machines with oracle access. That is, the parties interact with one another as usual interactive machines, but in addition they can make oracle queries. It is postulated that all oracle queries, regardless of the identity of the party making them, are answered by a single function, denoted $\mathcal{O}$, which is uniformly selected among all possible functions. The set of possible functions is determined by a length function, $\ell_{\rm out}(\cdot)$, and by the security parameter of the system. Specifically, on security parameter k we consider functions mapping $\{0,1\}^{\mathrm{poly}(k)}$ to $\{0,1\}^{\ell_{\mathrm{out}}(k)}$.

A set of interactive oracle machines as above corresponds to an *ideal system for one specific application.* The application also comes with a security requirement specifying the adversary's abilities and when it is considered successful. The abilities of the adversary include its computational power (typically, an arbitrary polynomial-time machine) and the ways in which it can interact with the other parties. The *success* of the adversary is defined by means of a predetermined polynomial-time predicate of the *application view*.[1] An ideal system is considered secure if any adversary with the given abilities has only a negligible probability of success.

1.1.2 Implementing an ideal system

Loosely speaking, by "implementing" a particular ideal system we mean using an easy-to-evaluate function f instead of the random oracle. That is, whenever the ideal system queries the oracle with a value x, the implementation instead evaluates the function $f(x)$. Formally defining this notion, however, takes some care. Below we briefly examine (and discard of) the notion of implementation by a single function, and then present the notion of implementation by a function ensemble, which is the notion we use throughout the paper.

Implementation by a single function. Each ideal system (for some specific application), Π, is transformed into a real system (for the same application) as follows. We transform each interactive oracle machine, into a standard interactive machine in the natural manner. That is, each oracle call is replaced by the evaluation of a fixed function f on the corresponding query.[2]

The above system is called an *implementation of* Π *using function* f. The adversary, attacking this implementation, may mimic the behavior of the adversary of the ideal system, by evaluating f at arguments of its choice, but it needs not do so. In particular, it may obtain some global insight into the structure of the function f, and use this insight towards its vicious goals. An implementation is called secure if any adversary attacking it may succeed only with negligible probability, where the success event is defined exactly as in the ideal system (i.e., it is defined by the same polynomial-time computable predicate of the application view).

Using this notion of an implementation, we would like to say that a function f is a "good implementation of a random oracle" if for any ideal system Π, if Π is secure then so is the implementation of Π using f. It is very easy to see, however, that no (single) polynomial-time computable function can provide a good implementation of a random oracle. Consider, for example, a candidate function f. Then, a (contrived) application for which f does not provide a good implementation consists of an oracle machine (representing an honest party) which upon receiving a message m, makes query m to the oracle and reveals its private input if the oracle answers with $f(m)$. Suppose that the adversary is deemed successful whenever the honest party reveals its private input. Clearly, this ideal system is secure (in the Random Oracle Model); however, its implementation using f is certainly not secure.

Implementation by a function ensemble. Implementations by a single function, as discussed above, is the straightforward interpretation of the Random Oracle methodology. A more sophisti-

[1] The application view consists of the initial inputs of all the parties (including the adversary), their internal coin tosses, and all the messages which were exchanged among them.

[2] Formally, the function f also takes as input the security parameter k, so that the function $f_k(\cdot) \stackrel{\mathrm{def}}{=} f(k,\cdot)$ maps $\{0,1\}^{\mathrm{poly}(k)}$ to $\{0,1\}^{\ell_{\mathrm{out}}(k)}$.

cated interpretation is indeed called for. Here one considers the substitution of the random oracle by a function randomly selected from a collection of functions. In this setting, we have a "system set-up" phase, in which the function is selected once and for all, and its description is available to all parties.[3] After this set-up phase, this function is used in place of the random oracle just as above.

A little more precisely, we consider a function ensemble $\mathcal{F} = \{F_k | k \in \mathsf{N}\}$, where $F_k = \{f_s : \{0,1\}^{\mathrm{poly}(k)} \mapsto \{0,1\}^{\ell_{\mathrm{out}}(k)}\}_{s \in \{0,1\}^k}$, such that there exists a polynomial time algorithm which on input s and x returns $f_s(x)$. The implementation of an ideal system, Π, by the function ensemble $\mathcal{F}$ is obtained as follows. On security parameter k, we uniformly select $s \in \{0,1\}^k$, and make s available to all parties including the adversary. Given this initialization phase, we replace each oracle call of an interactive oracle machine by the evaluation of the function f_s on the corresponding query. The resulting system is called an *implementation of* Π *using function ensemble* $\mathcal{F}$.

Again, the adversary may (but need not necessarily) mimic the behavior of the adversary in the Random Oracle Model by evaluating f_s at arguments of its choice. Such a real system is called **secure** if any adversary attacking it has only a negligible probability of success, where the probability is taken over the random choice of s, as well as the coins of all the parties. As before, we would like to say that an ensemble $\mathcal{F}$ provides a "good implementation of a random oracle" if for every ideal system Π, if Π is secure then so is the implementation of Π using $\mathcal{F}$. Notice that in this case, the contrived example from above does not work anymore, since the success event must be independent of the random choice of s. Nonetheless, the results which we obtain in this work imply that no function ensemble can provide a good implementation of a random oracle. We elaborate in the next subsection.

1.2 Our Results

1.2.1 Correlation intractability.

One property we certainly expect from a good implementation of a random oracle is that it should be infeasible to find inputs to the function which stand in some "rare" relationship with the corresponding outputs. Indeed, many applications of the random-oracle methodology (such as the Fiat-Shamir heuristic) assume that it is infeasible to find an input-output pair which stands in a particular relations induced by the application. Trying to formulate this property, we may require that given the description of the function it is hard to find a sequence of preimages which together with their images (under this function) satisfy some given relation. Clearly, this can only hold for relations for which finding such sequences is hard in the Random Oracle Model. That is, IF it is hard to find a sequence of preimages which together with their images under a random oracle satisfy relation R, THEN given the description of a "good" function f_s it should be hard to find a sequence of preimages which together with their images under f_s satisfy R.

In fact, in the sequel we only consider the task of finding a single preimage which together with its image satisfies some property. Loosely speaking, a relation is called **evasive** if when given access to a random oracle $\mathcal{O}$, it is infeasible to find a string x so that the pair $(x, \mathcal{O}(x))$ is in the relation. A function ensemble $\mathcal{F}$ (as above) is called **correlation intractable** if for every evasive relation, given the description of a uniformly selected function $f_s \in F_k$ it is infeasible to find an x such that $(x, f_s(x))$ is in the relation. We show that

Informal Theorem 1.1 *There exist no correlation intractable function ensembles.*

[3] In the sequel we consider examples of public key signature and encryption schemes. In these schemes, the initialization step is combined with the key-generation step of the original scheme.

Restricted correlation intractability. The proof of the above negative result relies on the fact that the description of the function is shorter than its input. Thus we also investigate the case where one restricts the function f_s to inputs whose length is less than the length of s. We show that the negative result can be extended to the case where the function description is shorter than the sum of the lengths of the input and output of the function. (Furthermore, if one generalizes the notion of correlation intractability to relations on sequences of inputs and outputs, then the negative result holds as long as the total length of all the inputs and outputs is more than the length of the function description.) This still leaves open the possibility that there exist function ensembles that are correlation intractable with respect to input-output sequences of a-priori bounded total length. See further discussion in Section 5.

1.2.2 A failure of the Random Oracle Methodology

Upon formulating the random oracle methodology, Bellare and Rogaway did warn that a proof of security in the Random Oracle Model should NOT be taken as guarantee to the security of implementations (in which the Random Oracle is replaced by functions such as MD5) [1]. However, it is widely believed that a security proof in the Random Oracle Model means that there are no "structural flaws" in the scheme. That is, any attack against an implementation of this scheme must take advantage of specific flaws in the function which is used to implement the oracle. In this work we demonstrate that these beliefs are false. Specifically, we show that

Informal Theorem 1.2 *There exists encryption and signature schemes which are secure in the Random Oracle Model, but have* NO SECURE IMPLEMENTATION *in the real model (where a Random Oracle does not exist). That is, implementing these secure ideal schemes, using any function ensemble, results in insecure schemes.*

The encryption and signature schemes presented to prove Theorem 1.2 are "unnatural". We do not claim (or even suggest) that a statement as above holds with respect to schemes presented in the literature. Still, the lesson is that the mere fact that a scheme is secure in the Random Oracle Model does not necessarily imply that a particular implementation of it (in the real world) is secure, or even that this scheme does not have any "structural flaws". Furthermore, unless otherwise justified, such ideal scheme may have *no secure implementations at all.*

1.3 Related Work

Our definition of correlation-intractability is related to a definition by Okamoto [17]. In terms of our terminology, Okamoto considers function ensembles for which it is infeasible to form input-output relations with respect to a specific evasive relation [17, Def. 19] (rather than all such relations). He uses the assumption that such function ensembles exists, for a specific evasive relation in [17, Thm. 20].

