Extracting Randomness from Extractor-Dependent Sources

Yevgeniy Dodis
NYU

Vinod Vaikuntanathan
MIT

Daniel Wichs
Northeastern University and NTT Research Inc.

March 4, 2020

Abstract

We revisit the well-studied problem of extracting nearly uniform randomness from an arbitrary source of sufficient min-entropy. Strong seeded extractors solve this problem by relying on a public random seed, which is unknown to the source. Here, we consider a setting where the seed is reused over time and *the source may depend on prior calls to the extractor with the same seed*. Can we still extract nearly uniform randomness?

In more detail, we assume the seed is chosen randomly, but the source can make arbitrary oracle queries to the extractor with the given seed before outputting a sample. We require that the sample has entropy and differs from any of the previously queried values. The extracted output should look uniform even to a distinguisher that gets the seed. We consider two variants of the problem, depending on whether the source only outputs the sample, or whether it can also output some correlated public *auxiliary information* that preserves the sample's entropy. Our results are:

Without Auxiliary Information: We show that every *pseudo-random function* (PRF) with a sufficiently high security level is a good extractor in this setting, even if the distinguisher is computationally unbounded. We further show that the source necessarily needs to be computationally bounded and that such extractors imply one-way functions.

With Auxiliary Information: We construct secure extractors in this setting, as long as both the source and the distinguisher are computationally bounded. We give several constructions based on different intermediate primitives, yielding instantiations based on the DDH, DLIN, LWE or DCR assumptions. On the negative side, we show that one cannot prove security against computationally unbounded distinguishers in this setting under any standard assumption via a black-box reduction. Furthermore, even when restricting to computationally bounded distinguishers, we show that there exist PRFs that are insecure as extractors in this setting and that a large class of constructions cannot be proven secure via a black-box reduction from standard assumptions.

Contents

1 Introduction

Extracting Randomness. Randomness is an important ingredient in many algorithmic tasks, and is especially crucial in cryptography. Indeed, much of cryptography relies on the assumption that parties can sample uniformly random bits. However, most natural sources of randomness are imperfect and not uniformly random. This motivates the study of *randomness extraction*, whose goal is to extract (nearly) uniform randomness from imperfect sources.

Ideally, we would have a deterministic function Ext that converts an imperfect source of randomness X into a (nearly) uniformly random output $\mathsf{Ext}(X)$. Furthermore, such an extractor should work for all sources of randomness X having a sufficiently large amount of (min-)entropy. Unfortunately, this is easily seen to be impossible, even if we only want to output 1 bit [CG85]: for every extractor function Ext, there is a source X that has almost full min-entropy yet the output of $\mathsf{Ext}(X)$ is completely fixed.

There have been two broad lines of work to get around this. The first line of work designs extractors for restricted types of sources X that satisfy additional requirements beyond just having entropy (see e.g., [von51, CGH+85, Blu86, LLS89, CG85, TV00, BST03, BIW04, CZ16]). While this is an important research direction, we often know very little about natural sources of randomness and they may fail to satisfy the imposed requirements. The second line of work considers *(strong) seeded* extractors [NZ93, NZ96], where the extractor is given a public uniformly random seed S, which is independent of the source X, and we require that the extracted output $\mathsf{Ext}(X;S)$ is close to uniform even given the seed S.

Extractor-Dependent Sources. In this work, we consider a seeded extractor and envision a scenario where a single uniformly random seed S is chosen once and then is reused over time by many different users and/or applications to extract randomness from various "natural" sources of entropy. For example, the seed S could be a part of a system *random number generator* (RNG) that extracts randomness from physical sources of entropy, such as the timing of interrupts etc. If the sources are truly independent of the seed S, then standard (strong) seeded extractors suffice to guarantee that the extracted outputs are nearly uniform. However, since the seed S is continuously reused, past outputs of the extractor will make their way back into "nature" and may affect the sources in the future. For example, interrupts may depend on processes that themselves rely on previous outputs of the extractor. Furthermore, since we cannot assume that all users/applications use the extractor securely, we have to allow for the possibility that some of the prior calls to the extractor were made on arbitrary samples that may not have any entropy. Unfortunately, if the source can depends on prior calls to the extractor with the same seed S, we violate the condition that the source is independent of the seed and can no longer rely on the security of standard seeded extractors. We emphasize that, although the seed S is public, the sources are *not* fully adversarial and *not* arbitrarily dependent on S. (A restriction of this sort is of course necessary to circumvent the obvious impossibility result.) Instead, we assume that the sources can only depend on prior calls to the extractor with the given seed S, but are otherwise independent of S. We call such sources "extractor-dependent". Can we design *extractors for extractor-dependent sources (ED-Extractors)* that manage to extract nearly uniform randomness in this setting?

