Towards a Theory of Extractable Functions

Ran Canetti[1]⋆ and Ronny Ramzi Dakdouk[2]⋆⋆

[1] Tel Aviv University, Tel Aviv, Israel.
canetti@tau.ac.il
[2] Yale University, New Haven, CT.
dakdouk@cs.yale.edu

Abstract. Extractable functions are functions where any adversary that outputs a point in the range of the function is guaranteed to "know" a corresponding preimage. Here, knowledge is captured by the existence of an efficient extractor that recovers the preimage from *the internal state of the adversary*. Extractability of functions was defined by the authors (ICALP'08) in the context of perfectly one-way functions. It can be regarded as an abstraction from specific knowledge assumptions, such as the Knowledge of Exponent assumption (Hada and Tanaka, Crypto 1998).

We initiate a more general study of extractable functions. We explore two different approaches. The first approach is aimed at understanding the concept of extractability in of itself; in particular we demonstrate that a weak notion of extraction implies a strong one, and make rigorous the intuition that extraction and obfuscation are complementary notions.

In the second approach, we study the possibility of constructing cryptographic primitives from simpler or weaker ones while maintaining extractability. Results are generally positive. Specifically, we show that several cryptographic reductions are either "knowledge-preserving" or can be modified to be so. Examples include reductions from extractable weak one-way functions to extractable strong ones, from extractable pseudorandom generators to extractable pseudorandom functions, and from extractable one-way functions to extractable commitments. Other questions, such as constructing extractable pseudorandom generators from extractable one way functions, remain open.

1 Introduction

Extractability plays a central role in cryptographic protocol design and analysis. In its basic form, it relates to two-party protocols where one of the parties (a "prover") has secret input, and tries to convince the other party (a "verifier") that it holds the secret. The idea is to argue that if the verifier accepts the interaction, then the prover indeed "knows" the secret. More concretely, extractability makes the following requirement: Given access to the internals of *any* (potentially malicious) prover, it is possible to explicitly and efficiently compute the secret value as long as the verifier accepts an interaction. (Many variants of this notion exist, of course. See e.g. [12].)

⋆ Supported by NSF grant CFF-0635297 and US-Israel Binational Science Foundation Grant 2006317, a European Union Marie Curie grant, and the Check Point Institute for Information Security.

⋆⋆ Supported by NSF grant #0331548.

The notion of **extractable functions** extends the concept of extractability to the more basic setting of computing a function. Here the task of "convincing a verifier" is replaced by "outputting a value in the range of the function". More specifically, any machine that generates a point in the range "knows" a corresponding preimage in the sense that a preimage is efficiently recoverable given the internal state of the machine.

Extractable functions were coined in [8] for the specific goal of defining extractable perfectly one-way (EPOW) functions.[3] These functions were demonstrated to have some interesting applications, such as new ways to realize Random Oracles and new three-round Zero-Knowledge arguments based on weaker assumptions than previously known. Furthermore, it was demonstrated that extractable functions can be viewed as an abstraction from specific knowledge assumptions, such as the Knowledge of Exponent (KE) assumption [16, 3] or the Proof of Knowledge (POK) assumption [20], in much the same way as the notion of one-way function is an abstraction of the Discrete Log (DL) assumption.

This work attempts to initiate a more general study of extractable functions. Specifically, we address two goals: First, we try to understand exactly what extraction means and how different notions of extraction (and lack of it) are related. Second, we study the possibility of constructing complex primitives from simpler ones while preserving extractability. We note that the latter approach may help in basing cryptographic protocols that use or require specific knowledge assumptions, on a general computational notion, which in turn may be concretely realized by alternative assumptions.

Before discussing this work in more detail, we provide a high level overview of the two versions of knowledge extraction defined in [8]: interactive and noninteractive extraction.

Noninteractive extraction. Noninteractive extraction is an abstraction of specific knowledge assumptions as mentioned in the previous paragraph. Informally, there is a family of functions and the adversary gets a description of a specific function from the family, and tries to output a point in the range of this function. This function family is considered noninteractively extractable if whenever the adversary generates a point in the range, it knows a corresponding preimage. In other words, for every such adversary there is a corresponding extractor that computes a preimage from the private input of the adversary. One extreme example of extractable functions is the identity function where the output itself reveals the input. Obviously, such functions are of lesser interest to cryptographic applications than functions with computational hardness properties. On another extreme, if the function is a one-way permutation, then it is easy to output a valid image without knowing a preimage; specifically, output a random string in the range. In this work, we concentrate on functions that enjoy both properties, namely, extractability and computational hardness.

Unlike proofs of knowledge [15, 2], this notion of extraction does not *require* efficient verification. In other words, the range of the function is not necessarily efficiently verifiable. Therefore, it may not be possible to decide if the adversary generates a point in the range (and consequently, knows a preimage). However, this notion guarantees the implication: If the adversary generates an image, it knows a preimage. We mention that the construction in [8] has a range that is efficiently verifiable in the presence of some auxiliary information (about the function itself).

[3] Informally, a probabilistic function is perfectly one-way if it hides all partial information about the input [7].

Extraction can be studied with or without auxiliary information. We would like to consider extraction in the presence of auxiliary information as this is a more useful and meaningful notion. Auxiliary information can be either dependent or independent [14] (here, the dependence is on the specific function under study). We remark that dependent auxiliary information is inseparable from independent auxiliary information when extraction is required for a single function, f. This is so because it is not possible to prevent an adversary with access to auxiliary information from receiving dependent auxiliary information, e.g., $f(x)$. Moreover, the notion of a single extractable function with auxiliary information is not realizable for one-way functions. Specifically, by the one-wayness assumption, there is no extractor for the adversary that receives $f(x)$, for a uniform x, and simply outputs it. Consequently, we relax the requirement to extraction for a family of functions, i.e., a function is chosen uniformly from the family. Indeed, the KE assumption is already formulated in terms of function families.

In this work, we focus on extraction with independent auxiliary information only. Formulating and realizing extraction with dependent auxiliary information is tricky. For instance, it is possible that $f(x)$ is hidden in the input in some clever way such that it is easy to recover $f(x)$ but not x. For example, the input of the adversary may look like $(r_1, u_1), \ldots, (r_n, u_n)$, where the ith bit of $f(x)$ is $\langle r_i, u_i \rangle$. [8] addresses this issue by restricting the dependency of auxiliary information so that only a sequence of images under f can be part of dependent auxiliary information. Moreover, Zheng and Seberry [25] follow the same approach under the notion of "sole-samplability".