First steps in the direction of identifying and studying useful special-purpose properties of the Random Oracle Model have been recently taken by Canetti [4]. Specifically, Canetti considered a property called "perfect one-wayness", and provided a definition of this property, constructions which possess this property (under some reasonable assumptions), and applications for which such functions suffice. Additional constructions have been recently suggested by Canetti, Micciancio and Reingold [5].

Our proof of Theorem 1.2 uses CS-proofs (in the Random Oracle Model), as defined and constructed by Micali [15], in an essential way. (The underlying ideas of Micali's construction [15] can be found in Kilian's construction [14].)

1.4 Organization

In Section 2 we present syntax necessary for the rest of the paper. In Section 3 we discuss the reasoning that led us to define the correlation intractability property, and prove that even such a minimalistic definition cannot be met by a function ensemble. In Section 4 we present our main negative results – demonstrating the existence of secure ideal signature and encryption schemes which do not have secure implementations. Restricted correlation intractability is defined and studied in Section 5.

2 Function Ensembles

To make the discussion in the Introduction more precise, we explicitly associate a length function, $\ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$, with the random oracle and its candidate implementations. We always assume that the length functions are superlogarithmic and polynomially bounded (i.e. $\omega(\log k) \leq \ell_{\text{out}}(k) \leq \text{poly}(k)$). We refer to an oracle with length function ℓ_{out} as an ℓ_{out}-**oracle**. On security parameter k, such an oracle answers each query with a string of length $\ell_{\text{out}}(k)$. A candidate implementation of a random ℓ_{out}-oracle is an ℓ_{out}-ensemble as define below.

Definition 1 (function ensembles) *Let $\ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$ be a length function. An ℓ_{out}-*ensemble *is a sequence $\mathcal{F} = \{F_k\}_{k \in N}$ of families of functions, $F_k = \{f_s : \{0,1\}^* \mapsto \{0,1\}^{\ell_{\text{out}}(k)}\}_{s \in \{0,1\}^k}$, so that the following holds*

Length requirement. *For every $s \in \{0,1\}^k$ and every $x \in \{0,1\}^*$, $|f_s(x)| = \ell_{\text{out}}(k)$.*

Efficiency. *There exists a poly-time algorithm* EVAL *so that for all $s, x \in \{0,1\}^*$, $\text{EVAL}(s, x) = f_s(x)$.*

In the sequel we often call s the description *or the* seed *of the function f_s.*

Remark 1. The length of the seed in the above definition serves as a "security parameter" and is meant to control the "quality" of the implementation. It is important to note that although $f_s(\cdot)$ is syntactically defined on every input, in a cryptographic applications it is only used on inputs of length at most $\text{poly}(|s|)$. We stress that all results presented in this paper refer to such usage.

Remark 2. One may even envision applications in which a more stringent condition on the use of f_s holds. Specifically, one may require that the function f_s be only applied to inputs of length at most $\ell_{\text{in}}(|s|)$, where $\ell_{\text{in}} : \mathsf{N} \mapsto \mathsf{N}$ is a specific (polynomially bounded) length function (e.g., $\ell_{\text{in}}(k) = 2k$). We discuss the effects of making such a stringent requirement in Section 5.

3 Correlation Intractability

In this section we present and discuss the difficulty of defining the requirements that a function ensemble "behaves like a random oracle" even when its description is given. In particular, we show that even minimalistic definitions cannot be realized.

An obvious failure. We first comment that an obvious maximalistic definition, which amount to adopting the pseudorandom requirement of [8], fails poorly. That is, we cannot demand that an (efficient) algorithm that is given the description of the function cannot distinguish its input-output behavior from the one of a random function. This is so since the function description determines its input-output behavior.

Towards a minimalistic definition. Although we cannot require the value of a fully specified function to be "random", we may still be able to require that it has some properties. For example, we may require that given a description of a family and a function chosen at random from a this family it is hard to find two preimages which the function maps to the same image. Indeed, this sound definition coincides with the well-known *collision-intractability* property [6]. Trying to generalize, we may replace the "equality of images" relation by any other relation among the pre-images and images of the function. Namely, we would like to say that an ensemble is *correlation intractable* if for *any* relation, given the description of a randomly chosen function, it is infeasible to find a sequence of preimages which together with their images satisfy this relation.

This requirement, however, is still unreasonably strong since there are relations which are easy to satisfy even in the Random Oracle Model. We therefore restrict the above infeasibility requirement by saying that it holds only with respect to relations which are hard to satisfy in the Random Oracle Model. That is, IF it is hard to find a sequence of preimages which together with their images under a random function satisfy relation R, THEN given the description of a randomly chosen function f_s it should be hard to find a sequence of preimages which together with their images under f_s satisfy R.

This seems to be a minimalistic notion of correlation intractable collection of functions, yet we show below that no collection can satisfy it. In fact, in the definition below we only consider the task of finding a single preimage which together with its image satisfies some property. Namely, instead of considering all possible relations, we only consider binary ones. Since we are showing impossibility result, this syntactic restriction only strengthens the result.

3.1 Definitions

We start with a formal definition of a relation which is hard to satisfy in the random oracle model.

Definition 2 (Evasive Relations) *A binary relation R is said to be* evasive *with respect to length function $\ell_{\rm out}$ if for any probabilistic polynomial time oracle machine M*

$$\Pr_{\mathcal{O}}[x \leftarrow M^{\mathcal{O}}(1^k),\ (x, \mathcal{O}(x)) \in R] \;=\; \mathrm{negl}(k)$$

where $\mathcal{O} : \{0,1\}^* \mapsto \{0,1\}^{\ell_{\rm out}(k)}$ is a uniformly chosen function and $\mathrm{negl}(\cdot)$ *is a negligible function.*[4]

A special case of evasive relations consists of R's for which there exists a negligible function $\mathrm{negl}(\cdot)$ so that for all k

$$\max_{x\in\{0,1\}^*} \left\{ \Pr_{y\in\{0,1\}^{\ell_{\rm out}(k)}}[(x,y)\in R] \right\} \;=\; \mathrm{negl}(k)$$

(All the relations used in the sequel falls into this category.) The reason such an R is evasive is that any oracle machine, M, making at most $\mathrm{poly}(k)$ queries to a random $\mathcal{O}$ satisfies

$$\begin{aligned}\Pr_{\mathcal{O}}[x \leftarrow M^{\mathcal{O}}(1^k),\ (x, \mathcal{O}(x)) \in R] \;&\le\; \mathrm{poly}(k)\cdot \max_{x\in\{0,1\}^*}\{ \Pr_{\mathcal{O}}[(x,\mathcal{O}(x))\in R] \} \\ &\le\; \mathrm{poly}(k)\cdot \mathrm{negl}(k)\end{aligned}$$

We are now ready to state our minimalistic definition of a correlation intractable ensemble:

[4] A function $\mu : \mathsf{N} \mapsto R$ is negligible if for every positive polynomial p and all sufficiently large n's, $\mu(n) < 1/p(n)$.

Definition 3 (correlation intractability) *Let $\ell_{\text{out}} : N \to N$ be length function, and let $\mathcal{F}$ be an ℓ_{out}-ensemble. We say that $\mathcal{F}$ is* correlation intractable *if for every polynomial time machine M and every evasive relation R (w.r.t. ℓ_{out}), it holds that*

$$\Pr_{s\in\{0,1\}^k}[x \leftarrow M(s),\ \ (x, f_s(x)) \in R] = \text{negl}(k)$$

where $\text{negl}(\cdot)$ *is a negligible function, and the probability is taken over the choice of $s \in \{0,1\}^k$ and the coins of M.*

Remark 3. In the above definition we quantify over all evasive relations. A weaker notion, called weak correlation intractability, is obtained by quantifying only over all polynomial-time recognizable evasive relations (i.e., we only consider those relations R such that there exists a polynomial time algorithm which given (x, y) decides whether or not $(x, y) \in R$). In the sequel we consider both notions.