Defining the Problem. We now specify the problem in more detail. Our goal is to design a seeded extractor EDExt that extracts randomness from extractor-dependent sources. We consider a

setting where a seed S is chosen uniformly at random. A *source* $\mathcal{S}^{\mathsf{EDExt}(\cdot,S)}$ gets oracle access to the extractor with the seed S and outputs a sample X along with some public auxiliary information AUX. We say that such a source $\mathcal{S}$ is a *legal* extractor-dependent source of entropy α if two conditions hold: (1) the (conditional min-entropy) of X given S, AUX is at least α, and (2) the source never queries the oracle on the value X that it outputs. An α-ED-Extractor needs to ensure that for all legal extractor-dependent sources of entropy α, the output $\mathsf{EDExt}(X, S)$ is indistinguishable from uniform, even given the seed S and the auxiliary information AUX.

Discussion on the Legality Conditions. We motivate the reason behind the two legality conditions imposed by the definition.

Firstly, just like for standard (seeded) extractors, we need to assume that X has a sufficient level of entropy even conditioned on AUX in order to extract randomness from it. In our case, the source also has access to the oracle $\mathsf{EDExt}(\cdot, S)$ with a random seed S, but we want the entropy to come from the internal randomness of the source rather than from the seed S since the latter is public and known to the distinguisher. Therefore, it is natural to also condition on S.

The second condition is clearly necessary: without it we could define a source that queries the oracle on random values and outputs the first such value on which the extracted output starts with a 0. Such a source would have almost full entropy, yet the extracted output would be easily distinguishable from uniform. Moreover, this condition is also reasonable when modeling our intended scenario since the sample should have entropy even given all the prior extractor calls that influenced nature, and therefore it should differ from all of them.

In particular, the two legality conditions include the following simpler sub-class of sources, which already intuitively captures our intended scenario. Consider sources $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ that consists of two components. The first component $\mathcal{S}_1^{\mathsf{EDExt}(\cdot,S)}$ makes arbitrary oracle calls to the extractor and models the influence that these calls have on nature; it outputs some value state. The second component $\mathcal{S}_2(\mathsf{state})$ then outputs X, AUX without making any further oracle queries and captures the entropic process that produces the sample. The only condition we impose is that, for every possible fixed value of state, the entropy of X conditioned on AUX when they are sampled according to $\mathcal{S}_2(\mathsf{state})$ should be at least α. If α is large enough then $\mathcal{S}$ satisfies both of the previous legality conditions. In particular, $\mathcal{S}_1$ could not have queried the oracle on X since the entropy of X comes only from the random coins of $\mathcal{S}_2$ that are unknown to $\mathcal{S}_1$.

Discussion on Auxiliary Info. Our default definition allows the source to output some public auxiliary info AUX that can be correlated with the sample X as long as it preserves its (average conditional min-)entropy. It is natural that some such information may be public (e.g., the source X denotes the timing of interrupts, but the adversary can learn some auxiliary info AUX denoting the high-order bits of such timings by interacting with the system). We also consider a weaker setting *without auxiliary info*, where we don't have AUX. In the case of standard seeded extractors, it turns out that there is not much difference between a setting with auxiliary info and without [DORS08]. However, as we will see, there is a significant difference between the two settings when it comes to ED-Extractors.

