Computational Extractors and Pseudorandomness

Dana Dachman-Soled[1], Rosario Gennaro[2], Hugo Krawczyk[2], and Tal Malkin[3]

[1] Microsoft Research New England
dadachma@microsoft.com
[2] IBM Research
rosario@us.ibm.com, hugo@ee.technion.ac.il
[3] Columbia University
tal@cs.columbia.edu

Abstract. *Computational extractors* are efficient procedures that map a source of sufficiently high min-entropy to an output that is computationally indistinguishable from uniform. By relaxing the statistical closeness property of traditional randomness extractors one hopes to improve the efficiency and entropy parameters of these extractors, while keeping their utility for cryptographic applications. In this work we investigate computational extractors and consider questions of existence and inherent complexity from the theoretical and practical angles, with particular focus on the relationship to pseudorandomness.

An obvious way to build a computational extractor is via the "extract-then-prg" method: apply a statistical extractor and use its output to seed a PRG. This approach carries with it the entropy cost inherent to implementing statistical extractors, namely, the source entropy needs to be substantially higher than the PRG's seed length. It also requires a PRG and thus relies on one-way functions.

We study the necessity of one-way functions in the construction of computational extractors and determine matching lower and upper bounds on the "black-box efficiency" of generic constructions of computational extractors that use a one-way permutation as an oracle. Under this efficiency measure we prove a direct correspondence between the complexity of computational extractors and that of pseudorandom generators, showing the optimality of the extract-then-prg approach for generic constructions of computational extractors and confirming the intuition that to build a computational extractor via a PRG one needs to make up for the entropy gap intrinsic to statistical extractors.

On the other hand, we show that with stronger cryptographic primitives one can have more entropy- and computationally-efficient constructions. In particular, we show a construction of a very practical computational extractor from any weak PRF without resorting to statistical extractors.

See [DGKM11] for the full version of this paper.

1 Introduction

Randomness extractors (or simply 'extractors') are algorithms that map sources of sufficient min-entropy to outputs that are statistically close to uniform. Randomness extraction has become a central and ubiquitous notion in complexity theory and theoretical computer science with innumerable applications and surprising connections to other notions. Cryptography, too, has greatly benefited from this notion. Cryptographic applications of randomness extractors range from the construction of pseudorandom generators from one-way functions to the design of cryptographic functionalities from noisy and weak sources (including applications to quantum cryptography) to the more recent advances in areas such as leakage- and exposure-resilient cryptography, circular encryption, lattice-based cryptosystems, and more. Randomness extractors have also found important uses in practical applications, particularly for the construction of key derivation functions. In many of these cryptographic applications, the defining property of randomness extractors, namely, statistical closeness of their output to a uniform distribution, can often be relaxed and replaced with computational indistinguishability. Extractors that provide this relaxed guarantee are called *computational extractors*, and they are the main object studied in this paper.

Let us review informally some basic facts about statistical extractors and the associated parameters n, m, k, δ. A function $Ext : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ is a $(k, 2^{-\delta})$-*statistical extractor* if for any distribution X on $\{0,1\}^n$ with min-entropy k, the statistical distance between $Ext(X, U_\ell)$ and U_m is at most $2^{-\delta}$, where U_ℓ, U_m denote the uniform distribution over $\{0,1\}^\ell, \{0,1\}^m$, respectively. Note that extractors are randomized via the second argument called a seed or key (in our actual definitions we require the seed to be output, i.e., the so called *strong* extractor). We are interested in extractors where the values k and $2^{-\delta}$ are as small as possible (i.e., we want to minimize the entropy requirement from the source and get as small as possible statistical distance of the output to uniform). It is known how to construct statistical extractors that achieve $\delta = (k + \ell - m)/2$ [NZ96,HILL99]. Radhakrishnan and Ta-Shma [RTS00] show that this bound on δ is optimal, by showing how to build, for every extractor with parameters as above, a source distribution of min-entropy k for which the output of the extractor is $2^{-\delta}$-far from uniform for $\delta = (k + \ell - m)/2$. In the sequel we refer to this as the *RT bound*.

A major motivation to study computational extractors is that they allow us to go beyond the RT bound by replacing statistical closeness to uniform with computational indistinguishability. Indeed, an obvious way to do so is to first use a statistical extractor applied to the source distribution to obtain a short statistically close-to-uniform string and then use this string as a seed to a pseudorandom generator (PRG) to obtain more bits that are indistinguishable from uniform. We will refer to this as the *extract-then-prg* approach.

While the latter is a natural way to build computational extractors, it is not the only one or necessarily the best one, especially when implemented in practical settings. In particular, this approach carries with it the entropy limitations of statistical extractors as set by the RT bound, a serious concern in cases where

the entropy of the source is too small to produce (via the statistical extractor) a sufficiently long key for the PRG. For example, consider the use of an extractor to convert a 160-bit elliptic curve Diffie-Hellman value (which by the DDH assumption has 160 bits of computational min-entropy) into a 128-bit seed for an AES-based PRG. Applying a statistical extractor to the DH value only guarantees a poor indistinguishability bound of 2^{-16} (i.e., $\delta = (160 - 128)/2$). If we wanted to preserve, say, 100-bit security we would need $\delta = 100$ bringing the required source entropy to 328 ($= 128 + 2 \cdot 100$).

One way around this problem is to build dedicated computational extractors based on cryptographic functions. Such an approach is taken in [Kra10,DGH+04], where computational extractors are built using *specific* schemes (HMAC and CBC) under assumptions that are specific to these schemes (and directed to the use of these extractors in the context of key derivation functions) including random-oracle type assumptions. On the other hand, the recent results of [BDK+11] show that for some key derivation applications one may relax the entropy requirements dictated by the RT bound (see more discussion on these issues in Section 7).

In this work, we further investigate computational extractors and consider questions of existence and inherent complexity from the theoretical and practical angles, with particular focus on the relationship to pseudorandomness. In particular, we ask how intrinsic is the use of pseudorandomness in constructing computational extractors, to what extent can we build computational extractors without resorting to a statistical extractor, and whether the "entropy penalty" of the extract-then-prg approach is avoidable.

Our Results

On the existence of computational extractors. The most basic question with respect to computational extractors is whether they exist at all and if they do under what (if any) assumption. The trivial answer is affirmative: statistical extractors are also computational. But we are interested in non-trivial computational extractors that output "more bits" than a statistical one. To capture this, we define the notion of *stretch*. For a security parameter p consider an extractor acting on a $k(p)$-entropy source: its stretch σ is the difference between the extractor's output length and its input's min-entropy, i.e., $\sigma(p) = m(p) - k(p) - \ell(p)$. Computational extractors with negative stretches of the form[1] $-\omega(\log p)$ exist unconditionally since a statistical extractor (that matches the RT bound) generates an output that is $2^{-\omega(\log p)}$-close to uniform and therefore is computationally indistinguishable from uniform. Thus, non-trivial computational extractors are those for which the stretch is at least $-O(\log p)$: we call such stretches and their associated extractors *proper*. The fact that proper computational extractors can be built on the basis of one-way functions via the extract-then-prg approach, raises the fundamental question: *Are one-way functions necessary for building proper computational extractors?* One would expect the answer to be "of course

[1] $\omega(\cdot)$ stands for any superlinear function (i.e., one that grows faster than any linear function of its argument).

they are!". However, we can only provide a partial answer: We can show this to be the case for proper extractors of *positive* stretch. But for stretches in the range between $-O(\log p)$ and 0 the question remains open. Interestingly, however, we can provide an affirmative answer under the assumption that *the RT bound applies to efficiently samplable distributions.* We refer to this as the *SRT Assumption* (see details in Section 3):

SAMPLABLE RT (SRT) ASSUMPTION. *Let* $Ext : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ *be a poly-time computable* statistical *extractor. Then, for* $k < n$ *there exists a* poly-time samplable *source* X *of min-entropy* k *such that the statistical distance between the distributions* $Ext(X, U_\ell)$ *and* U_m *is at least* $2^{-O(k+\ell-m)}$.