Interactive extraction. This notion is geared towards probabilistic functions, and can work for single functions as well as families. In interactive extraction, the adversary engages in a 3-round game with a challenger. The objective of the game is to show that the adversary is capable of computing a function, f, on some point, x, that he chooses, but using random coins for f that the challenger chooses. In other words, the goal is to show that the adversary is capable of computing a "large" fraction of the possible images of x under f (recall f is probabilistic). In more detail, the adversary, A, sends, in the first round, a point, $y_0 = f(x, r_0)$, where x and r_0 are chosen by A. The challenger responds with random coins, r_1, in the second round and A has to send back $y_1 = f(x, r_1)$. In this setting, consistency means that y_0 and y_1 have a common preimage x. Interactive extraction means if the adversary is able to answer consistently, then it knows a common preimage. As in the noninteractive case, this form of knowledge is captured computationally by the existence of an extractor that recovers a preimage from the private input of the adversary. We emphasize that no verification of consistency is assumed to occur. The knowledge requirement states that *if the adversary is consistent*, it must know a preimage.

Unlike noninteractive extraction, interactive extraction is required to work for any function. In other words, the function is fixed once and for all, and any auxiliary information is allowed to depend on this function. Intuitively, this is realizable because the challenge in the second round forces the adversary to compute an image "online".

In interactive extraction, we focus mainly on *probabilistic* functions because for deterministic functions, this notion is equivalent to noninteractive extraction. (To use the 3-round game of interactive extraction on a deterministic function, f, view f as a probabilistic function that simply ignores the random coins, i.e. $f(x, r) = f(x)$ for any x and r.)

It is worth mentioning that noninteractive extraction can be viewed as a two-round interactive extraction analogous to the three-round extraction discussed above.

Specifically, in the first round the challenger sends a random function from the family and the adversary responds with a point in the range of this function. That is, there is a fixed function, g, the challenger sends a random r, and the adversary responds with $g(x, r) = f_r(x)$.

1.1 Our work

We approach extractable functions from two different angles.
First, we attempt to address the question: What makes a function extractable? Moreover, if a function is extractable with noticeable success, does this mean that it is extractable in a strong sense? Towards answering these questions, we show that every function satisfies either a "mild" form of obfuscation [1] or a "mild" form of extraction. In other words, lack of extractability can be viewed as inability to "reverse engineering" or obfuscatability. This is indeed what one might naïvely expect - a function is either extractable or obfuscatable, and we show that this naïve thinking is correct to some extent. We then address the second question posed at the beginning of this paragraph. We find out that for a large class of functions, notably, POW functions with auxiliary information, the answer to this question is positive.
Second, we try to construct complex extractable primitives from simpler ones. In general, extractable functions exist, e.g., the identity function. However, extractable functions are more useful in cryptographic applications if they satisfy certain hardness assumptions. Thus, in the second line of work, we address the question: Is it possible to build primitives with complex hardness properties from weaker hardness assumptions while maintaining extractable properties? For instance, suppose we have an extractable weak one-way function, can we build an extractable strong one-way function? Results indicate that answers to such questions are mostly positive.

On the first line of work. We discuss interactive extraction before noninteractive extraction.

On interactive extraction versus obfuscation. This line of work starts with an observation that extraction and obfuscation complement each other in a natural way. In other words, if a function is not extractable, then this lack of extractability is some form of obfuscation. Specifically, we call a function weakly (and interactively) extractable if for any adversary that is consistent in the interactive game with noticeable probability, there is a corresponding extractor that recovers a preimage with noticeable success. Moreover, the obfuscation mentioned previously relates to inability to "reverse engineer" an obfuscated program that produces images under the function. In other words, there is an obfuscated code that receives r as input and computes $f(x, r)$ for some x "hidden" in the obfuscated code. In more detail, we call f weakly obfuscatable if the following holds. There is an obfuscator that produces a program capable of correctly computing the function $f_x(r) = f(x, r)$ with noticeable probability, where x is chosen according to some well-spread distribution and then "hidden" in the program. Also, the program is considered obfuscated in the sense that it is hard to recover x from the obfuscated program, when x is drawn from the well-spread distribution mentioned above. The corresponding theorem can be stated in words as:
Theorem 1: Every family of probabilistic functions is either weakly extractable or weakly obfuscatable.

We emphasize that Theorem 1 is a general observation on any family of functions and does not assume anything about the family, not even that it is efficiently computable. Informally, this theorem can be argued for as follows. Suppose a function, f, is not weakly extractable. Then, there is an adversary A that answers consistently in the 3-round game of interactive extraction, and yet there is no extractor that recovers a preimage x. We use A to construct an obfuscation for the function f_x. The obfuscation simply contains the description of A and a corresponding private input that causes A to answer consistently. To compute $f_x(r)$, simulate A, send r in the second round of the extraction game, and output the response of A. Functionality of this obfuscation follows from consistency of A while the hiding property follows directly from the assumption that no extractor is able to recover x. We point out that finding an obfuscation of f_x may not be efficient, however, the obfuscation itself is efficient because A is.

Amplifying knowledge extraction. Theorem 1 is not entirely satisfactory because extraction is guaranteed to occur only noticeably often. So, we address the issue of amplifying extraction. We show how to do so under a necessary (for the class of injective functions) and sufficient assumption on the function. Specifically, we assume what we call “weak verification”. Weak verification is a notion introduced to show that some form of verification is necessary and sufficient for knowledge amplification. Moreover, it is implied by common verification notions such as public verification for probabilistic functions [7]. Informally, weak verification means for any adversary A that outputs images in the range of f, there is a corresponding verifier, V, which given some x and the private input of A, decides whether the output of A is a valid image of x under f. In other words, V has to decide whether there exists an r such that $f(x,r) = A(z, r_A)$, where z and r_A are the auxiliary information and random coins for A. Moreover, V is allowed to fail with some arbitrary small, yet noticeable probability. We use the term “extraction (respectively, verification) with vanishing but noticeable error” to mean that for every polynomial, p, there is an extractor (respectively, verifier) that fails no more than $\frac{1}{p}$ fraction of the time. The corresponding theorem can be stated in words as follows.

Theorem 2: Every weakly-verifiable family of probabilistic functions is either weakly obfuscatable or extractable with vanishing but noticeable error. Moreover, if an injective family of functions is extractable with vanishing but noticeable error, then it is weakly verifiable.

At a very high level, the proof of Theorem 2 uses a variant of Impagliazzo’s hard-core lemma [19] to amplify weak extraction to extraction with vanishing but noticeable error. Informally, we use the lemma to construct a family, $\mathbb{U}$, of machines that take the input of A and attempt to extract a preimage, x, from it. This family has the property that when all its members fail, no machine can succeed *noticeably*. We then construct a family of distributions on the input of A, one distribution for each input length n, such that any member of $\mathbb{U}$ succeeds only negligibly often (as n increases). Consequently, if $\mathbb{U}$ is not a family of extractors with vanishing but noticeable error, then the distributions just mentioned have a noticeable weight in proportion to the original one. Using Theorem 1 on A and the new distributions imply the existence of an extractor with noticeable success. However, this contradicts the amplification lemma.

Interactively-extractable POW functions. An important corollary to Theorem 2 is that every POW function with auxiliary information is interactively extractable (see Corollary 2 for a more formal presentation). This supersedes the corresponding transformation of [8] from POW with auxiliary information to extractable POW function. Moreover, the current result is more efficient in that the challenger needs to send a single challenge instead of n.