Prior Work. The work of Coretti et al. [CDKT19] initiates the study of extracting from extractor-dependent sources in the special case where the extractor is a *random oracle*. While their definition

is specifically tailored to the random-oracle model, our definition can be seen as the natural extension of it to the standard model. In particular, they consider the setting where $\mathcal{O}(\cdot) = \mathsf{EDExt}(\cdot, S)$ is a truly random function. They show that this is an α-ED-Extractor for any super-logarithmic entropy α, as long as the source only makes polynomially many queries, but even if the distinguisher is computationally unbounded and can see the entire truth table of the oracle. This gives us heuristic evidence that a "good" cryptographic hash function is an ED-Extractor in the standard model even against computationally unbounded distinguishers (as long as the source is computationally bounded). The main open question is therefore whether we can construct ED-Extractors under standard computational assumptions.

1.1 Our Results

We give positive and negative results for ED-Extractors with and without auxiliary info.

Without Auxiliary Info. On the positive side, we show that any pseudo-random function (PRF) with a sufficiently high security level is a good ED-Extractor without auxiliary info. In particular, assuming the existence of sub-exponentially secure one-way functions, there exist α-ED-Extractors with any output size m for entropy $\alpha = m + \omega(\log \lambda)$, where λ is the security parameter. Furthermore, such extractors achieve security even against computationally unbounded distinguishers, as long as the source runs in polynomial time. If we only want security against polynomial-time distinguishers, we can allow the output size to grow to an arbitrary polynomial m while only requiring entropy $\alpha = \lambda^{\Omega(1)}$.

On the negative side, we show that ED-Extractors imply one-way functions and therefore cannot be constructed unconditionally. This holds even without auxiliary info, even if we require that the source has almost full entropy, and even if the extractor outputs only 1 bit. Furthermore, we show that such ED-Extractors cannot exist for computationally unbounded sources.

With Auxiliary Info. We construct ED-Extractors in the setting with auxiliary info under standard assumptions. In particular, we give three constructions.

- The first construction relies on (adaptively secure) *constrained PRFs* [BGI14,KPTZ13,BW13] for NC1 constraints. These can be instantiated under the sub-exponential security of either the *learning with errors (LWE)* [BV15] or the *Decisional Diffie-Hellman Inversion (DDHI)* assumption in arbitrary prime-order groups (without requiring pairings) [AMN+18].[1]
- The second construction relies on *shift-hiding shiftable functions* [PS18], which can be seen as a type of constraint-hiding constrained PRFs, and can be instantiated under LWE without requiring sub-exponential security.
- The third construction relies on lossy functions and can be instantiated under any of: decisional Diffie-Hellman (DDH), decisional-linear (DLIN), LWE, or decisional composite residuosity (DCR) assumptions.

[1] The DDHI assumption in a cyclic group $\mathbb{G}$ of order q with generator g states that, given any polynomially many values of the form $(g, g^a, g^{a^2}, \ldots, g^{a^L})$ where $a \leftarrow \mathbb{Z}_q$, the value $g^{1/a}$ is computationally indistinguishable from uniform.

In all cases, we prove security against polynomial-time sources and distinguishers. Our α-ED-Extractors can have arbitrarily large polynomial input size n and output size m, and require entropy $\alpha = \lambda^{\Omega(1)}$.

Note that, in the setting without auxiliary info, we achieved security even against computationally unbounded distinguishers. Furthermore, the random-oracle based result of [CDKT19] heuristically suggests that good cryptographic hash functions achieve security against computationally unbounded distinguishers even in the auxiliary info setting. However, our constructions in the auxiliary info setting from standard assumptions only achieve security against polynomial-time distinguishers. Unfortunately, we show that this is inherent. In particular, we show that in the auxiliary info setting, one cannot prove the security of any ED-Extractor against computationally unbounded distinguishers under any standard assumption via a black-box reduction.

Furthermore, our instantiations in the auxiliary info setting rely on "cryptomania" assumptions (known to imply public-key encryption]) rather than one-way functions, and we ask whether this is necessary. While we do not resolve this question, we give some evidence that the two settings necessitate substantially different constructions. Firstly, one may be tempted to conjecture that every PRF is also a good ED-Extractor even in the auxiliary info setting. We show that this is not the case: there exist PRFs that are insecure as ED-Extractors in the auxiliary info setting even for very high levels/rates of entropy α. Moreover, we show that a large class of natural PRFs (e.g., the Naor-Reingold PRF) cannot be proven to be secure ED-Extractors in the setting of auxiliary info via a black-box reduction from any standard assumption.