In other words, the SRT assumption strengthens the RT bound by requiring it to hold even if we restrict our attention to efficiently samplable distributions. On the other hand, it weakens the RT bound by only requiring it to hold for efficient extractors and by reducing the lower bound requirement to $2^{-c\cdot(k+\ell-m)}$ for *any* constant c (in the RT bound, $c = 1/2$). To the best of our knowledge, the validity of this assumption has not been settled. Interestingly, given our results, any resolution of the assumption will have significant consequences. Disproving the assumption would open the door to the possibility of more effective statistical extractors for applications that are only concerned with efficient sources; e.g., it would mean that extractors based on the Leftover Hash Lemma may not be the best in practice (a surprising conclusion that may actually indicate the plausibility that the SRT does hold). And if the SRT assumption does hold, then our work settles affirmatively the question of existential equivalence of proper computational extractors and one-way functions.

Black-box constructions of proper extractors from OWPs. After investigating the relationship between proper extractors and one-way functions, we examine the question of whether we can have black-box constructions of proper extractors from OWPs that are more efficient than going through the extract-then-prg approach. As the measure of efficiency we use "OWP-complexity", namely, the number of invocations to the OWP in a black-box construction, following [GGKT05]. We prove a lower bound on the OWP-complexity of black-box constructions of proper extractors from OWPs. We show that, under the SRT assumption, the OWP-complexity of the extract-then-prg construction is optimal by showing a *tight lower bound* on the number of invocations to the OWP for *any* black-box construction of proper extractors from OWPs. Interestingly, this result confirms the intuition that in order to build a proper computational extractor one needs to make up for the entropy gap intrinsic to the RT bound (as explained above).

The above result applies to *any* black-box construction of a proper extractor that has oracle access to a OWP and it puts no restriction on the security reduction (which efficiently transforms an extractor-attacker into a OWP-attacker). A more restricted form of black-box constructions, known as *fully black-box*, also requires that the reduction between attackers be black-box (i.e., the reduction cannot access the code of the extractor-attacker). Interestingly, we prove a similar bound for fully black-box constructions, but *unconditionally*, i.e. without a

need for the SRT assumption. Thus, we trade the more restricted form of black-box reduction for a lower bound that fully dispenses with the SRT assumption. (For a thorough treatment of the semi- and fully-black-box notions and their meanings and implications please refer to [RTV04].)

Constructions based on Stronger Primitives. Next, we investigate the possibility of avoiding the intrinsic entropy loss in the generic extract-then-prg construction by assuming stronger primitives as the basis for the construction.

Our first result in this direction shows that given an exponentially-hard OWP, one can build a proper computational extractor where the OWP is applied *directly* to the high-entropy source without having to go through an initial extraction phase, hence avoiding the need to compensate for the entropy gap of the extractor. In order to achieve this result we replace the standard extract-then-prg approach with a dual prg-then-extract scheme that exploits the exponential hardness of the OWP to build a PRG that uses as its seed the very input from the high-entropy source.[2]

A practical computational extractor based on wPRF. We show a very simple construction of computational extractors based on *weak pseudorandom functions* (i.e., PRFs whose output is indistinguishable from uniform by adversaries that only see values of the function computed on *random* independent inputs). For this we resort to a lemma by Pietrzak [Pie09] showing that weak PRFs retain some of their security even when the keys are chosen from an imperfect source. More specifically, [Pie09] shows that if the original keys are of length n but they are chosen from a source with min-entropy $k \leq n$ then their security degrades roughly by an (optimal) factor of $2^{-(n-k)}$. This allows us to construct (strong) computational extractors where the source distribution is used to sample a key for the PRF and the extractor's random seed is used as an input to the PRF. This results in a very practical construction of computational extractors that fully dispenses with statistical extractors and perfectly fits the needs of randomness extraction in the context of key derivation functions (KDF) as studied in [Kra10] and as extensively used in real-world applications. In particular, one obtains a very practical KDF for cases where the input to the KDF (the source of key material) is at most of the size of the wPRF key. The security of the scheme solely depends on the security of the underlying (weak) PRF and it implies meaningful security bounds even in constrained cases where the entropy-output gap is small (or even negative). See Section 7 for details.

Relations to work on statistical extractors. While a main theme of our work is the role of pseudorandom generators in the construction of *computational* extractors, it is interesting to point out that pseudorandomness also plays a fundamental role in the development of *statistical* extractors. Starting with the work of Trevisan [Tre01] it has been realized that constructions of "non-cryptographic" pseudorandom generators such as [NW88,IW97] can lead to efficient statistical

[2] This construction is somewhat reminiscent of the techniques used by Kalai *et al.* in [KLR09] for building two-source or network extractors, though the context and goals of these constructions are different.

extractors. The notion of pseudorandomness in these works is usually weaker than the traditional cryptographic notion (that we use in our definition of computational extractors), e.g., they allow for super-polynomial (on the seed length) running time or consider more limited adversaries. Also the focus on efficiency in statistical extractors has traditionally been geared towards minimizing the size of the random seed as this determines the utility of these extractors in derandomization applications. See [Sha02] for a survey of results in this area. It would be very interesting to find closer relations between results in the above area and the questions raised by our work. In particular, in spite of the large body of work on statistical extraction, there seems to be little work that investigates statistical extractors against (efficiently) samplable sources. The only paper on the subject that we are aware of is by Trevisan and Vadhan [TV00] who show that if we only care about samplable distributions we can use deterministic extractors; however, this only works as long as the sampler of the source is computationally weaker than the extractor itself. Indeed, [TV00] shows that if we allow the source to depend on the extractor and to have higher computational complexity then deterministic extraction is not possible. In terms of our SRT assumption, what this shows is that the SRT does apply to deterministic extractors (for each such extractor there is a samplable source where the extractor fails). For all we know, the seemingly fundamental question of the entropy bounds that apply to statistical extractors when acting on samplable sources has not been studied. We hope that our work will provide motivation to investigate this question.

2 Proper Computational Extractors

We recall the definitions of statistical extractors, define proper computational extractors and give some of their basic properties. All extractor definitions presented here are stated in an asymptotic setting; in Section 5 we provide definitions in a concrete-complexity framework.

2.1 Preliminaries

Terminology. A *probability ensemble* X is an infinite sequence of probability distributions $\{X_p\}$ indexed by a parameter p. We usually assume that for all p, X_p has support in $\{0,1\}^{n(p)}$ where $n(\cdot)$ is a polynomially bounded function. For any integer t we use the symbol U_t to denote the uniform distribution on $\{0,1\}^t$. The *statistical distance* between two probability ensembles X, Y with common support ensemble $\{0,1\}^{n(p)}$ is defined as the function $\Delta_{X,Y}(p) = \max_{T \subseteq \{0,1\}^{n(p)}} |\Pr[X_p \in T] - \Pr[Y_p \in T]|$. We say that a distribution X has *min-entropy* $k(p)$ if for all x in the support of X_p it holds that $Pr_{X_p}[x] \leq 2^{-k(p)}$. For simplicity, in what follows we assume that the entropies denoted $k(p)$ are positive integers (in case $k(p)$ is not an integer, our results hold by replacing it with $\lceil k(p) \rceil$).