Towards negligible error. We can obtain negligible failure probability if we relax the notion of extraction so that it applies only to "reliably-consistent adversaries". Intuitively, an adversary is reliably consistent if its consistency is noticeable. In other words, disregarding input on which the adversary is consistent only negligibly often, there is a fixed polynomial, p, such that $\frac{1}{p}$ is a lower bound on the probability of consistency (here, the probability is taken over the random challenge). The corresponding theorem can be stated as follows:
Theorem 3: Every weakly-verifiable family of probabilistic functions is either weakly obfuscatable or extractable with negligible error for adversaries that are reliably consistent.
Moreover, if an efficiently computable and verifiable family of functions is extractable with negligible error, then every corresponding adversary is reliably consistent.
The proof this theorem is very similar to the previous one but it uses a stronger amplification lemma in the uniform model. Informally, the lemma states that there is a family of polynomial-time machine, $\mathbb{U}$, such that no machine can succeed in inverting a function where all members of $\mathbb{U}$ fail. (Contrast this lemma with the previous one, where the guarantee is that no machine can succeed *noticeably* where $\mathbb{U}$ fails.)

On noninteractive extraction versus obfuscation. Results similar to those for interactive extraction hold in this case. However, they are weaker in the sense that functions seem to be more likely to satisfy a weaker notion of obfuscation. Informally, the obfuscated program receives a function description, k, as input and outputs $f_k(x)$ for some x hidden in the program that may depend on k. Moreover, it is hard to recover x from the obfuscated code. The results and proofs are similar. Two issues are worth highlighting. First, following the discussion at the beginning of this introduction, the function is not fixed in advance. Rather, it is sampled from a well-spread distribution and given to the adversary. Second, a corollary to these results states that injective functions that are extractable with vanishing but noticeable error are extractable with negligible error.

On the second line of research: Constructing extractable functions.

Taking another approach towards a theory of extractable functions, we study knowledge-preserving reductions among cryptographic primitives. In other words, we address the question: given a noninteractively extractable cryptographic primitive, is it possible to construct another primitive while maintaining extraction? We attempt to answer this question by reviewing the literature on cryptographic reductions and investigating whether these reductions maintain extraction. Here, we focus solely on noninteractive extraction because deterministic one-way functions are not interactively extractable (Corollary 1). The results are positive: Most reductions maintain extractability or can be modified to do so. The following is a list of reductions that preserve extractability.

1. *Extractable weak one-way functions* $\Longrightarrow$ *extractable strong one-way functions.* (This is the standard reduction [24, 12].)
2. *Extractable pseudorandom generators* $\Longrightarrow$ *extractable pseudorandom functions.* This reduction uses the construction of [13]. We assume, in addition to the extractable pseudorandom generator, G_1, another pseudorandom

generator, G_2 that is not necessarily extractable but remains pseudorandom in the presence of G_1, i.e., $G_1(x), G_2(x)$ is pseudorandom when x is uniform.

3. *Extractable one-way functions* $\implies$ *extractable* $1-1$ *trapdoor functions.* This construction assumes, in addition, the existence of a $1-1$ trapdoor function that remains one-way in the presence of the extractable function.
4. *Extractable one-way functions* $\implies$ *extractable public-key encryption.* This reduction, assumes, in addition, a trapdoor permutation. Here, extractable public-key encryption is against passive adversaries and it means that it is hard to generate a ciphertext without knowledge of the plaintext and *without seeing another ciphertext*. On the other hand, extractability against active adversaries, that is adversaries that can see other ciphertext is known in the literature as plaintext-aware encryption [5, 18, 4, 11]. We mention that this notion requires extraction with dependent auxiliary information and is left for future work.
5. *Extractable one-way functions* $\implies$ *extractable* 2*-round commitments.* Extractable commitments means if the sender commits correctly (i.e., the commitment can be opened) then it knows the message at the commit stage. This reduction uses either the construction of [6] or of [21]. We note that [23] independently constructs extractable 2-round commitments from plaintext-aware encryption.

The main reduction missing from this list is from one-way functions to pseudorandom generators. Even though we give a reduction from the KE and DDH assumptions to extractable pseudorandom generators, constructing such generators from extractable one-way functions remains open. In this work, we take a step towards this goal by giving a reduction from a "strongly" extractable one-way function, where extraction is required to hold even when $f(x)$ is represented unambiguously in a different way. Refer to Section 4 for a detailed presentation of all results regarding knowledge-preserving reductions.

Organization. We present the first approach in the context of interactive extraction in Section 3 (the corresponding results on noninteractive extraction can be found in the full version of the paper), and the second line of research in Section 4. Formal definitions of extractable functions appear in Section 2. Due to space limitation, formal proofs appear only in the full version of the paper.

2 Preliminaries

We define here interactive and noninteractive extraction. Note that these definitions require negligible extraction error. In Section 3, we study weaker forms of extraction, where the extractor succeeds noticeably or fails with vanishing but noticeable probability.

Definition 1 (Noninteractive extraction). *A randomized family ensemble, $\mathbb{F} = \{\{F_k\}_{k \in K_n}\}_{n \in \mathbb{N}}$, is called **noninteractively extractable** if for any PPT A, any well-spread distribution, K_n, on the function description, any distribution, $\mathbb{ZR} = \{ZR_n\}_{n \in \mathbb{N}}$, on auxiliary information and the private input of A, there is polynomial-time machines, $\mathcal{K}$, such that:*

$$Pr[(z, r_A) \leftarrow ZR_n,\ k \leftarrow K_n,\ y = A(k, z, r_A),\ x = \mathcal{K}(k, z, r_A) :$$
$$\exists r, f_k(x, r) = y \text{ or } \forall x', r', y \neq f_k(x', r')] > 1 - \mu(n).$$

Definition 2 (Interactive Extraction). *A randomized family ensemble, $\mathbb{F} = \{\{F_k\}_{k\in K_n}\}_{n\in\mathbb{N}}$, is called **interactively extractable** if for any PPT A, any distribution, $\mathbb{ZR} = \{ZR_n\}_{n\in\mathbb{N}}$, on auxiliary information and the private input of A, there is polynomial-time machines, $\mathcal{K}$, such that for any $k \in K_n$:*

$$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1),\ x = \mathcal{K}(z, r_A) :$$

$$\exists r_0, f_k(x, r_0) = y_0 \textit{ or } (\forall x',\ (\forall r_0,\ y_0 \neq f_k(x', r_0)) \textit{ or } y_1 \neq f_k(x', r_1))] > 1 - \mu(n).$$

3 On Obfuscation versus Interactive Extraction

We present the three theorems mentioned in the introduction concerning the connection between obfuscation and interactive extraction with different extraction rates. Recall, the first theorem says that every function is either weakly extractable or weakly obfuscatable. The second theorem builds on the first one to imply that every weakly verifiable function is either weakly obfuscatable or extractable with vanishing but noticeable error. The final theorem states that negligible-error extraction can be achieved if and only if certain conditions on the adversary are met. These conditions, termed “reliable consistency” in the introduction, are discussed and formalized in Section 3.2.