3.2 Correlation-intractable ensembles do not exist

Theorem 4 *There exist no correlation intractable ensembles, not even in the weak sense.*

Proof: Let ℓ_{out} be a length function and let $\mathcal{F} = \{f_s\}$ be an ℓ_{out}-ensemble. We define the binary relation

$$R^{\mathcal{F}} \stackrel{\text{def}}{=} \bigcup_k \{(s, f_s(s)) : s \in \{0,1\}^k\} \tag{1}$$

Clearly, this relation is polynomial-time recognizable, since f_s can be computed in polynomial time. Also, the relation is evasive (w.r.t. ℓ_{out}) since for every $x \in \{0,1\}^*$ there is at most[5] one $y \in \{0,1\}^{\ell_{\text{out}}(k)}$ satisfying $(x, y) \in R^{\mathcal{F}}$, and so

$$\Pr_y[(x, y) \in R^{\mathcal{F}}] \le 2^{-\ell_{\text{out}}(k)} = 2^{-\omega(\log k)} = \text{negl}(k)\,.$$

On the other hand, consider the machine I which computes the identity function, $I(x) = x$ for all x. It violates the correlation intractability requirement, since for all k,

$$\Pr_{s\in\{0,1\}^k}[(I(s), f_s(I(s))) \in R^{\mathcal{F}}] \ = \ 1\,.$$

In fact, since $R^{\mathcal{F}}$ is polynomial-time recognizable, then even the weak correlation intractability requirement is violated. ■

4 A failure of the Random Oracle Methodology

In this section we show that the security of a cryptographic scheme in the Random Oracle Model may not always imply its security under some specific choice of "good hash function" which implements the random oracle. To prove this statement we construct signature and encryption schemes, which are secure in the Random Oracle Model, yet for which *any implementation* of the random oracle yield insecure schemes. Put in other words, although the ideal scheme is secure, any implementation of it is necessarily insecure.

[5]Such a y exists if and only if $\ell_{\text{out}}(|x|) = \ell_{\text{out}}(k)$.

The underlying idea is to start with a secure scheme (which may or may not use a random oracle) and modify it to get a scheme which is secure in the Random Oracle Model, but such that its security is easily violated when trying to replace the random oracle by any ensemble. This is done by using evasive relations as constructed in Theorem 4. The modified schemes start by trying to find a preimage which together with its image yields a pair in the evasive relation. In case the attempt succeeds the scheme does something which is clearly insecure (e.g., apply the identity transformation to the message, or output the secret key). Otherwise, the scheme behaves as the original (secure) scheme. The former case (i.e., finding a pair in the relation) will occur rarely in the Random Oracle Model, and thus the scheme will maintain its security there. However, the former case will always occur under an implementation of the Random Oracle Model, thus no implementation may be secure. For clarity of presentation we start with the case of a signature scheme, and present the construction in three steps.

- In the first step we carry out the above idea in a naive way. This allows us to prove a weaker statement, saying that for any function ensemble $\mathcal{F}$, there exists a signature scheme which is secure in the Random Oracle Model, but is not secure when implemented using $\mathcal{F}$.

  This, by itself, means that one cannot construct a function ensemble which provides secure implementation of any cryptographic scheme which is secure in the Random Oracle Model. But it does not rule out the possibility (ruled out below) that for any cryptographic scheme which is secure in the Random Oracle Model there exists a secure implementation (via a different function ensemble).

- In the second step we use diagonalization techniques to reverse the order of quantifiers. Namely, we show that there exists a signature scheme which is secure in the Random Oracle Model, but for which *any* implementation (using any function ensemble) results in an insecure scheme. However, the scheme constructed in this step utilizes signing and verification procedures which are slightly super-polynomial time.

- In the third step we use CS-proofs [15] to get rid of the super-polynomial running-time, hence obtaining a standard signature scheme which is secure in the Random Oracle Model, but has no secure implementation.

  Specifically, in this step we use CS-proofs as a tool to "diagonalize against all polynomial-time ensembles in polynomial time". (As noted by Silvio Micali, this technique may be useful also in other settings where diagonalization techniques are applied.)

The reader is referred to [10] for basic terminology regarding signature schemes and corresponding notions of security. As a starting point for our constructions, we use a signature scheme, denoted $\mathcal{S} = (G, S, V)$, where G is the key-generation algorithm, S is the signing algorithm, and V is the verification algorithm. We assume that the scheme (G, S, V) is existentially unforgeable under adaptive chosen message attack, in the Random Oracle Model. Somewhat surprisingly, we do not need to rely on any cryptographic assumptions here, since such (ideal) schemes can be easily shown to exist using the techniques of [16, 19] (since one-way functions exist in the Random Oracle Model).[6]

[6] Alternatively, we could use an 'ordinary' signature scheme, but then our Theorem 8 would be conditioned on the existence of one-way functions.

Conventions. In the three steps below we assume, without loss of generality, that the security parameter (i.e., k) is implicit in the keys generated by $G(1^k)$. Also, let us fix some length function $\ell_{\text{out}} : \mathbb{N} \mapsto \mathbb{N}$, which would be implicit in the discussions below (i.e., we assume that the random oracles are all ℓ_{out}-oracles, the relations are evasive w.r.t. ℓ_{out}, etc.).

4.1 First Step

Definition. Let $\mathcal{S} = (G, S, V)$ be a signature scheme (which may or may not use a random oracle), and let R be any binary relation, which is evasive w.r.t. length function ℓ_{out}. Then, by $\mathcal{S}_R = (G, S_R, V_R)$ we denote the following modification of $\mathcal{S}$ which utilizes a random ℓ_{out}-oracle:

Modified signature, $S_R^{\mathcal{O}}(\text{sk}, \text{msg})$**,** of message msg using signing key sk:

1. If $(\text{msg}, \mathcal{O}(\text{msg})) \in R$, output (sk, msg).
2. Otherwise (i.e., $(\text{msg}, \mathcal{O}(\text{msg})) \notin R$), output $S^{\mathcal{O}}(\text{sk}, \text{msg})$.

Modified verification, $V_R^{\mathcal{O}}(\text{vk}, \text{msg}, \sigma)$**,** of alleged signature σ to msg using verification key vk:

1. If $(\text{msg}, \mathcal{O}(\text{msg})) \in R$ then `accept`
2. Otherwise output $V^{\mathcal{O}}(\text{vk}, \text{msg}, \sigma)$.

(Note that the key-generation algorithm, G, is the same as in the original scheme $\mathcal{S}$.) Item 1 in the signing/verification algorithms is a harmful modification to the original signature scheme. Yet, if R is evasive, then it has little effect on the ideal system, and the behavior of the modified scheme is "indistinguishable" from the original one. In particular,

Proposition 5 *Suppose that R is evasive* (w.r.t. ℓ_{out}) *and that $\mathcal{S}$ is existentially unforgeable under a chosen message attack in the Random Oracle Model. Then $\mathcal{S}_R$ is also existentially unforgeable under a chosen message attack in the Random Oracle Model.*

Proof Sketch: Since R is evasive, it is infeasible for the forger to find a message m so that $(m, \mathcal{O}(m)) \in R$. Thus, a forgery of the modified scheme must be due to Item (2), which yields a breaking of the original scheme. $\square$

The modification enables to break the modified scheme when implemented with a real ensemble $\mathcal{F}$, in the case where R is the relation $R^{\mathcal{F}}$ from Proposition 4. Indeed, as corollary to Propositions 4 and 5, we immediately obtain:

Corollary 6 *For every efficiently computable ℓ_{out}-ensemble $\mathcal{F}$, there exists an efficient signature scheme which is secure in the Random Oracle Model, yet when implemented with $\mathcal{F}$, the resulting scheme is totally breakable under an adaptive chosen message attack, and existentially forgeable under a key-only attack.*

Proof Sketch: Recall that when we use an ensemble $\mathcal{F}$ to implement the random oracle in the scheme $\mathcal{S}_R$, we obtain the following real scheme (which we denote $\mathcal{S}'_R = (G', S'_R, V'_R)$)

$G'(1^k) \stackrel{\text{def}}{=}$ Uniformly pick $s \in \{0,1\}^k$, set $(\text{sk}, \text{vk}) \leftarrow G^{f_s}(1^k)$, and output $(\langle \text{sk}, s \rangle, \langle \text{vk}, s \rangle)$.

$S'_R(\langle \text{sk}, s \rangle, \text{msg}) \stackrel{\text{def}}{=}$ Output $S_R^{f_s}(\text{sk}, \text{msg})$.

$V'_R(\langle \text{vk}, s \rangle, \text{msg}, \sigma) \stackrel{\text{def}}{=}$ Output $V_R^{f_s}(\text{vk}, \text{msg}, \sigma)$.