Definition 1. *An* extractor family *(or simply* extractor*) is an infinite family $E = \{E_p\}$, indexed by a parameter p, of the form $E_p : \{0,1\}^{n(p)} \times \{0,1\}^{\ell(p)} \rightarrow$*

$\{0,1\}^{m(p)}$ *where the functions* $n(p), \ell(p), m(p)$*, are all polynomial in* p*. The extractor family* E *is called* $(k(p), \varepsilon(p))$-statistical *if for any probability ensemble* X *with support in* $\{0,1\}^{n(p)}$ *and min-entropy* $k(p)$*, it holds that the statistical distance between* $\langle U_{\ell(p)}, E_p(X_p, U_{\ell(p)})\rangle$ *and* $U_{\ell(p)+m(p)}$ *is at most* $\varepsilon(p)$*.*

The probability distribution from which the first input is taken is called the *source* and the second input is the *seed*. This definition of an extractor, requiring the joint distribution of output and seed to be ε statistically-close to uniform, is sometimes referred to in the literature as a *strong* extractor. A weaker flavor of this definition, referred to as a *weak* extractor, is one where one only considers the distance between the output $E_p(X_p, U_{\ell(p)})$ and the uniform distribution $U_{m(p)}$ (without the seed, which may remain hidden). *In this paper, unless otherwise noted, an "extractor" refers to a strong extractor.*

Intuitively, the goal of an extractor is to extract close-to-uniform bits out of a source with sufficiently high min-entropy, using a "short" uniformly random seed. We require that the output is longer than the seed,[3] specifically that $m(p) > \ell(p) + 1$.

Ideally, we'd like to extract all the randomness from the input, getting $m = k + \ell$ truly uniform bits (with $\varepsilon = 0$). However, this is impossible in general. From the results of [RTS00,NZ96,HILL99] we have the following lemma (which holds even for weak extractors) showing a tight relationship between how much of the input entropy $k + \ell$ can be extracted, and the distance ε from uniform.

Lemma 1 (RT Bound [RTS00]). *Let* E *be a* $(k(p), \varepsilon(p))$*-statistical extractor with parameters* $n(p), \ell(p), m(p)$ *where* $k(p) < n(p) - O(1)$[4] *and* $\varepsilon(p) < 1/2$*. Then* $\varepsilon(p) \geq 2^{-\frac{k(p)+\ell(p)-m(p)+O(1)}{2}}$*. That is, for every such* E *there is a probability ensemble* X *with min-entropy* $k(p)$ *for which* $E_p(X_p, U_{\ell(p)})$ *has statistical distance* $\min\{\frac{1}{2}, 2^{-\frac{k(p)+\ell(p)-m(p)}{2}}\}$ *from* $U_{m(p)}$*. This bound is tight and achieved, in particular, by statistical extractors implemented via pairwise independent hash functions.*

2.2 Proper Computational Extractors and Proper Stretch

We start by defining computational extractors, which differ from statistical ones in that the output is only required to be computationally indistinguishable from uniform rather than statistically close, the extractor itself needs to be efficient, and it is only required to work on efficiently samplable distributions.

Definition 2. *A family* E *of extractors is called* $k(p)$-computational *if* E_p *is polynomial-time computable, and for all efficiently-samplable probability ensembles* X *with min-entropy* $k(p)$*, the joint distribution* $(U_{\ell(p)}, E_p(X_p, U_{\ell(p)}))$ *is computationally indistinguishable from* $U_{\ell(p)+m(p)}$*.*

[3] Without this condition, the trivial extractor that outputs its seed works for any source (even with 0 entropy).

[4] The symbol $O(1)$ represents a specific constant calculated in [RTS00].

In this definition "efficiently samplable" means samplable by a polynomial-time algorithm and "computationally indistinguishable" refers to the regular notion of negligible advantage for all polynomial-time distinguishers. In a non-uniform setting, polynomial-time is be replaced by poly-size circuits.

Discussion: The defined notion corresponds to a strong extractor (see Section 2.1). A *weak* computational extractor is defined similarly but only requiring that the output $E_p(X_p, U_{\ell(p)}))$ (without the seed) is indistinguishable from uniform. Although *our lower bounds hold even for weak extractors,* we focus our treatment on strong extractors, because in the computational setting, weak extractors are not very interesting. Indeed, any PRF is, by definition, a weak computational extractor that works for *any* source distribution.

We require the output of the computational extractor to be pseudorandom only when the input is an efficiently samplable distribution. Indeed, for computational uses (where we model feasible computation as polynomial-time) a hard-to-sample distribution is of little interest. In particular, we would not want to disqualify a good computational extractor just because it fails on a hard to compute source. Also, samplable sources allow to use the same seed – as long as it has been chosen at random and independently of the source – with multiple samples (this is crucial in some applications, including key derivation as discussed in Section 7).

At the same time, it is worth noting that we could consider a flavor of our definition where efficient samplability is replaced with oracle access (for the attacker) to an arbitrary distribution. The lower-bound results from Sections 3 and 4 hold for this definition, while the upper bound from Lemma 7 holds as long as the OWP is secure against non-uniform attackers (non-uniformity is necessary to argue that access to a hard-to-compute distribution does not help the attacker break the OWP or other primitives such as a PRG). Finally, we note that for our results on fully black-box reductions from Section 5, we do consider the latter setting, namely, arbitrary distributions to which the attacker gets oracle access.

It is clear that any efficient $(k(p), \varepsilon(p))$-statistical extractor for a negligible $\varepsilon(p)$, is also a $k(p)$-computational extractor. Thus, the upper bound of Lemma 1 implies the following.

Lemma 2. *There exist extractors with parameters $n(p), \ell(p), m(p)$ that are $k(p)$-computational for any $k(p) < n(p) - O(1)$ such that*

$$k(p) = m(p) - \ell(p) + \omega(\log p) \tag{1}$$

Note that the Lemma is unconditional, i.e., computational extractors with parameters as in (1) exist unconditionally. In this sense, *non-trivial* computational extractors are those whose parameters beat (1), and in particular have an output that is (indistinguishable from but) statistically far from uniform. We call such extractors *proper*, defined as follows.

Definition 3. *The* stretch $\sigma(p)$ *of a $k(p)$-computational extractor with parameters $n(p), \ell(p), m(p)$is defined as $\sigma(p) = m(p) - k(p) - \ell(p)$. The stretch $\sigma(p)$*

is proper *if* $\sigma(p) \geq -O(\log p)$ *(i.e., there exists a constant* c *such that* $\sigma(p) \geq -c \log p$ *for all* p*). A* $k(p)$*-computational extractor is* proper *if its stretch is proper.*

Note that the stretch does not only depend on the extractor but also on the input entropy $k(p)$ (though, for simplicity, we sometimes omit the explicit $k(p)$ notation when talking about proper extractors). Since, for simplicity, we have assumed that $k(p)$ is integer (or else we consider $\lceil k(p) \rceil$) then the stretch is integer and can be negative, zero, or positive. Hereafter, when we say "proper extractor" we mean "proper computational extractor."

3 The Equivalence of Proper Extractors and One-Way Functions

Note that statistical extractors have statistical distance from uniform of at least $\varepsilon(p) = 2^{-\frac{k(p)+\ell(p)-m(p)}{2}}$ which is $1/poly(p)$ (hence non-negligible) in the case of proper extractors. Thus, statistical extractors do not immediately yield proper computational extractors.

This raises the question: *Do proper computational extractors exist?* The following Lemma answers this in the affirmative, assuming one-way functions exist.