The statement that any function is either extractable or obfuscatable is to some degree intuitive. After all, these two notions are complementary in some way. For instance, suppose there is an obfuscated program that hides a license key inside it and is able to compute a new hash of the key. If we look at such a program from an extractability point of view, this means that there is a machine that simulates this program and computes the functionality mentioned above. Moreover, no extractor can recover the license key by the assumption that the obfuscated program hides it. Going in the reverse direction, it seems intuitive that the existence of an extractor for every adversary implies the absence of an obfuscation of such a functionality.

In the next theorem, we formalize and show that the intuition mentioned in the previous paragraph is sound. In more detail, statement 1 of this theorem (the obfuscation clause) states that there is a well-spread distribution, $\mathbb{X}$, on the input (think of this as the license key of the previous example) and an obfuscator, G_n, that takes a license key, x, and produces an obfuscated program, $g(x)$. In turn, $g(x)$ takes an input r and produces a new image of x using r as random coins for the function, i.e., $g(x)(r) = f(x, r)$. Moreover, $g(x)$ is required to be one-way in x but not required to succeed in computing this functionality more than noticeably often. In the theorem, we use the terminology $g(x)(\perp)$ to refer to a fixed hash of x available in the clear in the obfuscated program. On the other hand, statement 2 (the extraction clause) says that any adversary, A, with any distribution on its input, z, r_A (z is auxiliary information and r_A is the random coins for A), that is consistent in the 3-round game discussed in the introduction, there is a corresponding extractor that recovers a preimage. In more detail, A is supposed to produce, with noticeable success, an image, y_0 in the first round and then again y_1 in the third round, such that there is a preimage common to both y_0 and y_1. Moreover, the extractor is supposed to succeed only noticeably often.

Theorem 1. *Let $\mathbb{F} = \{f_n\}_{n\in\mathbb{N}}$ be any randomized family of functions and $\mathbb{R} = \{R_n\}_{n\in\mathbb{N}}$ be any distribution on the randomness domain of $\mathbb{F}$. Then, exactly one of the following two statements should hold:*

1. *There is a well-spread distribution $\mathbb{X}$ on the input domain of $\mathbb{F}$, a probabilistic function, $\mathbb{G} = \{G_n\}$ such that for any nonuniform polynomial-time machine, A:*
   (Obfuscation)

   $$Pr[x \leftarrow X_n,\ g(x) \leftarrow G_n(x),\ x' = A(g(x)) : \exists r',\ g(x)(\perp) = f_n(x', r')] \leq \mu(n).$$ [4]

   (Functionality)

   $$Pr[x \leftarrow X_n,\ g(x) \leftarrow G_n(x),\ r \leftarrow R_n : \exists r',\ g(x)(r) = f_n(x, r) \textit{ and } g(x)(\perp) = f_n(x, r')],$$

   is nonnegligible in n. Moreover, $g(x)(r)$ is efficiently computable, for any r.
2. *For any probabilistic polynomial-time machine (PPT), A, any infinite subset of security parameters, $\mathbb{N}'$, any distribution, $\mathbb{ZR} = \{ZR_n\}_{n \in \mathbb{N}'}$, on auxiliary information and the private input of A, if:*
   (Consistency)

   $$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1) : \\ \exists x',\ r_0,\ y_0 = f_n(x', r_0)) \textit{ and } y_1 = f_n(x', r_1))], \tag{1}$$

   is nonnegligible in n, then there exists a nonuniform polynomial-time machine, $\mathbb{K}$, such that:
   (Extraction)

   $$Pr[(z, r_A) \leftarrow ZR_n,\ (y_0, s) = A(z, r_A),\ x = \mathcal{K}(z, r_A) : \exists r_0,\ y_0 = f_n(x, r_0)], \tag{2}$$

   is nonnegligible in n.

We emphasize that the previous theorem holds for any function. That is, it does not assume anything about the function, not even that it is efficiently computable. At a high level, the proof proceeds as follows. If f is not extractable, we take an adversary that violates this property and construct from it a distribution on the input to f (for clarity, refer to this as the license distribution) and an obfuscation on this distribution such that the obfuscation hides the license but is able to compute new images of it. In more detail, the license distribution is the distribution induced by A on preimages of its consistent output. For instance, if A *always* outputs $f_n(0, r_0)$ in the first round and $f_n(0, r_1)$ in the third round (in this case there is a straightforward extractor), then the induced distribution always samples 0. Moreover, the corresponding obfuscation is simply the input of A that causes A to output valid images of the license. Observe that the license distribution is well-spread because otherwise the nonuniform extractor can invert with noticeable probability. Therefore, using this license distribution with the corresponding obfuscation, statement 1 follows from the negation of statement 2. The other direction is easier to see and has been referred to in the second paragraph of this section.

Corollary 1. *Any deterministic one-way function is not even weakly extractable. That is, any deterministic one-way function satisfies statement 1 of Theorem 1. Moreover, this remains true if the function is not efficiently computable.*

[4] Here and in the rest of the paper, μ denotes a negligible function.

3.1 Amplifying Extraction

Theorem 1 says each function has a weakly extractable or weakly obfuscatable property. Next, we investigate conditions that allow for amplifying knowledge extraction in the interactive setting. In particular, the goal in this section is to reach a vanishing but noticeable extraction error. Recall from the introduction, this term means that for every polynomial, p, there is an extractor that may depend on p and fails at most $\frac{1}{p}$ of the time. In Section 3.2, we address extraction with negligible error.

Not surprisingly, functions that admit such a property require more than the negation of statement 1 of Theorem 1. Recall that Theorem 1 holds for any function, in particular, not efficiently-computable functions. However, to decrease the extraction error, efficient verification is needed. For the purpose of amplifying extraction, common notions of verification (e.g., Definition 3) are sufficient. However, a weaker but contrived form of verification is also sufficient, and, in the case of injective functions (i.e., for all y, there is no more than one x such that $y = f_n(x, r)$ for some r), is also necessary. Thus, we use this notion in the following theorem for the purpose of achieving a characterization instead of an implication. Informally, weak verification means that there is a verifier tailored for every adversary, A. It receives x and the input of A and determines whether the output of A is a valid image of x. Moreover, the verifier is allowed to fail, when A is consistent, with noticeable probability.