1.2 Our Techniques

ED-Extractors without Auxiliary Info from PRFs. Our first result shows that every PRF is already a good ED-Extractor in the setting without auxiliary info. In particular, the seed of the extractor is the PRF key and the extractor just evaluates the PRF on the sample X. The main difficulty in proving ED-Extractor security is that the distinguisher gets the seed of the ED-Extractor, but PRF security only holds if the key is never revealed. Our insight is to design a reduction that never calls the distinguisher – indeed, this allows us to prove security even for computationally unbounded distinguishers.

Let's start with the case where the PRF/Extractor only outputs 1 bit. If the extracted output is statistically far from uniform given the seed, it means that it is biased towards either 0 or 1, but the direction of the bias is unknown and may be different for each seed. Consider running the source $\mathcal{S}$ twice with independent randomness, while giving it oracle access to the PRF/Extractor with the same random key/seed. Let X_0, X_1 be the samples that the two runs output respectively. Then the PRF/Extractor evaluations on those samples are more likely to agree than disagree, since they are biased in the same direction. But the legality conditions ensure that X_0, X_1 were never queried during either of the two runs and are different from each other (since each run cannot query its own output and the output of the other run should have enough entropy to be be unpredictable). So, given oracle access to the PRF, we can use the source $\mathcal{S}$ to find two values X_0, X_1 that we haven't yet queried, but if we then proceed to query the PRF on them, the outputs are noticeably more likely to agree than disagree. This cannot be the case given oracle access to a random function, and therefore allows us to distinguish the two and break PRF security. The analysis extends to a larger output size m, but the advantage of the reduction shrinks by a factor of 2^{-m}. Therefore, we need very secure PRFs that cannot be distinguished from random functions with advantage better than $\mathrm{negl}(\lambda)2^{-m}$, which requires sub-exponential security assumptions.

Note that the above argument completely breaks down in the setting with auxiliary info. The problem is that now the direction of the bias can be different for each choice of the key/seed *and* the auxiliary info. But the two independent runs of the source $\mathcal{S}$ are unlikely to produce the same auxiliary info and hence we cannot argue that the bias would go in the same direction. Indeed, we show that there are PRFs that are completely insecure as ED-Extractors in the setting with auxiliary info.

ED-Extractors imply One-Way Functions. We show that ED-Extractors cannot exist if the source is allowed to be computationally unbounded. This holds even in the setting without auxiliary info, even if we only consider polynomial-time distinguishers, even if we require that the source has almost full entropy, and even if the extractor outputs only 1 bit. The high level idea is that a computationally unbounded source $\mathcal{S}$ with oracle access to the function $\mathsf{EDExt}(\cdot, S)$ can learn the function sufficiently well to predict its output on a random value with high probability. It can then sample a random X subject to predicting that $\mathsf{EDExt}(X, S) = 0$, without querying the extractor on X. This is a legal source with almost full entropy, yet the extractor output is highly biased towards 0. We extend the above argument to showing that such extractors imply one-way functions.

ED-Extractors with Auxiliary Info from Constrained PRFs. We construct ED-Extractors in the setting with auxiliary info, using constrained pseudorandom functions (C-PRF). A C-PRF allows us to constrain a PRF key k on some constraint function C to yield a constrained key, denoted $k\{C\}$. The constrained key allows us to evaluate the PRF on all points x such that $C(x) = 0$. However, given the constrained key $k\{C\}$, the PRF outputs at all points x for which $C(x) = 1$ look random. We need to rely on *adaptively secure* constrained PRFs, where the adversary can choose the constraint C after seeing some PRF outputs.