Lemma 3. *If one-way functions exist then strong proper computational extractors exist too.*

Proof sketch: Let $E = \{E_p\}$ be a $k(p)$-computational extractor with parameters $n(p), \ell(p), m(p)$ for which equation (1) holds (such an extractor exists for any functions $m(p), k(p)$ as in Lemma 2). Also assume $\omega(\log p) \leq p$. Let $\{G_p\}$ be a pseudorandom generator with seed length $m(p)$ and output length $k(p)+\ell(p)$ (assuming OWFs, PRGs exist for some function $m(p)$ and output length $m(p)+p$). Construct extractor E' that first applies E and uses the output to seed the PRG. It is easy to see that E has parameters $n(p), \ell(p), m'(p) = k(p) + \ell(p)$ and its output is indistinguishable from $U_{m'(p)}$. But $m'(p) = k(p) + \ell(p)$, thus E' is proper.

Somewhat surprisingly we can't immediately prove *equivalence* between proper extractors and one-way functions. The opposite direction of Lemma 3 can be easily proven only for proper computational extractors with *positive stretch* as shown in the following Lemma.

Lemma 4. *From any (even weak) computational extractor with positive stretch one can build a pseudorandom generator.*

Proof sketch: Let E be a $k(p)$-computational extractor with parameters $n(p)$, $\ell(p)$, $m(p)$ and positive stretch $\sigma(p)$, i.e. $m(p) > k(p) + \ell(p)$. We build a PRG G with random seeds of length $s(p) = k(p) + \ell(p)$ and output length $m(p) > s(p)$. G partitions its seed into a $k(p)$-long value x and an $\ell(p)$-long value y, and calls E on (x', y) where x' consists of x padded with $n(p) - k(p)$ zeros. Clearly, the

input distribution to E has entropy $k(p)$, hence its output is pseudorandom. Since G outputs more bits than its seed then G is a pseudorandom generator.

The last two lemmas leave the following question: *Does the existence of proper computational extractors, even those with* non-positive *proper stretch imply the existence of one-way functions? In particular, is this the case for computational extractors of stretch 0?* To provide an affirmative answer we need to resort to an additional assumption about the RT bound.

Samplable RT (SRT) Assumption. *For every* polynomial-time computable *extractor E with parameters $n(p), \ell(p), m(p)$ and every function k such that $k(p) < n(p) - O(1)$, there exists a* poly-time samplable *probability ensemble X of min-entropy k such that the statistical distance between the distributions $E_p(X_p, U_{\ell(p)})$ and $U_{m(p)}$ is at least* $\min\{\frac{1}{2}, 2^{-O(k(p)+\ell(p)-m(p))}\}$.

In other words, we are assuming that if we restrict attention to efficiently samplable sources then the RT bound still applies. More accurately, we assume a weaker bound where the RT bound $2^{-\frac{1}{2}(k(p)+\ell(p)-m(p))}$ is replaced with $2^{-c\cdot(k(p)+\ell(p)-m(p))}$ for *any* constant c, possibly much larger than $1/2$. In addition, we assume this to be the case only for efficient extractors[5]. This assumption is not implied by the proof in [RTS00] which builds a source on which the extractor incurs the claimed bound but this source may not be efficiently samplable. Quite interestingly, the question raised by this conjecture does not seem to have been widely researched. Any answer to it, positive or negative, would be of interest. If true it implies the equivalence of proper computational extractors and pseudorandom generators (see Theorem 1). If disproven it would open the possibility of building efficient extractors that beat the RT and Leftover-Hash-Lemma bounds on efficient sources.

Lemma 5. *Under the SRT assumption, the existence of a proper extractor implies the existence of a OWF.*

Proof sketch: Let E be a proper $k(p)$-computational extractor and let X be a polynomial-time samplable ensemble of min-entropy $k(p)$, then the output of E on X induces a polynomial-time samplable distribution that is statistically far from uniform but computationally indistinguishable. Thus, the pair of distributions $(E_P(X_P, U_{\ell(p)}), U_{m(p)})$ are efficiently samplable, have statistical distance greater than $1/\text{poly}(p)$ for some polynomial and are computationally indistinguishable. Using the results of [Gol90,HILL99], constructing such a pair of distributions is sufficient to construct pseudorandom generators (PRG). This in turn implies the existence of OWF.

From Lemmas 3 and 5 we get:

Theorem 1. *Under the SRT assumption, proper computational extractors exist if and only if one-way functions exist.*

[5] It is most likely (using a counting argument) that the conjecture does not hold for super-polynomial extractors, namely, there may be inefficient extractors that beat the RT bound on all efficiently samplable distributions.

4 The Cost of Black-Box Constructions of Proper Extractors from OWPs

In this section we follow the methodology from [GGKT05] for quantifying the cost, as a number of OWP invocations, of (semi) black-box constructions of proper computational extractors from OWPs. We show a lower bound on the number of calls to the OWP that depends on the strength of the OWP and the stretch of the extractor. This result reflects the intuition that in order to build a computational extractor one needs to first make up for the entropy gap intrinsic to the RT bound. Indeed, the result shows that it is not enough to call the OWP just to generate as many bits as the extractor's stretch but one needs to generate $\omega(\log p)$ additional bits to cover for the loss of entropy. Comparing with the corresponding results of [GGKT05] about pseudorandom generators, we see that making up for this entropy gap is the only intrinsic difference between proper extractors and PRGs (under this black-box complexity measure). We also prove that the lower bound is tight.

Remark: Our lower bounds deal with constructions of computational extractors from one-way permutations. However, we note that our results extend to the case of one-way functions since our lower bounds are proven using random permutations which are not efficiently distinguishable from one-way functions. However we do not know if for the case of OWF our bounds are tight (i.e. the currently known constructions based on OWF have a larger number of queries).

In Section 3 we showed that proper extractors are equivalent to one-way functions. Here we formalize a notion of *black-box* constructions for computational extractors: such constructions access a one-way function as an *oracle*, rather than having access to the code of an algorithm computing it.

We start by developing an analogue of the treatment from [GGKT05] to the asymptotic setting of our analysis. For any integers t, n, $t \leq n$, we denote by Π_n the set of all permutations over $\{0,1\}^n$ and by $\Pi_{t,n}$ the set of permutations in Π_n that arbitrarily permute the first t bits of input while leaving the remaining $n-t$ bits fixed.

For a security parameter p denote with $n(p), k(p), \ell(p), m(p)$ and $t(p)$ integer functions that grow polynomially in p. Assume also that $t(p) \leq n(p)$ and $k(p) \leq n(p) - O(1)$ for all p. Consider an infinite family of permutations $\Pi = \{\pi_p\}_{p=1}^{\infty}$ where π_p is chosen in $\Pi_{n(p)}$. We say that Π is $T(p)$-hard if for sufficiently large p, any attacker running in time $T(p)$ succeeds in inverting π_p with probability less than $1/T(p)$. We say that Π is *one-way* if it is $T(p)$-hard for every polynomial $T(\cdot)$.

With Π^* we denote such a family $\Pi^* = \{\pi_p^*\}_{p=1}^{\infty}$ where each permutation π_p^* is chosen at random from the set $\Pi_{t(p),n(p)}$. The following Lemma (based on [IR89]) proves that for any hardness $T(p)$, if we choose $t(p) = 3\log T(p)$ (and an additional technical condition that $t(p) \geq 6\log p$), then this family is $T(p)$-hard with probability 1.