Definition 3 (Efficient Verification, [7]).
A function family , $\mathbb{F} = \{f_n\}_{n\in\mathbb{N}}$, satisfies efficient verification if there exists a deterministic polynomial time algorithm, $V_{\mathbb{F}}$ such that:

$$\forall n \in \mathbb{N},\ x \in \{0,1\}^n,\ y \in range(f_n),\ V_{\mathbb{F}}(x, y) = 1 \textit{ iff } \exists r, y = f_n(x, r).$$

Definition 4 (Weak Verification).
A function family , $\mathbb{F} = \{f_n\}_{n\in\mathbb{N}}$, satisfies weak verification if for every PPT, A (with input z, r_A), any distribution, $\mathbb{ZR} = \{ZR_n\}_{n\in\mathbb{N}'}$, on auxiliary information and the private input of A, and any polynomial p, there exists a nonuniform polynomial-time machine, $V_{A,ZR,p}$, such that for sufficiently large $n \in \mathbb{N}'$:

$$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1) :$$
$$(\exists x, r_0,\ V_{A,ZR,p}(x, z, r_A) \neq 1 \textit{ and } f_n(x, r_0) = y_0 \textit{ or } \exists x, V_{A,ZR,p}(x, z, r_A) = 1 \textit{ and } \forall r_0, f_n(x, r_0) \neq y_0)$$
$$\textit{and } (\exists x, r_0,\ f_n(x, r_0) = y_0 \textit{ and } f_n(x, r_1) = y_1)] < \frac{1}{p(n)}.$$

Theorem 2. *Let $\mathbb{F} = \{f_n\}_{n\in\mathbb{N}}$ be any randomized function family that is weakly extractable (satisfies statement 2 of Theorem 1). If $\mathbb{F}$ is weakly verifiable (as in Definition 4), then for any PPT A, any distribution, $\mathbb{ZR} = \{ZR_n\}_{n\in\mathbb{N}'}$, on auxiliary information and the private input of A, there exists a family of nonuniform polynomial-time machines, $\mathbb{U} = \{U_i\}_{i\in\mathbb{N}}$ such that for any polynomial p, there is an index i_p where for all $i \geq i_p$ and sufficiently large $n \in \mathbb{N}'$:*

$$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1),\ x = U_i(z, r_A) :$$
$$(\exists r_0, f_n(x, r_0) = y_0 \textit{ or } (\forall x',\ (\forall r_0,\ y_0 \neq f_n(x', r_0)) \textit{ or } y_1 \neq f_n(x', r_1))] > 1 - \frac{1}{p(n)}. \tag{3}$$

Moreover, this implication is an equivalence for injective functions.

The proof uses, in an essential way, an amplification lemma which is a version of Impagliazzo's hard-core lemma [19] applied to this setting. At a very high level, this lemma asserts the existence of a family of machines, $\mathbb{U}$, such that "no machine can succeed noticeably where all of these machines fail". Using this lemma, we then claim that for every polynomial, p, there is a member $U_{i_p} \in \mathbb{U}$ that fails in extracting a preimage with a probability at most $\frac{1}{p}$. If this were not to be the case, then this means that there is some polynomial p, where every machine in $\mathbb{U}$ fails with probability at least $\frac{1}{p}$. This implies that there is a noticeable fraction of the domain where A is consistent yet all members of $\mathbb{U}$ fail. Lets restrict the distribution on the input of A to those on which such an event occurs. We then apply Theorem 1, in particular, statement 2, to obtain an extractor with noticeable success contradicting the lemma.
The following corollary is one of the main applications of this result.

Corollary 2. *Every POW function with auxiliary information that is collision resistant and has public randomness is extractable with vanishing but noticeable error in the interactive setting (as in Theorem 2).*

3.2 Towards Extraction with Negligible Error

The previous section underscores the conditions that are necessary (at least for injective functions) and sufficient for extraction with vanishing but noticeable error. Here, we address the question of obtaining extraction with negligible error. As before, we show necessary and sufficient conditions to achieve this objective. However, unlike the previous results, the conditions are on the adversary itself and not on the function under study. Moreover, as we discuss later on, this result is in the uniform setting only.

Conditions for extraction with negligible error. As we mentioned in the introduction, extraction with negligible error requires "reliable consistency" on the behalf of the adversary. Informally, we show that negligible extraction error is possible for a particular adversary, A, if it can answer challenges consistently with probability bounded from below by the inverse of some fixed polynomial. Informally, it may be the case that A answers consistently with noticeable probability. Yet, depending on its input, its corresponding consistency probability (taken over the random coins of the challenger) can be arbitrary small though still noticeable. In such a scenario, extraction can not achieve negligible error because as answers are less likely to be consistent, extraction requires more effort and time to find a preimage. On the other hand, if for almost all of its input, A answers consistently with a probability bounded from below by an inverse polynomial, this bound can be translated into an upper bound on the running time of the extractor.
We elaborate on these conditions through a toy example. Suppose there is a function, f and an adversary A with the following properties. A outputs a consistent pair (y_0, y_1) with probability $\frac{1}{n^i}$ for every element in the i^{th} $\frac{2^n}{n}$ fraction of the input domain for A. Here, the probability is taken over random coins sent by the challenger in round 2. Formally, we have for every n, and every $(z, r_A) \in [\frac{i2^n}{n}, \frac{(i+1)2^n}{n}]$:

$$Pr[r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1) : \exists x, r_0,\ f_n(x, r_0) = y_0$$

$$\text{and } f_n(x, r_1) = y_1] = \frac{1}{n^i}.$$

Now, it may be the case that extraction depends on how successful A is in answering challenges. If this is so, then extraction is proportional to consistency. In other words, as A becomes less consistent (that is, as its input is chosen from the upper fraction of the domain), extraction requires more time to achieve the same success rate. In such a scenario, it turns out that overwhelming success requires super-polynomial time. In other words, noticeable extraction error is unavoidable.
In the previous example, we assume that A has a noticeable success in every fraction of the input domain. Also, we assume that A can not do any better. In other words, A can not amplify its success rate. However, there are cases where A can indeed amplify its success, e.g., A may provide wrong answers intentionally even though it can easily compute the correct ones. In such a scenario, extraction with negligible error is possible. As an example, consider an adversary, A, that provides wrong answers intentionally. A receives x as input, computes i such that $x \in [\frac{i2^n}{n}, \frac{(i+1)2^n}{n}]$, and gives the correct answer only if $r_1 \in [0, \frac{2^n}{n^i}]$. Even though A satisfies the previous condition, an extractor can easily recover x by reading it from the input. So, we need a meaningful way to separate the notion of "truthful" failure from "intentional" failure. In the next theorem, we capture the notion of intentional failure through the existence of another machine A' that behaves similarly to A, yet it amplifies its consistency.