Our construction of ED-Extractors uses a constrained PRF and a standard (seeded) randomness extractors Ext. The seed of the ED-Extractor is a constrained PRF key $k\{C_{S,U}\}$, with the constraint $C_{S,U}(X)$ that outputs 1 (i.e., prevents evaluation) on all points X such that $\mathsf{Ext}(X; S) = U$, where S, U are chosen randomly. We choose the output size of the extractor to be $\ell = \omega(\log \lambda)$ and therefore the key is constrained on a negligible fraction of points. On input X, the ED-Extractor checks if $C_{S,U}(X) = 1$, in which case it outputs some fixed dummy value, and otherwise it uses the seed $k\{C_{S,U}\}$ to evaluate the PRF on X.

To argue ED-Extractor security, we consider a source $\mathcal{S}^{\mathsf{EDExt}(\cdot, k\{C_{S,U}\})}$ that gets oracle access to the ED-Extractor with a random seed $k\{C_{S,U}\}$ and outputs X, AUX. A distinguisher $\mathcal{D}$ then gets the seed $k\{C_{S,U}\}$ together with AUX and the extracted output $R = \mathsf{EDExt}(X, k\{C_{S,U}\})$. We first argue that this is statistically indistinguishable from giving the source $\mathcal{S}$ oracle access to the unconstrained PRF and setting R to be the output of the PRF with the unconstrained key on X (since the probability that any of the queries of $\mathcal{S}$ or its output lie in the constrained set is negligible). Now, instead of giving the distinguisher $\mathcal{D}$ the constrained key $k\{C_{S,U}\}$ where U is uniform, we give it the key $k\{C_{S,\mathsf{Ext}(X;S)}\}$ which is constrained on X. This is statistically indistinguishable since X has entropy even conditioned on AUX and is sampled independently of S; therefore $\mathsf{Ext}(X; S)$ is close to uniform even given AUX. But now we can switch R from the output of the PRF on X to uniform, and this is computationally indistinguishable even given the constrained PRF key $k\{C_{S,\mathsf{Ext}(X;S)}\}$ since it is constrained on X (and we know that the source didn't query the oracle on X). This shows that the extracted output is indistinguishable form uniform even given the ED-Extractor seed and the auxiliary info. (The above proof outline conveys the intuition but is

slightly oversimplified and ignores some subtleties; see the full proof for details).

Since standard extractors can be evaluated in NC1, we only need constrained PRFs for NC1 circuits. Fortunately, we have such constructions from the LWE and DDHI assumptions [BV15, AMN+18]. However, they only achieve selective security, where the constrained circuit needs to be chosen ahead of time before any PRF outputs are given out, while we need adaptive security. We can get this via standard complexity leveraging at the cost of having to assume the sub-exponential security of the LWE and DDHI assumptions.

Additional Constructions. We give two alternate constructions of ED-Extractors in the setting with auxiliary info. The first uses *shift-hiding shiftable functions* [PS18], which can be instantiated from standard LWE, without needing complexity leveraging. The construction and proof of security differ substantially from the one above. The second one uses *lossy functions*, which are essentially equivalent to lossy trapdoor functions (LTDFs) [PW08] without requiring a trapdoor. The construction can be instantiated from several different assumptions (DDH,DLIN,LWE,DCR). See Sections 5.2 and 5.3.

Not all PRFs are ED-Extractors with Aux Info. We construct PRFs, which fail to be good extractors in the setting of auxiliary info. For example, consider a PRF which first hashes the input x into a small digest using a collision-resistant hash function and then applies another PRF on the output. Consider a source that chooses a random x and sets the auxiliary info to be the hash of x. Since the hash is small, this does not reduce the entropy of x by much. However, if the distinguisher is given the PRF key (which is the ED-Extractor seed) and the auxiliary info, it can compute the PRF on x and therefore easily distinguish it from uniform. In this example, the auxiliary info reduces the entropy of x by some small super-logarithmic amount. We give an even more dramatic example of this type using fully-homomorphic encryption (FHE) where the auxiliary info reduces the entropy of x by only 1 bit.