Lemma 6. *Let $t(p) \geq 6\log p$. Then with probability 1, Π^* constructed as above is $T(p)$-hard for $T(p) = 2^{t(p)/3}$.*

Proof. Let A be an adversary that runs time $T(p)$ and attempts to invert Π^*. On expectation, over the choice of π_p^*, A succeeds in inverting with probability $T(p)/2^{t(p)} = 1/T^2(p)$, namely:

$$E_{\pi \sim \Pi_{t(p),n(p)}}[\Pr_{x \sim U_{n(p)}}[A(\pi(x)) = x]] = 1/T^2(p).$$

Using Markov's inequality we have that the probability over the choice of π_p^* that A inverts successfully with probability better than $T(p) \cdot 1/T^2(p)$ is at most $1/T(p)$:

$$\Pr_{\pi \sim \Pi_{t(p),n(p)}}[\Pr_{x \sim U_{n(p)}}[A(\pi(x)) = x] \geq T(p) \cdot 1/T^2(p)] \leq 1/T(p). \tag{2}$$

Since by choice of $t(p) \geq 6 \log p$ we have $1/T(p) \leq 1/p^2$ we get that the sum $\sum_{p \to \infty} 1/T(p)$ is finite. The convergence of this sum allows us to apply the Borel-Cantelli Lemma to (2) which implies that with probability 1 over the choice of Π^* the inequality $\Pr_{x \sim U_{n(p)}}[A(\pi_p^*(x)) = x] < 1/T(p)$ (where A is assumed to run time $T(p)$) holds for all but a finite number of p's. In other words, with probability 1 over the choice of Π^*, the resultant family Π^* is $T(p)$-hard.

Definition 4. *An* oracle extractor *construction (from a one-way permutation) is a family of oracle procedures* $\mathcal{E}^{(\cdot)} = \{E_p^{(\cdot)} : \{0,1\}^{n(p)} \times \{0,1\}^{\ell(p)} \to \{0,1\}^{m(p)}\}$ *such that* $E_p^{(\cdot)}$ *expects as an oracle a permutation* $\pi_p \in \Pi_{n(p)}$ *and* $E_p^{(\cdot)}$ *is computable in time polynomial in* p. *We say that* $\mathcal{E}^{(\cdot)}$ *has black-box access to a family* $\Pi = \{\pi_p\}_{p=1}^{\infty}$ *(and denote it as* $\mathcal{E}^{(\Pi)}$*) if* $E_p^{(\cdot)}$ *uses* $\pi_p \in \Pi$ *as its oracle.*

We say that $\mathcal{E}^{(\cdot)}$ *is a* $k(p)$-computational oracle extractor *if for every one-way family* Π *the family* $\mathcal{E}^{(\Pi)}$ *is a* $k(p)$*-computational extractor according to Definition 2*

Another way to restate the above definition is that there must be an efficient reduction from distinguishing the output of the extractor from uniform to inverting the permutation family. In other words, any distinguishing adversary can be used to construct an inverter for the permutation family. Note that the above definition formalizes the notion of *semi black-box* construction in which the construction (the extractor) has oracle access to the underlying primitive (the one-way permutation), but no restriction is made on the reduction (in particular, the reduction might be able to access the code of the adversary). The more restricted notion of fully black-box constructions (in which additionally the security reduction only has oracle access to the adversary breaking the construction) will be discussed in Section 5.

We now state the main theorem in this section. It shows that under the SRT assumption, proving a semi-black-box construction of a computational extractor for which $q(p) \cdot t(p) - \sigma(p) = O(\log p)$ is at least as hard as proving that OWFs exist (or, equivalently, proving such a construction is at least as hard as proving that the SRT assumption implies OWF).

Theorem 2. *Let $\mathcal{E}^{(\cdot)}$ be a proper $k(p)$-computational oracle extractor according to Definition 4, which has access to a $T(p)$-hard family where $T(p)$ is super-polynomial. Let $t(p) = 3\log T(p) = \omega(\log p)$. Assuming SRT, if $E_p^{\pi_p}$ has proper stretch $\sigma(p)$ and it calls the oracle π_p a total of $q(p)$ times, then $q(p)\cdot t(p) - \sigma(p) = \omega(\log p)$ or else one-way functions exist. This lower bound on $q(p)$ is tight.*

Proof. Let $\mathcal{E}^{(\cdot)}$ be a proper $k(p)$-computational oracle extractor with parameters $(n(p), \ell(p), m(p))$ and proper stretch $\sigma(p) = m(p) - k(p) - \ell(p)$. By assumption $\mathcal{E}^{\Pi}$ is $k(p)$-computational whenever the oracle Π is implemented with one-way permutation family, i.e. Π is $T(p)$-hard where $T(p)$ is a function growing faster than any polynomial. In particular, by Lemma 6 this is the case (with probability 1) when Π is implemented by the family Π^* with parameter $t(p) = 3\log T(p) = \omega(\log p)$. We will show that if $E_p^{\pi_p}$ calls $\pi_p \in \Pi^*$ a total of $q(p)$ times, we can construct a computational extractor E'_p with parameters $(n(p), \ell'(p) = \ell(p) + q(p)t(p), m(p))$ (and no oracle calls) such that for any distribution X_p with min-entropy $k(p)$, the output distributions $E'_p(X_p, U_{\ell'(p)})$ and $E_p^{\pi_p^*}(X_p, U_{\ell(p)})$ are $q^2(p)/2^{t(p)}$-statistically close, and since the latter distribution is pseudorandom so is the former (here we use the fact that $q(p)/2^{t(p)}$ is negligible since $q(p)$ is polynomial and $2^{t(p)} = T^3(p)$ super-polynomial).

More specifically, we construct $E'_p : \{0,1\}^{n(p)} \times \{0,1\}^{\ell'(p)} \to \{0,1\}^{m(p)}$, where $\ell'(p) = \ell(p) + q(p) \cdot t(p)$, in the following way: Let x, z' denote the input to E'_p. The string x and the first $\ell(p)$ bits of z' are used by E'_p to define the input (x, z) to $E_p^{(\cdot)}$ and the remaining bits of z' are used to select $q(p)$ distinct elements $y_1, \ldots, y_{q(p)} \in \{0,1\}^{t(p)}$. We then define: $E'_p(x, z, y_1, \ldots, y_{q(p)}) \stackrel{def}{=} E_p^{y_1,\ldots,y_{q(p)}}(x, z)$, namely, when $E_p^{(\cdot)}$ presents its i-th query to its oracle, call it w_i, we return as response the string y_i followed by the last $n(p) - t(p)$ bits of w_i.

Note that as long as all the y_i's are different the output distributions $E'_p(X_p, U_{\ell'(p)})$ and $E_p^{\pi_p^*}(X_p, U_{\ell(p)})$ are identical. The probability of a repeated y_i is $q^2(p)/2^{t(p)}$ and therefore the actual statistical distance between these distributions is negligible. In particular, we have that the output from E'_p is indistinguishable from random and therefore E'_p is a $k(p)$-computational extractor which makes no oracle calls. Moreover, its stretch $\sigma'(p)$ equals

$$\sigma'(p) = m(p) - k(p) - \ell'(p) = m(p) - k(p) - \ell(p) - q(p)t(p) = \sigma(p) - q(p)t(p).$$

If, for the sake of contradiction, we assume that $q(p)t(p) \leq \sigma(p) + c\log p$ for some constant c then we would get $\sigma'(p) \geq -c\log p$ meaning that E'_p is a regular (non-oracle) proper computational extractor from which, using Lemma 5 and the SRT assumption, we can construct a one-way function. This proves the theorem (the tightness of the bound on $q(p)$ is proven in Lemma 7 below).