Uniform Setting. The proof of Theorem 2 uses a diagonalization technique to show that no machine can succeed "substantially" where the family $\mathbb{U}$ fails. The diagonalization is over machines that succeeds noticeably over inputs of some length n. This technique works because this set of machines is enumerable. (Specifically, there are at most n machines that each succeeds exclusively with probability $\frac{1}{n}$ and so on.) However, this technique fails when we try to use it to achieve negligible error in polynomial time. Two factors seem to prevent this technique from working. First, the set of nonuniform polynomial-time machines is not enumerable and so we can not diagonalize over this set (as we discuss later on, we use the enumeration of uniform machines to prove this result in the uniform setting). Second, if we instead consider machines that succeed exclusively, as in the previous theorem, we need to take into account those that succeed with negligible probability, yet the probability is not "very negligible", say, $\frac{1}{n^{logn}}$. However, this causes $\mathbb{U}$ to be slightly super-polynomial. Consequently, the next theorem applies to the uniform setting only. It uses a uniform version of Theorem 1 which can be found in the full version of the paper.
In words, reliable consistency in the next theorem refers to a new machine, A', that replaces an adversary, A, with the purpose of undoing any intentional failure on behalf of A. The conditions on A' are as follows: (1) the output of A' is equivalent to A in the first round, (2) the consistency of A' is not any worse than that of A, and (3) there is a fixed polynomial, $p_{A'}$, such that almost all inputs to A' cause it to be either consistent negligibly or with probability at least $\frac{1}{p_{A'}}$. If there is such an A' then extraction with negligible extraction error is possible. Moreover, the converse is also true for efficiently computable and verifiable functions.

Theorem 3. *Let $\mathbb{F} = \{f_n\}_{n\in\mathbb{N}}$ be any randomized function family that satisfies the uniform version of statement 2 of Theorem 1 and is weakly verifiable (as in Definition 4, except with respect to uniform deterministic machines).*
Let A be any PPT and $\mathbb{ZR} = \{ZR_n\}_{n\in\mathbb{N}'}$ be any distribution on auxiliary information and the private input of A. If there is another PPT, A', satisfying the following three conditions of reliable consistency:

1. $A'(z, r_A) = A(z, r_A)$ *for all* z, r_A.

2.

$$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A'(z, r_A),\ y_1 = A'(s, r_1) :$$
$$\exists x',\ r_0,\ y_0 = f_n(x', r_0))\ \textit{and}\ y_1 = f_n(x', r_1))]$$
$$\geq Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1) :$$
$$\exists x',\ r_0,\ y_0 = f_n(x', r_0))\ \textit{and}\ y_1 = f_n(x', r_1))] - \mu(n)$$

3. There exists a polynomial $p_{A'}$*, such that for any polynomial* $q > p_{A'}$*:*

$$Pr[(z,r_A)\leftarrow ZR_n:$$

$$\tfrac{1}{q(n)} \leq Pr[r_1 \leftarrow R_n,\ (y_0,s)=A'(z,r_A),\ y_1=A'(s,r_1,a_{A'}) \colon \exists x',\ r_0,\ y_0=f_n(x',r_0)\ \textit{and}\ y_1=f_n(x',r_1)] \leq \tfrac{1}{p_{A'}(n)}] \leq \mu(n)$$

then there is a deterministic polynomial-time machine, $\mathcal{K}$ *such that for* $n \in \mathbb{N}'$*:*

$$Pr[(z, r_A) \leftarrow ZR_n,\ r_1 \leftarrow R_n,\ (y_0, s) = A(z, r_A),\ y_1 = A(s, r_1),\ x = \mathcal{K}(z, r_A) :$$
$$\exists r_0, f_n(x, r_0) = y_0\ \textit{or}\ (\forall x'(\forall r_0,\ y_0 \neq f_n(x', r_0))\ \textit{or}\ y_1 \neq f_n(x', r_1))] > 1 - \mu(n). \tag{4}$$

Moreover, if $\mathbb{F}$ *is efficiently computable and verifiable (as in Definition 3), then the converse is also true.*

The proof is similar to that of Theorem 2. There are two points worth highlighting. The proof uses a uniform version of the amplification lemma. Informally, this lemma provides a family of machines, $\mathbb{U}$, such that any machine can not succeed even negligibly where this family fails. At a high level, each $U_i \in \mathbb{U}$ contains the first i machines in an enumeration of uniform polynomial-time machine. This ensures that every polynomial-time machine is eventually included in the family. We claim that there is a member of this family that achieves negligible extraction error. If this were not to be the case, then for every member U_i there is a polynomial p_i such that U_i fails with probability at least $\frac{1}{p_i}$. Note that p_i may increase as i increases. However, by the third condition on A', consistency of A' is bounded from below by the inverse of a fixed polynomial which is independent of p_i. This is important because when we restrict the input distribution to where A' is consistent and $\mathbb{U}$ fails, A' remains consistent with noticeable probability. Consequently, we can apply Theorem 1 to get an extractor with noticeable success contradicting the lemma.

Corollary 3. *Any deterministic and efficiently-verifiable (i.e., given* x *and* y*, it is easy to decide whether* $f(x) = y$*) function is extractable with negligible error if and only if it is weakly extractable in the uniform setting.*

4 Knowledge-preserving Reductions

In Section 3, we investigate the relationships among different notions of extraction. We address questions regarding the possibility that functions satisfy some extractability properties, such as weak extraction, extraction with noticeable error, or extraction with negligible error. Results in this line of work show equivalence among some notions of extraction, e.g., extraction with noticeable error is equivalent to extraction with nonnegligible success for deterministic and efficiently verifiable functions (Corollary 3).

Here, we take a different approach. Specifically, we investigate building extractable functions with additional hardness properties from extractable functions with simpler computational assumptions. In particular, we revisit the literature on reductions among primitives to see if these reductions or variations of preserve noninteractive extraction.

The results are mostly positive. In particular, reductions from weak one-way functions to strong one-way functions, from one-way functions to 2-round commitments and public-key encryption scheme (assuming in addition a trapdoor permutation) are knowledge preserving or can be easily modified to be so. Moreover, extractable pseudorandom generators imply extractable pseudorandom functions and extractable 2-round commitments. One important open question is whether extractable one-way functions imply extractable pseudorandom generators. In pursuit of answering this question, we show that the HILL construction [17] is not knowledge preserving. On the other hand, an extractable pseudorandom generator can be constructed from the KE and the DDH assumptions.

Next, we provide a detailed presentation of these results. They address noninteractive extraction with negligible error only. Interactive extraction is primarily useful for probabilistic functions because by Corollary 1, deterministic one-way functions and pseudorandom generators are not interactively extractable. As for probabilistic functions, [8] provides a transformation from POW functions to interactively-extractable POW functions. Moreover, every POW function with auxiliary information and public randomness is interactively extractable (Corollary 2).

From extractable weak one-way to extractable strong one-way functions. The standard reduction from weak one-way functions to strong one-way functions [24, 12] is knowledge preserving. Specifically, let $\mathbb{F} = \{\{f_k\}_{k \in K_n}\}_{n \in \mathbb{N}}$ be a family of weak functions with $\frac{1}{p}$ as a lower bound on the failure probability of all polynomial-time machines. Furthermore, suppose that $\mathbb{F}$ is extractable with negligible error with respect to some well-spread distribution, $\mathbb{K}$, on the function description. Then, the family, $\mathbb{G} = \{\{g_k\}_{k \in K_n}\}_{n \in \mathbb{N}}$, where $g_k(x_1, ..., x_{np(n)}) = f_k(x_1), ..., f_k(x_{np(n)})$, is also extractable with respect to $\mathbb{K}$.