Consider now what happens when we use the ensemble $\mathcal{F}$ to implement the the scheme $\mathcal{S}_{R^{\mathcal{F}}}$ (recall the definition of $R^{\mathcal{F}}$ from Eq. (1)). Since $R^{\mathcal{F}}$ is evasive, then from Proposition 5 we infer that the $\mathcal{S}_{R^{\mathcal{F}}}$ is secure in the Random Oracle Model. However, when we use the ensemble $\mathcal{F}$ to implement the scheme, the seed s becomes part of the public verification key, and hence is known to the adversary. The adversary can simply output the pair (s, ϵ), which will be accepted by $V'_{R^{\mathcal{F}}}$ as a valid message-signature pair (since $(s, f_s(s)) \in R^{\mathcal{F}}$). Hence, the adversary achieves existential forgery (of $\mathcal{S}'_{R^{\mathcal{F}}}$) under key-only attack. Alternatively, the adversary can ask the legitimate signer for a signature on s, hence obtaining the secret signing-key (i.e., total forgery). $\square$

4.2 Second Step

Enumeration. Loosely speaking, for this (and the next) subsection we need an enumeration of all efficiently computable function ensembles. Such enumeration is achieved via an enumeration of all polynomial-time algorithms (i.e., candidates for evaluation of such ensembles). Several standard technicalities arise. First, enumerating all polynomial-time algorithms is problematic since there is no single polynomial that bounds the running time of all these algorithms. Instead, we fix an arbitrary super-polynomial proper complexity function[7] $t : \mathbb{N} \mapsto \mathbb{N}$ (e.g., $t(n) = n^{\log n}$), and enumerate all algorithms of running-time bounded by t. The latter is done by enumerating all possible algorithms, and modifying each algorithm by adding a time-out mechanism which terminates the execution in case more than $t(|\text{input}|)$ steps are taken. This modification does not effect the polynomial-time algorithms. Also, since we are interested in enumerating ℓ_{out}-ensembles, we modify each algorithm by viewing its input as a pair $\langle s, x\rangle$ (using some standard parsing rule[8]) and padding or truncating its output to length $\ell_{\text{out}}(|s|)$. Again, this modification has no effect on the ℓ_{out}-ensembles.

Let us denote by $\mathcal{F}^i$ the i^{th} function ensemble according to the above enumeration, and denote by f_s^i the function indexed by s from the ensemble $\mathcal{F}^i$. Below we again use some standard rule for parsing a string α as a pair $\langle i, s\rangle$ and viewing it as a description of the function f_s^i.

Universal ensemble. Let $\mathcal{U}$ denote the "universal function ensemble" which is induced by the enumeration above, namely, $\mathcal{U}(\langle i, s\rangle, x) = f_s^i(x)$. We remark that there exists a machine which computes the universal function $\mathcal{U}$ and works in time t.

Universal relation. Denote by $R^{\mathcal{U}}$ the universal relation which is defined with respect to the universal ensemble $\mathcal{U}$ similarly to the way that $R^{\mathcal{F}}$ is defined with respect to any ensemble $\mathcal{F}$

$$(x, y) \in R^{\mathcal{U}} \iff \begin{array}{l} y = \mathcal{U}(x, x) \\ (\text{i.e., } x = \langle i, s\rangle \text{ and } y = f_s^i(x)) \end{array}$$

Modified signature scheme. Let $\mathcal{S} = (G, S, V)$ be a signature scheme (as above). We then denote by $\mathcal{S}_u = (G, S_u, V_u)$ the modified signature scheme which is derived by using $R^{\mathcal{U}}$ in place of R in the previous construction. Specifically:

$S_u^{\mathcal{O}}(\text{sk}, \text{msg}) \stackrel{\text{def}}{=}$

[7] Recall that $t(n)$ is a *proper complexity function* (or time-constructible) if there exists a machine which computes $t(n)$ and works in time $O(t(n))$. This technical requirement is needed to ensure that the enumeration itself is computable in time $O(t(n))$.

[8] For example, we can parse a string α as $\langle s, x\rangle$ if $\alpha = C(s)x$, where C is any efficiently computable prefix-free encoding.

1. If $(\text{msg}, \mathcal{O}(msg)) \in R^{\mathcal{U}}$ (i.e., if msg $= \langle i, s\rangle$ and $\mathcal{O}(\text{msg}) = f_s^i(\text{msg})$) then output (sk, msg).
2. Otherwise, output $S^{\mathcal{O}}(\text{sk}, \text{msg})$

$V_u^{\mathcal{O}}(\text{vk}, \text{msg}, \sigma) \stackrel{\text{def}}{=}$

1. If $(\text{msg}, \mathcal{O}(msg)) \in R^{\mathcal{U}}$ then `accept`.
2. Otherwise, output $V^{\mathcal{O}}(\text{vk}, \text{msg}, \sigma)$.

We note that since these signature and verification algorithms need to compute $\mathcal{U}$, they both run in time $O(t)$, which is slightly super-polynomial.

Proposition 7 *Suppose that $\mathcal{S}$ is existentially unforgeable under a chosen message attack in the Random Oracle Model. Then $\mathcal{S}_u$ is also existentially unforgeable under a chosen message attack in the Random Oracle Model, but implementing it with* any function ensemble *yields a scheme which is totally breakable under chosen message attack and existentially forgeable under key-only attack.*

Recall that $\mathcal{S}_u$ is not a 'proper' signature scheme, as it operates in (slightly) super-polynomial-time.

Proof Sketch: Since $R^{\mathcal{U}}$ is evasive, then from Proposion 5 it follows that $\mathcal{S}_u$ is secure in the Random Oracle Model. On the other hand, suppose that one tries to replace the random oracle in the scheme by an ensemble $\mathcal{F}^i$ (where i be the index in the enumeration). An adversary, given a seed s of a function in $\mathcal{F}^i$ can then set msg $= \langle i, s\rangle$ and output the pair (msg, ϵ), which would be accepted as a valid message-signature pair by V_u. Alternatively, it can ask the signer for a signature on this msg, and so obtain the secret signing-key. $\square$

4.3 Third step

To eliminate the super-polynomial time in the above signature scheme, we use CS-proofs as defined and constructed by Micali [15]. Recall that in a CS-proof system, a prover, PRV, is trying to convince a verifier, VER, of the validity of an assertion of the type "machine M accepts input x within t steps". The central feature of CS-proofs is that the running-time of the prover on input x is (polynomially) related to the *actual* running time of $M(x)$ (rather than to the global upper bound t), whereas the verifier's running-time is poly-logarithmic relative to t. In our context, we use CS-proofs that work in the Random Oracle Model, where both prover and verifier have access to a random oracle. A construction for such CS-proofs was presented by Micali, and requires no computational assumptions. A more precise formulation of CS-proofs is given in the Appendix.

We use CS-proofs to construct a new signature scheme which works in the Random Oracle Model. This construction is similar to the one in Subsection 4.2, except that instead of checking that $(\text{msg}, \mathcal{O}(\text{msg})) \in R^{\mathcal{U}}$, the signer/verifier gets a CS-proof of that claim, and it only needs to verify the validity of that proof. Since verifying the validity of a CS-proof can be done much more efficiently than checking the claim "from scratch", the signing and verifications algorithms in the new scheme may work in polynomial time. This yields the following

Theorem 8 *There exists a signature scheme which is existentially unforgeable under a chosen message attack in the Random Oracle Model, but such that when implemented with any function ensemble, the resulting scheme is existentially forgeable using key-only attack and totally breakable under chosen message attack.*

We note again that unlike the "signature scheme" presented in Subsection 4.2, the signature scheme presented below works in polynomial-time.

Proof Sketch: Below we describe such a signature scheme. For this construction we use the following ingredients.

- $\mathcal{S} = (G, S, V)$ is a signature scheme, operating in the Random Oracle Model, which is existentially unforgeable under a chosen message attack.
- A fixed (and easily computable) parsing rule which interpret messages as triples of strings $\mathrm{msg} = \langle i, s, \pi \rangle$.
- The algorithms PRV and VER of the CS-proof system described above.
- Access to three independent random oracles. This is very easy to achieve given access to one oracle $\mathcal{O}$; specifically, by setting $\mathcal{O}'(x) \stackrel{\mathrm{def}}{=} \mathcal{O}(00x)$, $\mathcal{O}''(x) \stackrel{\mathrm{def}}{=} \mathcal{O}(01x)$ and $\mathcal{O}'''(x) \stackrel{\mathrm{def}}{=} \mathcal{O}(11x)$.

  Below we use oracle $\mathcal{O}'''$ for the basic scheme $\mathcal{S}$, oracle $\mathcal{O}''$ for the CS-proofs, and oracle $\mathcal{O}'$ for our evasive relation. We note that if $\mathcal{O}$ is an ℓ_{out}-oracle, then so are $\mathcal{O}', \mathcal{O}''$ and $\mathcal{O}'''$.
- The universal function $\mathcal{U}$ from Subsection 4.2, with proper complexity bound $t(n) = n^{\log n}$. We denote by $M_{\mathcal{U}}$ the universal machine which decides the relation $R^{\mathcal{U}}$. That is, on input $(\langle i, s \rangle, y)$, machine $M_{\mathcal{U}}$ invokes the i^{th} evaluation algorithm, and accepts if $f_s^i(\langle i, s \rangle) = y$.