Lemma 7. *The bound of Theorem 2 is tight: For any function $\sigma(p)$, polynomial in p, and any function $W(p)$ that grows as $\omega(\log p)$ there is a black-box construction of a strong proper extractor from OWP that attains stretch $\sigma(p)$ and calls the OWP $q(p)$ times such that $q(p)t(p) \leq \sigma(p) + W(p)$.*

Proof sketch: We start by noting that there are black-box constructions of pseudorandom generators from OWPs that for any PRG-stretch function $\sigma'(p)$ (defined as the length of the PRG output less the length of PRG seed) call the OWP $\sigma'(p)/t(p)$ times where $t(p)$ is defined as in Theorem 2. This is the case, in particular, for the Blum-Micali construction using Goldreich-Levin hard-core bits. Therefore, to prove the Lemma it suffices to show how to build a proper extractor of stretch $\sigma(p)$ using a PRG of stretch $\sigma(p) + W(p)$ for any $W(p)$ that grow as $\omega(\log p)$.

Let $G = \{G_p\}_p$ be a PRG family, indexed by a parameter p, with seed length $s(p)$ and output size $r(p) = s(p) + \sigma(p) + W(p)$, for a given (polynomial in p) function $\sigma(p)$. Assume G is $(T(p), \varepsilon(p))$-secure (where $\varepsilon(p)$ is negligible in p). Let E be a strong statistical extractor (e.g., based on pairwise independent hash functions) with parameters $n(p), \ell(p), m(p) = s(p) + \ell(p)$ that on input distributions of min-entropy $k(p)$ outputs a distribution that is $2^{-\frac{k(p)-s(p)}{2}}$-close to $U_{m(p)}$. Using both G and E we build a proper computational extractor E' with parameters $n(p), \ell(p), m'(p) = r(p) + \ell(p)$. On input (x, z), E' calls E on (x, z) and uses the $s(p)$-bit output from E as the seed to G to produce an output of bit length $r(p) = s(p) + \sigma(p) + W(p)$. This, plus the $\ell(p)$-bit input salt, are the outputs from E'.

Note that on distributions of min-entropy $k(p) = r(p) - \sigma(p)$, E' has stretch $\sigma(p)$; moreover, we claim that the output from E' is $(T(p), \varepsilon'(p))$-indistinguishable from uniform where $\varepsilon'(p)$ equals $\varepsilon(p)$ plus a negligible term $2^{-W(p)/2} = 2^{-\omega(\log p)}$. Indeed, the only loss of security with respect to G is in the derivation of the seed $z \in \{0,1\}^{s(p)}$ that is chosen from a distribution that is $2^{-(k(p)-s(p))/2} = 2^{-W(p)/2} = 2^{-\omega(\log p)}$-close to $U_{s(p)}$. Thus E' is a proper computational extractor with stretch $\sigma(p)$ built on the basis of a PRG of stretch $\sigma(p) + W(p)$ which, as said, implies the tightness of the bound.

Note. The Blum-Micali construction with a randomized hardcore like Goldreich-Levin [GL], requires extra perfect but non-secret randomness. Hence this auxiliary randomness can be supplied by the extractor's seed and be output as part of the strong extractor's output.

5 Unconditional Fully Black-Box Lower Bound

Next, we pose the question of what can be shown *without assuming SRT*. We show that by restricting our attention to fully black box constructions, not only can we get rid of the SRT but actually can show an *unconditional* lower bound on the number of OWP invocations.

We first show an analogous lower bound to the semi black-box case (though unconditional) in the asymptotic, uniform setting. We then show a tighter concrete-complexity result in the non-uniform setting.

To begin, we review the notion of fully black box construction/reduction.

Definition 5. *A* fully black-box reduction *from a primitive Q to a primitive P is a pair of* oracle *PPT Turing machines $(G^{(\cdot)}, S^{(\cdot,\cdot)})$ such that the following two properties hold:*

Correctness: For every implementation f of primitive P, $g = G^f$ implements Q.

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

Notice that in a full black-box reduction, the adversary is only accessed as an oracle. One consequence of this fact is that the adversary does not have to be efficient. We remark that an *implementation* of a primitive is any specific scheme that meets the requirements of that primitive (e.g., an implementation of a public-key encryption scheme provides samplability of key pairs, encryption with the public-key, and decryption with the private key).

5.1 Unconditional Lower Bound in the Asymptotic, Uniform Setting

In this section we show an analogue of the lower bound in Theorem 2 for the fully black-box setting. While the bound on the number of queries is the same as in Theorem 2, this result can be proven unconditionally (i.e., without requiring the SRT and without concluding that a construction that violates the bound implies a proof of the existence of one-way functions). However, Theorem 3 holds only when we consider a slightly modified definition of computational extractors where the output of the extractor is required to be computationally indistinguishable from uniform for *every* input probability ensemble X of min-entropy k. Observe that the construction outlined in Lemma 7 satisfies this stronger notion of security.

Theorem 3. *Let $\mathcal{E}^{(\cdot)}$ be a proper $k(p)$-computational fully black box extractor construction, which has access to a $T(p)$-hard family where $T(p)$ is super-polynomial. Further assume that such extractor remains proper $k(p)$-computational on* any *$k(p)$-entropy source, including those that are not efficiently samplable. Let $t(p) = 3 \log T(p) = \omega(\log p)$. If $E_p^{\pi_p}$ has proper stretch $\sigma(p)$ and it calls the oracle π_p a total of $q(p)$ times, then $q(p) \cdot t(p) - \sigma(p) = \omega(\log p)$.*

Proof. See full version [DGKM11].

Next, we present a stronger version of this result. It will be a tighter concrete (rather than asymptotic) lower bound, for *non-uniform* fully black-box constructions of proper extractors from OWP. In order to do that, we need to revisit definitions and preliminary Lemmas in a concrete, non-uniform context.

5.2 Unconditional Lower Bounds in the Concrete, Non-Uniform Setting

We start by adapting the definition of (oracle) computational extractors to the non-uniform and concrete (i.e., non-asymptotic) complexity setting.

We say that a permutation π over $\{0,1\}^n$ is S-hard if no circuit of size $\leq S$ and oracle access to π can invert π with probability better than $1/S$. Additionally, we say that two distributions are (S,ε)-indistinguishable if no circuit of size $\leq S$ can distinguish between them with probability better than ε.

Definition 6. *$E : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ is a $(k, S, 2^{-\delta})$-*computational extractor (CompEXT) *if for any distribution X on $\{0,1\}^n$ with $H_\infty(X) \geq k$, we have that $(E(X, U_\ell), U_\ell)$ and $U_{m+\ell}$ are $(S, 2^{-\delta})$-indistinguishable (where indistinguishability holds even for circuits given oracle access to a* Sampler *which samples from distribution X).*

Definition 7. *An* oracle computational extractor *(*OCompEXT*) construction (from a one-way permutation) is an oracle procedure $E^{(\cdot)} : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ that expects as an oracle a permutation $\pi \in \Pi_n$. We are interested in constructions where $E^{(\cdot)}$ is computable in time polynomial in n.*

*We say that $E^{(\cdot)}$ is is an $(k, S_\pi, S_E, 2^{-\delta})$-*OCompEXT construction from OWP *if for every permutation π that is S_π-hard, E^π is an $(k, S_E, 2^{-\delta})$-secure* CompEXT *(where indistinguishability holds even for circuits given oracle access to both* Sampler *and π).*

Using a standard averaging argument, the existence of a non-uniform attacker that succeeds in inverting a OWP with the help of such an oracle implies the existence of another attacker (of slightly larger size) that inverts the OWP without access to the oracle (just wire-in into the attacker circuit the source samples that maximize the attacker's inverting probability).