Let A be any adversary that receives k, z, r_A as input (where z and r_A are auxiliary information and random coins of A, respectively) and outputs y in the range of G_k. Let B be a machine that receives k, z, r_A, i as input and outputs y_i, where i is uniform and $A(k, z, r_A) = y_1, ..., y_{np(n)}$. Note that B outputs a valid image under f_k with at least the same probability as A outputs a valid image under g_k. Therefore, there is a corresponding extractor, $\mathcal{K}_B$, for B. Let $\mathcal{K}_A$ be an extractor for A that runs $\mathcal{K}_B$ on k, z, r_A, i for $i = 1$ to $np(n)$. Except with negligible probability, if A outputs a valid image, $\mathcal{K}_B$ computes the correct images for all $f_k(x_i)$. Thus, $\mathcal{K}_A$ is a negligible-error extractor for A.

From extractable one-way functions to extractable pseudorandom generators. First, we point out that the HILL construction [17] of pseudorandom generator from even injective one-way functions is not knowledge preserving. Specifically, the family, $\mathbb{G}$, is not extractable, where $G_k(x, h) = h(f_k(x)), h, p(x)$, f_k is an extractable, $1-1$ one-way function, h is a hash function, and p is a hardcore predicate for f_k. This is so because the adversary, that receives and outputs a random string, succeeds with noticeable probability in producing a valid image under G_k. On the other hand, no extractor can recover a preimage because G_k is pseudorandom.

Constructing extractable pseudorandom generators from extractable one-way functions remains open. The obstacle seems to be that somehow, $f_k(x)$, should be easy to compute from the output of the generator so that it is possible to use the original extractor to recover x. Consequently, for G to be a pseudorandom generator, it should also be easy to compute $f_k(x)$ from a random string, for some x. However, the range of f may be distinguishable from uniform, e.g., the first n bits may always be 0. So, it is not clear how to put $f_k(x)$ in the output without compromising pseudorandomness.

A point worth mentioning here is that it is possible to construct extractable pseudorandom generators from a stronger knowledge requirement on the one-way function. The original knowledge assumptions states that any adversary that outputs $f_k(x)$ *as a sequence of bits* "knows" x. Consider the following stronger version. Informally, if an adversary outputs $f_k(x)$ specified in another representation, it should still know x. In particular, the type of representation, $\mathcal{R}$, we are interested in is a randomized representation of strings, where $\mathcal{R}(y, r)$ is indistinguishable from uniform and every $\mathcal{R}(y, r)$ has a unique preimage (except with negligible probability). We give a concrete example: Let π be a one-way permutation and b be a corresponding hardcore predicate. Then, $\mathcal{R}(y, r_1, ..., r_{|y|}) = \pi(r_1), ..., \pi(r_{|y|}), y \oplus b(r_1), ..., b(r_{|y|})$. Note that $\mathcal{R}$ is pseudorandom and unambiguous, in that there is a single y as a valid preimage of any output. Now, if f_k is extractable with respect to this representation, then the following construction is an extractable family of pseudorandom generators.

$$G_k(x, r_1, ..., r_{|f_k(x)|}) = \mathcal{R}(f_k(x), r_1, ..., r_{|f_k(x)|}), G'(x) \oplus r_1, ..., r_{|f_k(x)|},$$

where G' is another pseudorandom generator with a suitable expansion factor that remains pseudorandom in the presence of f (but G' is not assumed to be extractable). In other words, $f(x), G'(x)$ is assumed to be indistinguishable from $f(x), U_{|G'(x)|}$ (in this section, U_l denotes a uniform variable over strings of length l).[5]

Finally, we mention that the knowledge of exponent assumption [16] (with the DDH assumption) imply the existence of extractable pseudorandom generators, specifically, $G_{g,g^a}(x) = g^x, g^{ax}$, where g is a generator for the group for which these assumptions apply.

From extractable pseudorandom generators to extractable pseudorandom functions. The notion of extractable pseudorandom functions is slightly different from the notions considered so far. Informally, a pseudorandom function is extractable if any adversary that computes $f_k(x, r)$, for any r that a challenger chooses, has a corresponding extractor that recovers x.

Formally, for any PPT A, any well-spread distribution, K_n, on the function description, any distribution, $\mathbb{ZR} = \{ZR_n\}_{n \in \mathbb{N}'}$, on auxiliary information and the private input of A, there is polynomial-time machines, $\mathcal{K}$, such that:

$$Pr[(z, r_A) \leftarrow ZR_n,\ k \leftarrow K_n,\ x = \mathcal{K}(k, z, r_A) :$$

$$\exists r, f_k(x, r) \neq A(k, z, r_A, r) \text{ and } \exists x', \forall r', f_k(x', r') = A(k, z, r_A, r')] \leq \mu(n).$$

The construction of extractable pseudorandom functions uses the construction of [13] on all input, except 0. On input 0, the output is exactly that of the extractable generator in order to allow for successful extraction. Formally, let G^1 be any injective and extractable pseudorandom generator with a $2n^2$ (or more) expansion

[5] Note that the machine that outputs a random string as a possible representation of $f_k(x)$ under $\mathcal{R}$ does not succeed considerably better than the machine that output a random string as a possible $f_k(x)$.

factor. Let b a hardcore bit for G^1 and $G^2_k(x_1,\ldots,x_n) = G^1_k(b(x_1),\ldots,b(x_n))$, where $|x_1| = \cdots = |x_n| = n$. W.l.o.g. assume G^2 has a $2n$ expansion factor, otherwise, trim the output to a suitable length. Let $\mathbb{F}'$ be the family of pseudorandom functions obtained by applying the construction of [13] on G^2. Then, the extractable family of pseudorandom functions, $\mathbb{F} = \{\{f_k\}_{k\in K_n}\}_{n\in\mathbb{N}}$, is defined as follows:

$$f_k((x_1,\ldots,x_n),r) = \begin{cases} G^1_k(x_1),\ldots,G^1_k(x_n) & \text{if } r = 0 \\ f'_k((x_1,\ldots,x_n),r) & \text{otherwise} \end{cases}$$

Let A be any PPT that receives k,z,r_A,r and outputs $f_k(x_1,\ldots,x_n,r)$ for some $x_1,\ldots,x_n$. Let B be a machine that receives k,z,r_A,i (where i is uniform), computes $A(k,z,r_A,0) = G^1_k(x_1),\ldots,G^1_k(x_n)$ and outputs $G^1(x_i)$. Since G^1 is extractable, there is a machine, $\mathcal{K}_B$ that recovers the corresponding x_i on input k,z,r_A,i. Then, the extractor, $\mathcal{K}_A$, for A and $\mathbb{F}$, simulates $\mathcal{K}_B$ on input k,z,r_A,i, for $i = 1,\ldots,n$, and outputs $x_1,\ldots,x_n$.