  We note that $M_{\mathcal{U}}$ works in time t in the worst case. More importantly, if $\mathcal{F}^i$ is a function ensemble which can be computed in time $p_i(\cdot)$ (where p_i is some polynomial), then for any strings s, y, machine $M_{\mathcal{U}}$ works on input $(\langle i, s \rangle, y)$ for only $\mathrm{poly}(|i|) \cdot p_i(|s|)$ many steps.[9]

Using all the above, we describe an ideal signature scheme $\mathcal{S}'_u = (G, S'_u, V'_u)$. As usual, the key generation algorithm G remains unchanged. The signature and verification algorithms proceed as follows.

${S'_u}^{\mathcal{O}}(\mathrm{sk}, \mathrm{msg}) \stackrel{\mathrm{def}}{=}$

1. Parse msg as $\langle i, s, \pi \rangle$, and set $x = \langle i, s \rangle$ and $y = \mathcal{O}'(x)$. Let $n = |(x, y)|$.
2. Apply $\mathrm{VER}^{\mathcal{O}''}$ to verify whether π is a valid CS-proof, w.r.t oracle $\mathcal{O}''$ and security parameter 1^{n+k}, for the claim that the machine $M_{\mathcal{U}}$ accepts the input (x, y) within time $t(n)$.
3. If π is a valid proof, output $(\mathrm{sk}, \mathrm{msg})$.
4. Otherwise, output $S^{\mathcal{O}'''}(\mathrm{sk}, \mathrm{msg})$.

${V'_u}^{\mathcal{O}}(\mathrm{vk}, \mathrm{msg}, \sigma) \stackrel{\mathrm{def}}{=}$

1+2. As above

3. If π is a valid proof, then `accept`
4. Otherwise, output $V^{\mathcal{O}'''}(\mathrm{vk}, \mathrm{msg}, \sigma)$.

The computation required in Item 2 of the signature and verification algorithms can be executed in polynomial-time. The reason being that (by definition) verifying a CS-proof can be done in polynomial-time, provided the statement can be decided in at most exponential time (which is the case here since we have $t(n) = O(n^{\log n})$). It is also easy to see that for every pair $(\mathrm{sk}, \mathrm{vk})$ output

[9] The point is merely that, for every fixed i, the expression $\mathrm{poly}(|i|) \cdot p_i(|s|)$ is bounded by a polynomial in $|s|$.

by G, and for every msg and every $\mathcal{O}$, the string $S_u'^{\mathcal{O}}(\mathrm{sk},\mathrm{msg})$ constitutes a valid signature of msg relative to vk and the oracle $\mathcal{O}$.

To show that the scheme is secure in the Random Oracle Model, we first observe that on security parameter 1^k it is infeasible to find a string x so that $(x,\mathcal{O}'(x)) \in R^{\mathcal{U}}$, since $R^{\mathcal{U}}$ is evasive. By soundness of CS-proofs, it is also infeasible to find (x,π) such that $(x,\mathcal{O}'(x)) \notin R_{\mathcal{U}}$ and yet π is a valid CS-proof of the contrary relative to $\mathcal{O}''$ (with security parameter $1^{|x|+\ell_{\mathrm{out}}(k)+k}$). Thus, it is infeasible for a polynomial-time adversary to find a message that would pass the test on Item 2 of the signature/verification algorithms above, and so we infer that the modified signature is secure in the Random Oracle Model.

We now show that for every candidate implementation, $\mathcal{F}$, there exists a polynomial-time adversary effecting total break via a chosen message attack (or, analogously, an existential forgery via a "key only" attack). First, for each function $f_s \in \mathcal{F}$ denote $f_s'(x) \stackrel{\mathrm{def}}{=} f_s(00x)$, $f_s''(x) \stackrel{\mathrm{def}}{=} f_s(01x)$, and $f_s'''(x) \stackrel{\mathrm{def}}{=} f_s(11x)$. Then denote by $\mathcal{F}'$ the ensemble of the f_s' functions.

Suppose that $\mathcal{F}'$ is the i^{th} function ensemble in the enumeration mentioned above, namely $\mathcal{F}' = \mathcal{F}^i$. Given a randomly chosen k-bit seed s, the adversary generate a message $\mathrm{msg} = \langle i,s,\pi\rangle$ so that π is a CS-proof (w.r.t the adequate security parameter) for the *true* statement that $M_{\mathcal{U}}$ accepts the input (x,y) within $t(|x|+|y|)$ steps, where $x = \langle i,s\rangle$ and $y = f_s'(x)$. Recall that the above statement is indeed true since we have

$$\mathcal{U}(x,x) = f_s^i(x) = f_s'(x) = y$$

and hence the adversary can generate a proof for it in time which is polynomial in the time that it takes to compute f_s^i. (Note that by perfect completeness of the CS-proof system, the ability to prove correct statements holds for *any* choice of the random oracle, and in particular when it is equal to f_s''. See Appendix.)

Also note that since this adversary is specifically designed to break the scheme in which the random oracle is implemented by $\mathcal{F}$, then the index i – which depends only on the choice of $\mathcal{F}$ – can be incorporated into the program of this adversary.

By the efficiency condition of CS-proofs, it is possible to find π (given an oracle access to f_s'') in time polynomial in the time that it takes $M_{\mathcal{U}}$ to accept the input (x,y). Since $\mathcal{F}^i$ is polynomial-time computable, then $M_{\mathcal{U}}$ works on the input $(x,y) = (\langle i,s\rangle, y)$ in polynomial time, and thus the described adversary also operates in polynomial-time.

By construction of the modified verification algorithm, ϵ is a valid signature on $\mathrm{msg} = \langle i,s,\pi\rangle$, and so existential forgery is feasible a-priori. Furthermore, requesting the signer to sign msg yields the signing key and thus total forgery. □

Remark 4. Note that, for the above argument, it is immaterial whether CS-proofs can be implemented in the "real world" (i.e., without access to random oracles). Specifically, it doesn't matter if one can cheat when the oracle is substituted by a candidate function ensemble, as in this case (i.e., in the real world implementation) it is sufficient for the adversary to invoke the proof system on valid statements. (We do rely, however, on the perfect completeness of CS-proofs which implies that valid statements can be proven for any possible choice of oracle used in the proof system.)

4.4 Encryption

The construction presented for signature schemes can be adapted to (public-key)[10] encryption schemes in a straightforward way, yielding the following theorem

Theorem 9 *(a) Assume that there exists a public key encryption scheme which is semantically secure in the Random Oracle Model. Then there exists a public key encryption scheme which is semantically secure in the Random Oracle Model but is not semantically secure when implemented with any function ensemble.*

(b) Assume that there exists a public key encryption scheme which is secure under adaptive chosen ciphertext attack in the Random Oracle Model. Then there exists a scheme which is secure under adaptive chosen ciphertext attack in the Random Oracle Model, but implementing it with any function ensemble yields a scheme which is not semantically secure, and in which a chosen ciphertext attack reveals the secret decryption key.

Proof: In this proof we use the same notations as in the proof of Theorem 8. Let $\mathcal{E} = (G, E, D)$ be an encryption scheme which is semantically secure in the Random Oracle Model, and we modify it to get another scheme $\mathcal{E}' = (G, E', D')$. The key generation algorithm remains unchanged, and the encryption and decryption algorithms utilize a random oracle $\mathcal{O}$, which is again viewed as three oracles $\mathcal{O}', \mathcal{O}''$ and $\mathcal{O}'''$.

Modified encryption, ${E'_{\mathrm{ek}}}^{\mathcal{O}}(\mathrm{msg})$**,** of plaintext msg using the public encryption-key ek:

1. Parse msg as $\langle i, s, \pi \rangle$, set $x = \langle i, s \rangle$ and $y = \mathcal{O}'(x)$, and let $n = |(x, y)|$.
2. If π is a valid CS-proof, w.r.t oracle $\mathcal{O}''$ and security parameter 1^{n+k}, for the assertion that $M_{\mathcal{U}}$ accepts the pair (x, y) within $t(n)$ steps, then output $(1, \mathrm{msg})$.
3. Otherwise (i.e., π is not such a proof), output $(2, E_{\mathrm{ek}}^{\mathcal{O}'''}(\mathrm{msg}))$.

Modified decryption, ${D'_{\mathrm{dk}}}^{\mathcal{O}}(c)$**,** of ciphertext c using the private decryption-key dk:

1. If $c = (1, c')$, output c' and halt.
2. If $c = (2, c')$, output $D_{\mathrm{dk}}^{\mathcal{O}'''}(c')$ and halt.
3. If $c = (3, c')$ then parse c' as $\langle i, s, \pi \rangle$, and set $x = \langle i, s \rangle$, $y = \mathcal{O}'(x)$, and $n = |(x, y)|$. If π is a valid CS-proof, w.r.t oracle $\mathcal{O}''$ and security parameter 1^{n+k}, for the assertion that $M_{\mathcal{U}}$ accepts the pair (x, y) within $t(n)$ steps, then output dk.
4. Otherwise output ϵ.