We now restate the lower bound of Radhakrishnan and Ta-Shma [RTS00] regarding the efficiency of statistical extractors (which was given in Lemma 1 for the asymptotic, uniform setting).

Lemma 8. *Let $E' : \{0,1\}^n \times \{0,1\}^{\ell'} \to \{0,1\}^m$ be a statistical extractor. Then, for any $k < n - C'$ there exists a distribution X of min-entropy k such that the two distributions $E'(X, U_{\ell'})$ and U_m are statistically $\min\{\frac{1}{2}, 2^{-((k+\ell'-m+C)/2)}\}$-far, where C and C' are universal constants.*

We are now ready to state our main result in this section, namely, a lower bound on the number of queries to the OWP by a fully black box construction of a computational extractor.

Theorem 4. *Let $E^{(\cdot)}$ be a fully black-box construction of a $(k, S_\pi, S_E, 2^{-\delta})$ proper oracle extractor which expects an S_π-hard one-way permutation π over n bits. Assume that $E^{(\cdot)}$ makes $q \leq S_\pi$ queries to its oracle and that $S_E \leq S_\pi$ and $2^{-\delta} \geq 1/S_\pi$. If $E^{(\cdot)}$ has proper stretch σ then $E^{(\cdot)}$ must call the one-way permutation q times, where $q \geq (2\delta + \sigma - C)/(5 \log S_\pi)$ for some constant C.*

Proof. See full version [DGKM11].

6 Construction from Exponentially-Hard One-Way Permutations

The results from Sections 4 and 5 indicate the optimality of the "extract-then-prg" approach when all we are interested in is minimizing the number of calls to a OWP in a black-box construction. However, a significant cost of this approach is that in order to use an n-bit OWP we need to start with an input distribution whose entropy is noticeably larger than n so we can apply the extraction part of the construction to it and still get n bits that are close to uniform and serve as input to the OWP. Here we show that one can make up for the entropy gap if the OWP has exponential hardness. In this case, we show a black-box construction based on such a OWP where one applies the OWP directly on the entropy source without an intermediate extractor step. For this we reverse the extract-then-prg approach and use instead an "prg-then-extract" construction where the OWP is applied first to expand (pseudo) entropy and then a statistical extractor is applied on this expanded entropy to generate a close-to-uniform output.[6]

The Construction. Given an $(S_\pi, 2^{-\delta}/2^{n-k})$-hard OWP, π, we present a construction of a k-entropy strong computational extractor

$$F : \{0,1\}^n \times \{0,1\}^{(2\delta+\sigma)\cdot n+\ell} \to \{0,1\}^{k+(2\delta+\sigma)\cdot n+\ell+\sigma}$$

with proper stretch σ in Figure 1.

On input $(x, z' = (r_0, \ldots, r_{2\delta+\sigma-1}, z))$, where $x \in \{0,1\}^n$, $r_i \in \{0,1\}^n$, $z \in \{0,1\}^\ell$, the extractor F does the following:

Step 1:
- Compute $(w_1, w_2) = \left(\left(\pi^{2\delta+\sigma}(x), \langle r_{2\delta+\sigma-1}, \pi^{2\delta+\sigma-1}(x)\rangle, \ldots, \langle r_0, x\rangle\right), (r_{2\delta+\sigma-1}, \ldots, r_0)\right)$

Step 2:
- Let $F' : \{0,1\}^{n+2\delta+\sigma} \times \{0,1\}^\ell \to \{0,1\}^{k+\ell+\sigma}$ be a statistical $(k+2\delta+\sigma, 2^{-\delta})$ strong extractor.
- Compute $(v, z) = F'(w_1, z)$.

Step 3: F outputs $(v, z, w_2) \in \{0,1\}^{k+(2\delta+\sigma)\cdot n+\ell+\sigma}$.

Fig. 1. Strong Computational Extractor from Exponentially-Hard OWP

The proof of Lemma 10 that F is indeed a strong extractor when π is an exponentially-hard OWP is based on the following lemma showing that exponentially-hard OWP's are "hard to invert" on arbitrary distributions of sufficiently high min-entropy.

[6] [BDK+11] also uses the prg-then-extract approach for constructing an extractor; in their case, however, the prg is used to expand the seed rather than for increasing the computational entropy of the source as in our case.

Lemma 9. *Let $\pi : \{0,1\}^n \to \{0,1\}^n$ be an (S, ε)-one way permutation and let X be a distribution over $\{0,1\}^n$ of min-entropy k where $k = n - \alpha$. Then for all adversaries A of size at most S it is the case that:*

$$\Pr_{x \sim X}[A(\pi(x)) = x] \leq \varepsilon \cdot 2^{\alpha}.$$

Lemma 10. *The construction of F from Figure 1 is a black-box construction of a $(k, S_\pi/\mathrm{poly}(n), \mathrm{poly}(n) \cdot 2^{-O(\delta)})$-strong* CompEXT *with proper stretch σ from any $(S_\pi, 2^{-\delta}/2^{n-k})$-*OWP *$\pi$.*

Proof. See full version [DGKM11].

7 Practical Computational Extractors from Weak PRF

In this section we explore a connection between computational extractors and pseudo-random functions. We show a very efficient construction of a strong computational extractor using *any* PRF, and demonstrate its practical utility in the context of key derivation functions. Actually, we do not need the full security of a PRF; it suffices that the PRF is secure against attackers that do not choose inputs to the function but only see pairs of (input, output) where the inputs are chosen uniformly at random. Such PRFs are referred to as *weak PRF (wPRF)* (note that in our application 'weak' is stronger). The proof of our scheme follows directly from recent results by Pietrzak [Pie09] about *leakage-resilient* wPRFs.

Weak PRF. A pseudo-random function family is a family of functions $\mathcal{F} = \{f_a : \{0,1\}^\ell \to \{0,1\}^m\}_{a \in \{0,1\}^n}$ with the property that if a is chosen uniformly at random in $\{0,1\}^n$, then the function f_a is computationally indistinguishable from a random function from $\{0,1\}^\ell$ to $\{0,1\}^m$. More specifically, no efficient algorithm which has oracle access to either f_a or to a random function, can decide which is the case. If the oracle access is restricted to query the function on randomly chosen inputs, one obtains the notion of weak PRF (wPRF). We quantify this notion by saying that $\mathcal{F}$ is a (S, q, ε)-wPRF family if no circuit of size S can distinguish between f_a (for a chosen uniformly at random) and a random function with advantage better than ε when seeing the value of the function on q *random* inputs.

The main contribution in this section is in presenting the following construction of a simple computational extractor from any wPRF and demonstrating its *practical* security.

wPRF-based computational extractor. Let $\mathcal{F} = \{f_a : \{0,1\}^\ell \to \{0,1\}^m\}_{a \in \{0,1\}^n}$ be a wPRF family. We define the extractor $F : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^{m+\ell}$ as $F(a, s) = (f_a(s), s)$.

Theorem 5. *If $\mathcal{F} = \{f_a : \{0,1\}^\ell \to \{0,1\}^m\}_{a\in\{0,1\}^n}$ is (S, q, ε)-weak PRF with $q^2 < \varepsilon 2^{\ell+1}$, then for $k \leq n$ the extractor F defined above is a (k, S', ε') strong (and proper[7]) computational extractor with $\varepsilon' \approx \varepsilon \cdot 2^{n-k}$ and $S' \approx S \cdot \varepsilon'$.*

Proof. See full version [DGKM11].