Black-Box Separation Results. Lastly, we give two black-box separation results showing that, in the auxiliary info setting, one cannot prove security (via a black-box reduction under a standard assumption) against computationally unbounded distinguishers or for certain natural classes of constructions. Our results rely on the framework of [Wic13] and rely on the fact that the ED-Extractor definition is expressed as a two-stage game where the attacker consists of two components (the source and the distinguisher) that cannot communicate. This allows us to give black-box separations showing that, in certain cases, we cannot prove security under any standard assumption which is in the form of a single-stage game between a challenger and a single stateful adversary.

1.3 Additional Related Work

RNGs. Our scenario is partially motivated by the problem of extracting randomness from physical sources as part of a system Random Number Generator (RNG). We note that extracting randomness is only one component of a good RNG; see e.g., [BH05, DPR+13, DSSW14, GT16, Hut16, CDKT19] for works that formally deal with the broader problem of RNG design.

Universal Computational Extractors (UCE). The notion of universal computational extractors (UCE) [BHK13, ST17] was originally proposed as a way of capturing "random-oracle like"

security properties of hash functions via a standard-model definition. While the format of the UCE definition is also given in terms of an extraction game with a source and a distinguisher, there are major differences between the UCE definition and that of ED-Extractors, both in terms of their syntactic structure, but also more conceptually in terms of what they aim to capture. The key such difference is that the notion of legal source is defined in the "ideal model", and permits sources which only have *computational* unpredictability in the "real" model (say, conditioned on the auxiliary information).[2] In contrast, this work only aimed to capture a smaller class of sources that have entropy even in the "real model", but could depend of the previous extractor output.

Unfortunately, it is known that even the weakest form of UCE security cannot be achieved under standard assumptions (via black-box reductions; this indirectly follows from [Wic13]), while our work shows that ED-Extractors can. It remains an interesting open problem whether ED-Extractors can be used in place of UCEs to get any broader cryptographic applications beyond the immediate ones of extracting randomness.

Low-Complexity Samplers. Introduced by Trevisan and Vadhan [TV00] and later extended by [KRVZ11], these seedless extractors assume that the entropy source producing input X is unable to run the extractor even once. In contrast, our sampler can be much slower than the extractor, but we use a seed and give the sampler oracle access to the extractor, before releasing the seed to the distinguisher.

Seed-Dependent condensers. This approach, formalized by Dodis, Ristenpart and Vadhan [DRV12], relaxes the security guarantees of the randomness extractor to only ensure that the output of the condenser is almost full entropy, but not necessarily close to uniform. In this sense it is weaker than ED-Extractors. However, the sampler is given the actual seed, which is stronger than our setting. Interestingly, the availability of auxiliary information also played a crucial role in the constructions of seed-dependent condensers from standard assumptions.

2 Preliminaries

When X is a distribution, or a random variable following this distribution, we let $x \leftarrow X$ denote the process of sampling x according to the distribution X. If X is a set, we let $x \leftarrow X$ denote sampling x uniformly at random from X.

Let X, Y be random variables with supports S_X, S_Y, respectively. We define their *statistical difference* as

$$\mathsf{SD}(X,Y) = \frac{1}{2} \sum_{u \in S_X \cup S_Y} \left| \Pr[X = u] - \Pr[Y = u] \right| .$$

The *min-entropy* of a random variable X is $H_\infty(X) = -\log(\max_x \Pr[X = x])$. Following Dodis et al. [DORS08], we define the (average) conditional min-entropy of X given Y as: $H_\infty(X|Y) = -\log\left(\mathbb{E}_{y \leftarrow Y}\left[2^{-H_\infty(X|Y=y)}\right]\right)$. Note that $H_\infty(X|Y) = k$ iff the optimal strategy for guessing X given Y succeeds with probability 2^{-k}.

Lemma 2.1. *For any random variables X, Y where Y is supported over a set of size T we have $H_\infty(X|Y) \leq H_\infty(X) - \log T$.*

[2] Somewhat confusingly, this is true even for so called "UCEs for statistically unpredictable sources".