The efficiency of this scheme follows as before. It is also easy to see that for every pair $(\mathrm{ek}, \mathrm{dk})$ output by G, and for every plaintext msg, ${D'_{\mathrm{dk}}}^{\mathcal{O}}({E'_{\mathrm{ek}}}^{\mathcal{O}}(\mathrm{msg})) = \mathrm{msg}$ holds for every $\mathcal{O}$. To show that the scheme is secure in the Random Oracle Model, we observe again that it is infeasible to find a plaintext which satisfies the condition in Item 2 of the encryption algorithm (resp., a ciphertext which satisfies the condition in Item 3 of the decryption algorithm). Thus, the modified ideal encryption scheme (in the Random Oracle Model) inherits all security features of the original scheme.

Similarly, to show that replacing the random oracle by any function ensemble yields an insecure scheme, we again observe that for any such ensemble there exists an adversary who – given the

[10] Similarly, we can adapt the argument to shared-key (aka private-key) encryption schemes. In this case no computational assumptions are needed since shared-key encryption is implied by one-way functions [3, 12, 8], and the later exist in the Random Oracle Model.

seed s – can generate a plaintext msg (resp., a ciphertext c) which satisfies the condition in Item 2 of the encryption algorithm (resp., the condition in Item 3 of the decryption algorithm). Hence, such an adversary can identify when msg is being encrypted (thus violates semantic security), or ask for a decryption of c, thus obtaining the secret decryption key. ■

Remark 5. As opposed to Theorem 8, here we need to make computational assumptions, namely, that there exist schemes which are secure in the Random Oracle Model. (The result in [13] imply that it is highly unlikely that such schemes are proven to exists without making any assumptions.) Clearly, any scheme which is secure without random oracles is also secure in the Random Oracle Model. Recall that the former exist, provided trapdoor permutations exist [9, 21].

5 Restricted correlation intractability

Faced with the negative result of Theorem 4, one may explore restricted (and yet possibly useful) versions of the correlation intractability property. One possibility is to put more stringent constraints on the use of the ensemble in a cryptographic scheme, and then to show that as long as the ensemble is only used in this restricted manner, it is guaranteed to maintain some aspects of correlation intractability.

In particular, notice that the proof of Theorem 4 relies heavily on the fact that the input to f_s can be as long as the seed s. This was used by "the adversary" to feed s as the input to the function f_s. Thus, one option would be to require that we only use f_s on inputs which are shorter than s. Specifically, we require that each function f_s will only be applied to inputs of length $\ell_{\text{in}}(|s|)$, where $\ell_{\text{in}} : \mathsf{N} \mapsto \mathsf{N}$ is some pre-specified function (e.g. $\ell_{\text{in}}(k) = k/2$). The corresponding restricted notion of correlation intractability is derived from Definition 3:

Definition 10 (restricted correlation intractability) *Let $\ell_{\text{in}}, \ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$ be length functions. A machine M is called ℓ_{in}-respectful if $|M(s)| = \ell_{\text{in}}(|s|)$ for all $s \in \{0,1\}^*$.*

A binary relation R is evasive with respect to $(\ell_{\text{in}}, \ell_{\text{out}})$ if for any ℓ_{in}-respectful polynomial time machine M

$$\Pr_{\mathcal{O}}[x \leftarrow M^{\mathcal{O}}(1^k),\ (x, \mathcal{O}(x)) \in R] \;=\; \text{negl}(k)$$

where $\mathcal{O} : \{0,1\}^{\ell_{\text{in}}(k)} \mapsto \{0,1\}^{\ell_{\text{out}}(k)}$ is a uniformly chosen function and $\text{negl}(\cdot)$ is a negligible function.

*We say that an ℓ_{out}-ensemble $\mathcal{F}$ is $(\ell_{\text{in}}, \ell_{\text{out}})$-*restricted correlation intractable *(or just ℓ_{in}-*correlation intractable*, for short), if for every ℓ_{in}-respectful polynomial time machine M and every evasive relation R w.r.t. $(\ell_{\text{in}}, \ell_{\text{out}})$, it holds that*

$$\Pr_{s \in \{0,1\}^k}[x \leftarrow M(s),\ (x, f_s(x)) \in R] = \text{negl}(k)$$

Weak ℓ_{in}-correlation intractability *is defined analogously by considering only polynomial-time recognizable R's.*

5.1 More negative results

The proof ideas of Theorem 4 can be easily applied to rule out the existence of certain restricted correlation intractable ensembles.

Proposition 11

(a) *If $\ell_{\rm in}(k) \geq k - O(\log k)$ for infinitely many k's, then there exists no ensemble which is $(\ell_{\rm in}, \ell_{\rm out})$-correlation intractable, even in the weak sense.*

(b) *If $\ell_{\rm in}(k) + \ell_{\rm out}(k) \geq k + \omega(\log k)$, there exists no ensemble which is $(\ell_{\rm in}, \ell_{\rm out})$-correlation intractable.*

Proof: The proof of (a) is a straightforward generalization of the proof of Theorem 4. Actually, we need to consider two cases: the case $\ell_{\rm in}(k) \geq k$ and the case $k - O(\log k) \leq \ell_{\rm in}(k) < k$. In the first case, we proceed as in the proof of Theorem 4 (except that we define $R^{\mathcal{F}} \stackrel{\rm def}{=} \{(x, f_s(x)) : s \in \{0,1\}^*, x = s0^{\ell_{\rm in}(|s|)-|s|}\}$). In the second case, for every ensemble $\mathcal{F}$, we define the relation

$$R^{\mathcal{F}} \stackrel{\rm def}{=} \{(x, f_{xz}(x)) : x, z \in \{0,1\}^*, |x| = \ell_{\rm in}(|xz|)\}$$

We show that $R^{\mathcal{F}}$ is evasive by showing that, for every $k \in \mathsf{N}$ and $x \in \{0,1\}^{\ell_{\rm in}(k)}$, there exist at most polynomially (in k) many y's such that $(x,y) \in R^{\mathcal{F}}$. This is the case since $(x,y) \in R^{\mathcal{F}}$ implies that there exists some z such that $\ell_{\rm in}(|xz|) = |x|$ and $y = f_{xz}(x)$. But using the case hypothesis we have $|x| = \ell_{\rm in}(|xz|) \geq |xz| - O(\log |xz|)$ which implies that $|z| = O(\log(|xz|))$ and hence also $|z| = O(\log |x|)$. Next, using the other case hypothesis (i.e., $k > \ell_{\rm in}(k) = |x|$), we conclude that $|z| = O(\log k)$. Therefore, there could be at most polynomially many such z's, and so the upper bound on the number of y's paired with x follows. The evasiveness of $R^{\mathcal{F}}$ as well as the assertion that $R^{\mathcal{F}}$ is polynomial-time computable follow (assuming that $\ell_{\rm in}$ is polynomial-time computable). On the other hand, consider the machine M which, on input s, outputs the $\ell_{\rm in}(|s|)$-bit prefix of s. Then, for every $s \in \{0,1\}^*$, we have $(M(s), f_s(M(s))) \in R^{\mathcal{F}}$.

For the proof of (b), assume that $\ell_{\rm in}(k) < k$ (for all but finitely many k's). We start by defining the "inverse" of the $\ell_{\rm in}$ function

$$\ell_{\rm in}^{-1}(n) \stackrel{\rm def}{=} \min\{k \ : \ \ell_{\rm in}(k) = n\}$$

(where, in case there exists no k such that $\ell_{\rm in}(k) = n$, we define $\ell_{\rm in}^{-1}(n) = 0$). By definition it follows that $k \geq \ell_{\rm in}^{-1}(\ell_{\rm in}(k))$, for all k's (because k belongs to the set $\{k' \ : \ \ell_{\rm in}(k') = \ell_{\rm in}(k)\}$), and that $\ell_{\rm in}(\ell_{\rm in}^{-1}(n)) = n$, whenever there exists some k for which $n = \ell_{\rm in}(k)$. Next we define

$$R^{\mathcal{F}} \stackrel{\rm def}{=} \{(x, f_{xz}(x)) \ : \ x, z \in \{0,1\}^*, |x| + |z| = \ell_{\rm in}^{-1}(|x|)\}$$