From extractable one-way functions to extractable public-key encryption. Before we discuss extractable public-key encryption, we briefly mention that private-key encryption with a “strong” extraction property (that is, plaintext-aware [5]) can be easily constructed from standard computational assumptions *without knowledge assumptions*. However, we emphasize that not all private-key encryption are extractable, e.g., a random string is a valid ciphertext under $E_{sk}(m,r) = r, m\oplus f_{sk}(r)$ [12], where f_{sk} is a pseudorandom function. However, the previous construction can be easily modified to become extractable. Specifically, $E_{sk=(sk_1,sk_2)}(m,r) = r, m\oplus f_{sk_1}(r), f_{sk_2}(m,r)$ has the property that without knowledge of sk, it is hard to find a *new* ciphertext even if the adversary sees encryption of multiple messages.

Extractable one-way functions can be used with a trapdoor permutation to construct public-key encryption schemes with the property that any adversary that computes a ciphertext *without seeing another ciphertext* “knows” the corresponding plaintext. This notion is similar to plaintext-aware encryption [5, 18, 4, 11]. Informally, the latter notion says that no adversary, with access to ciphertext of messages it may not know, can produce a ciphertext without knowing the corresponding plaintext. In this work we focus on extraction with independent auxiliary information only. So, we leave the study of constructing plaintext-aware encryption from extractable functions to future work as it requires extraction with dependent auxiliary information [8]. We note that [8] constructs plaintext-aware encryption from extractable POW functions with dependent auxiliary information.

Let $\mathbb{F} = \{\{f_k\}_{k\in K_n}\}_{n\in\mathbb{N}}$ and $\Pi = \{\{\pi_{pk}\}_{pk\in PK_n}\}_{n\in\mathbb{N}}$ be families of extractable one-way functions and trapdoor permutations, respectively. Moreover, suppose that $\mathbb{F}$ and Π remain one-way with respect to each other, specifically, for a uniform r,k,pk, $f_k(r),\pi_{pk}(r)$ is one-way. Let b be a hardcore predicate for the function $g_{k,pk}(r) = f_k(r),\pi_{pk}(r)$. Note that g is extractable and injective. Let $E_{k,pk}(m,(r_1,\ldots,r_n)) = g_{k,pk}(r_1),\ldots,g_{k,pk}(r_n), m\oplus b(r_1),\ldots,b(r_n)$. It can be show that for any adversary that computes a valid ciphertext, without seeing another ciphertext, there is an extractor that recovers $r_1,\ldots,r_n$ and consequently, m.

From extractable one-way functions to extractable $1-1$ trapdoor functions. Observe that g, as defined above, is an extractable $1-1$ trapdoor function if $\mathbb{F}$ and Π remain one-way with respect to each other. Moreover, the same result holds when Π is a family of $1-1$ trapdoor functions.

Extractable commitments. Informally, an extractable commitments guarantee *at the commit stage* that the sender knows the secret if the commitment is valid (that is, it can be opened). Even though in a stand-alone protocol, this additional property may seem irrelevant (because the sender reveals the secret in the decommit stage and nothing happens between these two stages), it is one of several important properties that come into play in more complex protocols with stronger security requirement. Thus, extractable commitments in the CRS model were introduced and studied in [22, 9, 10] as part of zero-knowledge proofs and universally-composable commitments.
We show that known commitments constructions from injective one-way function [6] and from pseudorandom generators [21] can be easily modified into 2-round extractable commitments if the underlying primitives are extractable. We note that Ventre and Visconti [23], independently construct 2-round extractable commitments from plaintext-aware encryption schemes (with additional assumptions).

Extractable commitments from $1-1$ *extractable, one-way functions.* Let $\mathbb{F}$ be a family of injective and extractable one-way functions. The 2-commitment starts with the receiver sending a random function description, k, and the sender responds with $f_k(u_1), \ldots, f_k(u_n), m \oplus b(u_1), \ldots, b(u_n)$, where b is a hardcore bit for f_k. Note that it is essential for the hiding property that the family, $\mathbb{F}$ be one-way with respect to *any* function in the family.

Extractable commitments from extractable pseudorandom generators. We modify the 2-round commitment scheme of [21] to make it extractable. In the first round, the receiver sends random strings $r_1, \ldots, r_n$ and the description, k, for the pseudorandom generator. In the second round, the senders responds with $g_k(u_1) \oplus r_1^{m_1}, \ldots, g_k(u_n) \oplus r_n^{m_n}$, where $r_i^{m_i} = r_i$ if $m_i = 0$ and $r_i^{m_i} = 0^{3n}$, otherwise. As in the previous construction, every function in the family is assumed to be pseudorandom.

References

1. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *Crypto*, 2001.
2. M. Bellare and O. Goldreich. On defining proofs of knowledge. *Crypto*, 1992.
3. M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. *Crypto*, 2004.
4. M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. *Asiacrypt*, 2004.
5. M. Bellare and P. Rogaway. Optimal asymmetric encryption. *EuroCrypt*, 1994.
6. M. Blum. Coin flipping by phone. *IEEE Computer conference*, 1982.
7. R. Canetti. Towards realizing random oracles. *Crypto*, 1997.
8. R. Canetti and R. R. Dakdouk. Extractable perfectly one-way functions. *ICALP, Track C*, 2008.
9. R. Canetti and R. Fischlin. Universally-composable commitment. *Crypto*, 2001.
10. Giovanni Di Crescenzo. Equivocable and extractable commitment schemes. *SCN*, 2002.

11. A. Dent. The cramer-shoup encryption scheme is plaintext aware in the standard model. *Eurocrypt*, 2006.
12. O. Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.
13. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33, 1986.
14. S. Goldwasser and Y. T. Kalai. On the impossibility of obfuscation with auxiliary input. *FOCS*, 2005.
15. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. *STOC*, 1985.
16. S. Hada and T. Tanaka. On the existence of 3-round zero-knowledge protocols. *Crypto*, 1998.
17. J. Hastad, L. Levin. R. Impagliazzo, and M. Luby. Construction of a pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 1999.
18. J. Herzog, M. Liskov, and S. Micali. Plaintext awareness via key registration. *Crypto*, 2003.
19. R. Impagliazzo. Hard-core distributions for somewhat hard problems. *FOCS*, 1995.
20. M. Lepinski. On the existence of 3-round zero-knowledge proofs. *M.S. Thesis*, 2002.
21. M. Naor. Bit commitments using pseudorandom generators. *Journal of Cryptology*, 1991.
22. A. De Santis, G. Di Crescenzo, and G. Persiano. Necessary and sufficient assumptions for non-interactive zero-knowledge proofs of knowledge for all np relations. *ICALP*, 2000.
23. C. Ventre and I. Visconti. Message-aware commitment schemes. *Unpublished manuscript*, 2008.
24. A.C. Yao. Theory and application of trapdoor functions. *FOCS*, 1982.
25. Y. Zheng and J. Seberry. Immunizing public key cryptosystems against chosen ciphertext attacks. *Journal on Selected Areas in Communication*, 1993.