7.1 Application to Key Derivation

A main application of a strong computational extractor in cryptography is for key derivation [Kra10]. In this case, the source distribution is some *key material*, derived from some statistical process or a key agreement protocol, that has some significant amount of min-entropy but is not uniformly random as needed to key cryptographic functions. Thus, we need a way to produce a cryptographic key (random or pseudorandom) out of this key material. This is where our computational extractor is useful. One *restriction* is that if we use a wPRF whose key size is n we need to consider sources of key material whose length is at most n. In this case, we simply use the key material (without any processing, except maybe for padding to n bits) as the key to the wPRF and choose as the input to the wPRF a random value of length ℓ. The latter is the seed of the extractor and is assumed that the application provides such random but public "salt" (see [Kra10] for discussions on this issue). Next we show concrete examples of the applicability of this method when the key material is derived from a Diffie-Hellman value (as is common in the settings of key exchange and ElGamal encryption).

We are given (S, q, ε)-wPRF and consider the ratio S/ε as its measure of security (here S is a function of ε and q). Assume that the wPRF has full security, i.e. for a key of size n we have $S/\varepsilon \approx 2^n$. In this case, Theorem 5 guarantees that the extractor F (the KDF in our application) has parameters (S', ε') such that:

$$\varepsilon' \approx \varepsilon \cdot 2^{n-k} \text{ and } S' \approx S \cdot \varepsilon' = 2^n \cdot \varepsilon \cdot \varepsilon' \approx 2^k \varepsilon'^2$$

As a concrete example, consider the case of a wPRF with a 256-bit key and security $S/\varepsilon = 2^{256}$ (this would apply, given current knowledge, to a PRF based on SHA-256, especially that we only consider attacks where the attacker cannot chose any inputs – it only sees the function applied to a set of random values). Assume now that the key, instead of being sampled uniformly at random, follows a distribution with min-entropy $k = 160$; this is the case, for example, when the key material is a Diffie-Hellman value computed over an elliptic curve of size 2^{160} [GKR04]. In this case we have that to distinguish the 256-bit output of the extractor from random with advantage $\varepsilon' \approx 2^{-40}$ we must invest $S' \approx 2^{80}$. If we want to double the advantage ε' we need to invest four times more work (circuit size). For example, to obtain $\varepsilon' = 2^{-20}$ we need to work $S' = 2^{120}$ and for $\varepsilon' = 0.001$ one needs $S' = 2^{140}$. Even if we consider a less-perfect function,

[7] We assume $m \geq n$; if this is not the case in the given family $\mathcal{F}$ we can achieve it using standard range expansion techniques to increase m, possibly at the cost of somewhat strengthening the weak PRF requirement.

say $S/\varepsilon = 2^{200}$ one still gets $S' = 2^{64}$ for $\varepsilon' = 2^{-20}$ and $S' = 2^{84}$ for $\varepsilon' = 0.001$. Note that in all these cases we are outputting more pseudorandom bits (256) than the source entropy (160).

In comparison, if we were applying a *statistical* extractor to the key material of min-entropy 160 to obtain a key of size 256, we could not claim any security at all (this is the case even if we only needed a 160-bit of output, and we would get security of only 2^{-16} if were outputting a 128-bit key). In comparing with statistical extractors another main advantage of our PRF-based computational extractor is the fact that PRFs are already available in practical cryptographic protocols for other uses (including key expansion as often needed in the context of key derivation) and hence do not require of additional mechanisms such as a statistical extractor.

Related Schemes. It is worth noting the *duality* between the above KDF construction and the HKDF scheme from [Kra10]. In our case, the imperfect key material is used to key the (weak) PRF and the seed is used as an input to the KDF. In HKDF these roles are reversed. This gives HKDF the advantage of being appropriate for input distributions of arbitrary length while in our scheme we are limited to the key size. On the other hand, the very non-standard use of a *known value* (the seed) as a key to a PRF in the HKDF scheme, makes the latter much more restricted on the type of PRFs one can use (actually, the known analysis of HKDF is for particular PRFs, mainly HMAC, and under dedicated assumptions). In contrast, our scheme can use any PRF and even any wPRF.

The recent work of Barak et al. [BDK+11] builds a computational extractor in the traditional way, namely, using a statistical extractor to get a close-to-uniform key and using a PRG or PRF to get additional pseudorandom bits as needed. The novelty of that work, however, is that they show that if the output from the statistical extractor (implemented via a suitable hash function) is used as a key to a wPRF and this wPRF is applied to a random point then the best possible distinguishing advantage against the output of this scheme is the wPRF's best distinguishing advantage plus $2^{-(k-m)}$. This is an improvement over the generic analysis using statistical extractors where the latter term would be $2^{-(k-m)/2}$. This relaxes the entropy requirement from the source and is significant in cases as those considered above (e.g. when generating keys from Diffie-Hellman protocols of relative small order). Moreover, depending on the security parameters, the analysis from [BDK+11] can sometimes be used, as in our case, to generate keys that are even larger than the available entropy. The crucial difference with our construction, however, is that [BDK+11] requires the implementation of a statistical extractor (with its corresponding seed) in addition to the wPRF. In contrast, our scheme re-uses the PRF already available in most cryptographic implementations without requiring extra machinery (which may seem a minor issue considering the relative simplicity of statistical extractors but represents a significant barrier for adoption into standardized protocols, particularly those requiring hardware support). On the downside, our scheme is limited to situations where the source of key material produces values that are no longer than the key of the wPRF, while [BDK+11,Kra10] have no such length restrictions.

References

[BDK+11] Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In *CRYPTO*, volume 6841 of *Lect. Notes in Comp. Sci.*, pages 1–20, 2011.

[DGH+04] Yevgeniy Dodis, Rosario Gennaro, Johan Håstad, Hugo Krawczyk, and Tal Rabin. Randomness extraction and key derivation using the cbc, cascade and hmac modes. In *CRYPTO*, pages 494–510, 2004.

[DGKM11] Dana Dachman-Soled, Rosario Gennaro, Hugo Krawczyk, and Tal Malkin. Computational extractors and pseudorandomness, 2011. Full version of this paper. Available from `eprint.iacr.org/2011/708`.

[GGKT05] Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM J. Comput.*, 35(1):217–246, 2005.

[GKR04] Rosario Gennaro, Hugo Krawczyk, and Tal Rabin. Secure Hashed Diffie-Hellman over Non-DDH Groups. In *EUROCRYPT*, volume 3027 of *Lect. Notes in Comp. Sci.*, pages 361–381, 2004.

[Gol90] Oded Goldreich. A note on computational indistinguishability. *Inf. Process. Lett.*, 34(6):277–281, 1990.

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

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proc. 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 44–61, 1989.

[IW97] R. Impagliazzo and A. Widgerson. P = bpp unless e has subexponential circuits: derandomizing the xor lemma. In *Proceedings of the Twenty-Ninth Annual Symposium on Theory of Computing*, pages 220–229, 1997.

[KLR09] Yael Tauman Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *FOCS*, pages 617–626, 2009.

[Kra10] Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In *CRYPTO*, pages 631–648, 2010.

[NW88] Noam Nisan and Avi Wigderson. Hardness vs. randomness. In *Proc. 29th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 2–11. IEEE Computer Society Press, 1988.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

[Pie09] Krzysztof Pietrzak. A leakage-resilient mode of operation. In *EUROCRYPT*, pages 462–482, 2009.

[RTS00] Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discrete Math.*, 13(1):2–24, 2000.

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

[Sha02] Ronen Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the EATCS*, 77:67–95, 2002.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, July 2001.

[TV00] Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–42, 2000.