This relation is well defined since, by the conditions on the lengths of x and z, we have $\ell_{\rm in}(|xz|) = \ell_{\rm in}(\ell_{\rm in}^{-1}(|x|)) = |x|$ and so the function f_{xz} is indeed defined on the input x. In case $\ell_{\rm in}(k) \leq k - \omega(\log k)$, this relation may not be polynomial-time recognizable. Still, it is evasive w.r.t. $(\ell_{\rm in}, \ell_{\rm out})$, since with security parameter k we have for every $x \in \{0,1\}^{\ell_{\rm in}(k)}$

$$\begin{aligned}
\left|\left\{y \in \{0,1\}^{\ell_{\rm out}(k)} \ : (x,y) \in R^{\mathcal{F}}\right\}\right| &= \left|\{f_{xz}(x) \ : |z| = \ell_{\rm in}^{-1}(\ell_{\rm in}(k)) - \ell_{\rm in}(k)\} \ \cap \ \{0,1\}^{\ell_{\rm out}(k)}\right| \\
&\leq 2^{\ell_{\rm in}^{-1}(\ell_{\rm in}(k)) - \ell_{\rm in}(k)} \\
&\leq 2^{k - \ell_{\rm in}(k)}
\end{aligned}$$

Using $k - \ell_{\rm in}(k) \leq \ell_{\rm out}(k) - \omega(\log k)$, we conclude that the set of y's paired with x forms a negligible fraction of $\{0,1\}^{\ell_{\rm out}(k)}$, and so that $R^{\mathcal{F}}$ is evasive. Again, the machine M, which on input s outputs the $\ell_{\rm in}(|s|)$-bit prefix of s, satisfies $(M(s), f_s(M(s))) \in R^{\mathcal{F}}$, for all s's. ■

Open Problems: Proposition 11 still leaves open the question of existence of $(\ell_{\text{in}}, \ell_{\text{out}})$-restricted correlation intractable ensembles, for the case $\ell_{\text{in}}(k) + \ell_{\text{out}}(k) < k + O(\log k)$.[11] We believe that it is interesting to resolve the situation either way: Either provide negative results also for the above special case, or provide a plausible construction. Also open is the sub-case where $\ell_{\text{in}}(k) + \ell_{\text{out}}(k) = k + \omega(\log k)$ but one considers only *weak* $(\ell_{\text{in}}, \ell_{\text{out}})$-restricted correlation intractability. (Recall that Case (b) of Proposition 11 is proven using relations which are not known to be polynomial-time recognizable.)

We comment that even if restricted correlation intractable ensembles exist, then they are very non-robust constructs. For example, even if the ensemble $\mathcal{F} = \{f_s : |s| = k\}_k$ is correlation intractable with respect to some length functions $(\ell_{\text{in}}, \ell_{\text{out}})$, the ensemble which is obtained by applying many independent copies of $\mathcal{F}$ and concatenating the results may not be. That is, for $m : \mathsf{N} \mapsto \mathsf{N}$, define

$$\mathcal{F}^m \stackrel{\text{def}}{=} \{f'_{s_1\|\cdots\|s_{m(k)}} : |s_1| = \cdots = |s_{m(k)}| = k\}_k\,, \tag{2}$$

where, for $x_1\|\cdots\|x_{m(k)} \in \{0,1\}^{m(k)\cdot\ell_{\text{in}}(k)}$,

$$f'_{s_1\|\cdots\|s_{m(k)}}(x_1\|\cdots\|x_{m(k)}) \stackrel{\text{def}}{=} f_{s_1}(x_1)\|\cdots\|f_{s_{m(k)}}(x_{m(k)})\,. \tag{3}$$

Then, for sufficiently large m (e.g., $m(k) \geq k/\ell_{\text{in}}(k)$ will do), the "direct product" ensemble $\mathcal{F}^m$ is not correlation intractable (not even in the natural restricted sense). That is,

Proposition 12 *Let $\ell_{\text{in}}, \ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$ be length functions, and let $m : \mathsf{N} \mapsto \mathsf{N}$ be another function so that $m(k) \geq k/\ell_{\text{in}}(k)$. Let $\mathcal{F}$ be an arbitrary function ensemble, and $\mathcal{F}^m$ be as defined in Eq. (2) and (3). Then, $\mathcal{F}^m$ is not correlation intractable, not even in the natural $(\ell^m_{\text{in}}, \ell^m_{\text{out}})$-restricted sense, where $\ell^m_{\text{xx}}(m(k)\cdot k) \stackrel{\text{def}}{=} m(k)\cdot\ell_{\text{xx}}(k)$, for $\text{xx} \in \{\text{in}, \text{out}\}$.*

Proof: We assume, for simplicity that $m(k) = k/\ell_{\text{in}}(k)$. Given $\mathcal{F}^m$ as stated, we again adapt the proof of Theorem 4. This time, we define the relation

$$R^{\mathcal{F}^m} \stackrel{\text{def}}{=} \bigcup_k \{\ (s, f_s(s')\|t)\ :\ |s| = k,\quad s' \text{is the } \ell_{\text{in}}(k)\text{-prefix of } s,\quad |t| = (m(k)-1)\cdot\ell_{\text{out}}(k)\ \}$$

Notice that in this definition we have $|s| = \frac{k}{\ell_{\text{in}}(k)}\cdot\ell_{\text{in}}(k) = m(k)\cdot\ell_{\text{in}}(k) = \ell^m_{\text{in}}(m(k)\cdot k)$, and also $|f_s(s')| + |t| = m(k)\cdot\ell_{\text{out}}(k) = \ell^m_{\text{out}}(m(k)\cdot k)$, so this relation is indeed $(\ell^m_{\text{in}}, \ell^m_{\text{out}})$-restricted.

Again, it is easy to see that $R^{\mathcal{F}}$ is polynomial-time recognizable, and it is evasive since every string $x \in \{0,1\}^k$ is coupled with at most a $2^{-\ell_{\text{out}}(k)}$ fraction of the possible $(m(k)\cdot\ell_{\text{out}}(k))$-bit long strings, and $\ell_{\text{out}}(k) = \omega(\log k) = \omega(\log(m(k)\cdot k))$.

On the other hand, consider a (real-life) adversary that given the seed $s = s_1\|\cdots\|s_{m(k)} \in \{0,1\}^{m(k)\cdot k}$ for the function $f'_{s_1\|\cdots\|s_{m(k)}}$, sets the input to this function to be equal to s_1. If we denote the $\ell_{\text{in}}(k)$-prefix of s_1 (equiv., of s) by s'_1 then $f_{s_1}(s'_1)$ is a prefix of $f'_{s_1\|\cdots\|s_{m(k)}}(s_1)$ and so $(s_1, f'_{s_1\|\cdots\|s_{m(k)}}(s_1)) \in R^{\mathcal{F}}$. Thus, this real-life adversary violates the (restricted) correlation intractability of $\mathcal{F}^m$. ■

5.2 Generalized notion of correlation intractability

Recall that Proposition 11 does not rule out the existence of restricted ensembles having seeds which are longer than the sum of lengths of their inputs and outputs. However, even for this

[11] In fact such ensembles do exist in case $k \geq 2^{\ell_{\text{in}}(k)}\cdot\ell_{\text{out}}(k)$ (as the seed may be used to directly specify all the function's values), but we dismiss this trivial and useless case below.

special case the only thing which is not ruled out is a *narrow definition* which refers to forming rare relationships between a *single* input-output pair. Furthermore, if one generalizes the definition of correlation intractability so as to consider evasive relations over unbounded sequences of inputs and outputs, then the negative result in Proposion 11 can be extended for arbitrary ℓ_{in} and ℓ_{out}. That is,

Definition 13 (generalized restricted correlation intractability) *Let $\ell_{\text{in}}, \ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$ be length functions.*

We consider probabilistic polynomial-time oracle machines which on input 1^k have oracle access to a function $\mathcal{O} : \{0,1\}^{\ell_{\text{in}}(k)} \mapsto \{0,1\}^{\ell_{\text{out}}(k)}$. A relation R over pairs of binary sequences is evasive with respect to $(\ell_{\text{in}}, \ell_{\text{out}})$ (or $(\ell_{\text{in}}, \ell_{\text{out}})$-evasive) *if for any polynomial-time machine M as above it holds that*

$$\Pr_{\mathcal{O}}\left[(x_1,...,x_m) \leftarrow M^{\mathcal{O}}(1^k)\,;\quad \begin{array}{l}|x_1| = \ldots = |x_m| = \ell_{\text{in}}(k)\\ \text{and } ((x_1,...,x_m),(\mathcal{O}(x_1),...,\mathcal{O}(x_m)) \in R\end{array}\right] \;=\; \text{negl}(k)$$

As usual, $\mathcal{O} : \{0,1\}^{\ell_{\text{in}}(k)} \mapsto \{0,1\}^{\ell_{\text{out}}(k)}$ is a uniformly chosen function.

We say that an ℓ_{out}-ensemble $\mathcal{F}$ is $(\ell_{\text{in}}, \ell_{\text{out}})$-restricted general correlation intractable *(or just ℓ_{in}-*general correlation intractable*, for short), if for every $(\ell_{\text{in}}, \ell_{\text{out}})$-evasive relation R and every polynomial time oracle machine M it holds that*

$$\Pr_{s}\left[(x_1,...,x_m) \leftarrow M^{\mathcal{O}}(s)\,;\quad \begin{array}{l}|x_1| = \ldots = |x_m| = \ell_{\text{in}}(k)\\ \text{and } ((x_1,...,x_m),(f_s(x_1),...,f_s(x_m)) \in R\end{array}\right] \;=\; \text{negl}(k)$$

Proposition 14 *Let $\ell_{\text{in}}, \ell_{\text{out}} : \mathsf{N} \mapsto \mathsf{N}$ be arbitrary length functions, with $\ell_{\text{out}}(k) = \omega(\log k)$. Then there exist no $(\ell_{\text{in}}, \ell_{\text{out}})$-restricted* general *correlation intractable function ensembles.*

Proof: Below we assume for simplicity that $k/\ell_{\text{in}}(k)$ is always an integer, and denote $m(k) = k/\ell_{\text{in}}(k)$. Let $\mathcal{F}$ be an ℓ_{out}-ensemble. Adapting the proof of Theorem 4, we define the relation

$$R^{\mathcal{F}} \stackrel{\text{def}}{=} \bigcup_k \left\{((x_1,...,x_{m(k)}),(f_s(x_1),...,f_s(x_{m(k)}))) \;:\; \begin{array}{l}|x_1| = \ldots = |x_{m(k)}| = \ell_{\text{in}}(k)\\ \text{and } s = x_1 \cdots x_{m(k)}\end{array}\right\}$$

We note that this relation is polynomial-time recognizable. Clearly, it is evasive, since every sequence of x_i's is coupled with at most one sequence. However, as before the identity function I demonstrates that the corresponding general restricted correlation intractability condition does not hold: For any $s \in \{0,1\}^k$, parsing $s = x_1,...,x_{m(k)}$ where each x_i is of length $\ell_{\text{in}}(k)$, we have $((x_1,...,x_{m(k)}),(f_s(x_1),...,f_s(x_{m(k)}))) \in R^{\mathcal{F}}$. ■

5.3 Discussion

Propositions 11, 12 and 14 demonstrate that there is only a very narrow margin in which strict correlation-intractability may be used. Still, even ensembles which are (strict) correlation-intractable with respect to relations of a-priori bounded total length (of input-output sequences) may be useful in some applications. Typically, this may hold in applications where number of invocations of the cryptosystem is a-priori bounded (or where the security of the system depends only on an a-priori bounded partial history of invocations; e.g., the current one). We note that the Fiat-Shamir heuristic (for transforming interactive identification protocols into signature schemes [7]) does not fall into the above category, since the function's seed needs to be fixed with the public key, and used for signing polynomially many messages, where the polynomial is not a-priori known.

Acknowledgments

We wish to thank Silvio Micali for enlightening discussions.

References

[1] M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In *1st Conf. on Computer and Communications Security*, ACM, pages 62–73, 1993.

[2] M. Bellare and P. Rogaway. The Exact Security of Digital Signatures: How to Sign with RSA and Rabin. In *EuroCrypt96*, Springer LNCS (Vol. 1070), pages 399–416.

[3] M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. *SICOMP*, Vol. 13, pages 850–864, 1984. Preliminary version in *23rd FOCS*, 1982.

[4] R. Canetti. Towards Realizing Random Oracles: Hash Functions that Hide All Partial Information. In *Crypto97*, Springer LNCS (Vol. 1294), pages 455–469.

[5] R. Canetti, D. Micciancio and O. Reingold. Perfectly One-Way Probabilistic Hashing. *30th ACM Symposium on the Theory of Computing*, 1998.

[6] I.B. Damgård. Collision free hash functions and public key signature schemes. In *EuroCrypt87*, LNCS (Vol. 304), Springer-Verlag, 1988. Pages 203–216.

[7] A. Fiat and A. Shamir. How to Prove Yourself. Practical Solutions to Identification and Signature Problems. In *Crypto86*, Springer-Verlag LNCS (Vol. 263), pages 186–189, 1987.

[8] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *JACM*, Vol. 33, No. 4, pages 792–807, 1986.

[9] S. Goldwasser and S. Micali. Probabilistic Encryption. *JCSS*, Vol. 28, No. 2, pages 270–299, 1984. Preliminary version in *14th STOC*, 1982.

[10] S. Goldwasser, S. Micali, and R.L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SICOMP*, April 1988, pages 281–308.

[11] L.C. Guillou and J.J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessors Minimizing both Transmission and Memory. In *EuroCrypt88*, Springer-Verlag LNCS (Vol. 330), pages 123–128.

[12] J. Håstad, R. Impagliazzo, L.A. Levin and M. Luby. Construction of Pseudorandom Generator from any One-Way Function. To appear in *SICOMP*. Preliminary versions by Impagliazzo et. al. in *21st STOC* (1989) and Håstad in *22nd STOC* (1990).

[13] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. *21th ACM Symposium on the Theory of Computing*, 1989.

[14] J. Kilian. A Note on Efficient Zero-Knowledge Proofs and Arguments. In *24th ACM Symposium on the Theory of Computing*, pages 723–732, 1992.

[15] S. Micali. CS Proofs. In *35th IEEE Symposium on Foundations of Computer Science*, pages 436–453, 1994.

[16] M. Naor and M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *21th ACM Symposium on the Theory of Computing*, pages 33-43, 1989.

[17] T. Okamoto Provably Secure and Practical Identification Scheme and Corresponding Signature Scheme. In *Crypto91*, pages 31–53.

[18] D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. In *EuroCrypt96*, Springer-Verlag LNCS (Vol. 1070), pages 387–398.

[19] J. Rompel. One-Way Functions are Necessary and Sufficient for Secure Signatures. In *22nd ACM Symposium on the Theory of Computing*, pages 387–394, 1990.

[20] C.P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, Vol. 4, No. 3, pages 161–174 (1991).

[21] A.C. Yao. Theory and Application of Trapdoor Functions. In *23rd IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.

Appendix : CS-Proofs

Following is the formulation of CS-proofs, as defined in [15].
A Computationally Sound (CS) proof system is a non-interactive system for proving statements of the type *machine M accepts input x within t steps.* It consists of two polynomial-time oracle machines, a Prover $\textsc{Prv}$ and a Verifier $\textsc{Ver}$. On security parameter k, the prover has input $(1^k, \langle M\rangle, x, 1^t)$ and access to an oracle $\mathcal{O}$, and it computes a proof $\pi = \textsc{Prv}^{\mathcal{O}}(1^k, \langle M\rangle, x, 1^t)$. The verifier has input $(1^k, \langle M\rangle, x, t, \pi)$, with t encoded in binary, and access to $\mathcal{O}$, and it decides whether to accept or reject the proof. The proof system satisfies the follwing conditions

Efficiency conditions: An important aspect of Micali's definition (and construction) is that the prover satisfies a stronger condition than just being polynomial-time in the worst case. Specifically, if it is given an input $(\langle M\rangle, x, 1^t)$ such that M accepts x within time $t' < t$, then on that input it works in time which is polynomial in t' (rather than in the upper bound t).

Perfect completeness: For any M, x, t such that machine M accepts the string x within t steps, and for any k,

$$\Pr_{\mathcal{O}}\left[\begin{array}{l}\pi \leftarrow \textsc{Prv}^{\mathcal{O}}(1^k, \langle M\rangle, x, 1^t),\\ \textsc{Ver}^{\mathcal{O}}(1^k, \langle M\rangle, x, t, \pi) = \texttt{accept}\end{array}\right] = 1$$

(In our context it is important that this probability is 1, so that we are guaranteed that a proof for a true statement will always be convincing, even when the random oracle is replaced by some fixed function.)

Computational soundness: For any polynomial time oracle machine $\textsc{Bad}$ and any input $w = (\langle M\rangle, x, 1^t)$ such that M *does not* accepts x within t steps, it holds that

$$\Pr_{\mathcal{O}}\left[\begin{array}{l}\pi \leftarrow \textsc{Bad}^{\mathcal{O}}(1^k, \langle M\rangle, x, 1^t),\\ \textsc{Ver}^{\mathcal{O}}(1^k, \langle M\rangle, x, t, \pi) = \texttt{accept}\end{array}\right] \le \frac{poly(k + |w|)}{2^k}$$

Remark 6. The soundness condition that we use in the proof of Theorem 8 is that, given the machine M (and the complexity bound $t(\cdot)$), it is hard to find *any pair* (x, π) such that M does not accept x within t steps and yet $\textsc{Ver}$ will accept π as a valid CS-proof to the contrary. The reason that we can use this (seemingly stronger) soundness condition, is that on security parameter k and a string x of length n, we always run $\textsc{Ver}$ with security parameter $n + k$. Hence, even if the bad prover can pick an n-bit x depending on the oracle $\mathcal{O}$, the probability that it will be able to fool the verifier is still at most $\text{poly}(k)/2^k$.