Definition 2.2 ((Strong, Average-Case) Seeded Extractor [NZ96])**.** *We say that an efficient function* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^\ell$ *is an* (α, ε)-extractor *if for all random variables* (X, Z) *such that* X *is supported over* $\{0,1\}^n$ *and* $H_\infty(X|Z) \geq \alpha$ *we have* $\mathsf{SD}(\ (Z, S, \mathsf{Ext}(X; S))\ ,\ (Z, S, U_\ell)) \leq \varepsilon$ *where* S, U_ℓ *are uniformly random and independent bit-strings of length* d, ℓ *respectively.*

Theorem 2.3 ([ILL89])**.** *There exist* (α, ε)*-extractors with input length* n *and output length* ℓ *as long as* $\alpha \geq \ell + 2\log(1/\varepsilon)$*.*

Definition 2.4 ((Strong, Average-Case) Two-Source Extractor [CG88])**.** *We say that an efficient function* $2\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *is an* (e_1, e_2, δ)-strong 2-source extractor *if for all random variables* (X_1, X_2, Z) *such that* X_1, X_2 *are independent conditioned on* Z *and* $H_\infty(X_1|Z) \geq e_1, H_\infty(X_2|Z) \geq e_2$ *we have* $\mathsf{SD}(\ (Z, X_2, 2\mathsf{Ext}(X_1; X_2))\ ,\ (Z, X_2, U_m)) \leq \delta$ *where* U_m *is a uniformly string of length* m*.*

Theorem 2.5 ([Raz05])**.** *For any polynomial input length* $n = \mathrm{poly}(\lambda)$*, any* $e_1 = \lambda^{\Omega(1)}$ *and any* $e_2 = (1/2 + \Omega(1))n$*, there exist* (e_1, e_2, δ)*-extractor with input length* n*, output length* $m = \lambda^{\Omega(1)}$ *and error* $\delta = 2^{-\lambda^{\Omega(1)}}$*.*

Definition 2.6. *The collision probability of a random variable* A *is defined as* $\mathsf{Col}(A) = \Pr[a = a' \ :\ a \leftarrow A, a' \leftarrow A]$*. The conditional collision probability of* A *given* B *is defined as* $\mathsf{Col}(A|B) = \Pr[a = a' \ :\ b \leftarrow B, a \leftarrow (A|B = b), a' \leftarrow (A|B = b)]$*.*

Claim 2.6.1 (Statistical Distance vs Collision Probability [IZ89])**.** *Let* A *be a random variable supported over* $\{0,1\}^m$ *such that* $\mathsf{SD}(A, U_m) \geq \varepsilon$*, where* U_m *is uniform over* $\{0,1\}^m$*. Then* $\mathsf{Col}(A) \geq \frac{1}{2^m}(1 + 4\varepsilon^2)$*.*

Furthermore, let A, B *be correlated random variables, where* A *is supported over* $\{0,1\}^m$ *and*

$$\mathsf{SD}(\ (A, B)\ ,\ (U_m, B)\ ) \geq \varepsilon.$$

Then $\mathsf{Col}(A|B) \geq \frac{1}{2^m}(1 + 4\varepsilon^2)$*.*

Proof. See [IZ89] for the first part of the claim. For the second part of the claim, we have:

$$\begin{aligned}
\mathsf{Col}(A|B) &= \mathbb{E}_{b \leftarrow B}[\mathsf{Col}(A|B = b)] \\
&\geq \mathbb{E}_{b \leftarrow B}[\frac{1}{2^m}(1 + 4\varepsilon_b^2)] \quad \text{where} \quad \varepsilon_b = \mathsf{SD}((A|B = b), U_m) \\
&\geq \frac{1}{2^m}(1 + 4\mathbb{E}_{b \leftarrow B}[\varepsilon_b^2]) \\
&\geq \frac{1}{2^m}(1 + 4\mathbb{E}_{b \leftarrow B}[\varepsilon_b]^2) \\
&\geq \frac{1}{2^m}(1 + 4\varepsilon^2)
\end{aligned}$$

where the first line follows from the definition of (conditional) collision probability, the second line follows by applying the first part of the claim on the distributions $(A|B = b)$, the third line follows by the linearity of expectation and the fourth line follows by Jensen's inequality. □

Learning with Errors. The (n, m, q, χ) LWE assumption states that $(A, sA+e)$ is computationally indistinguishable from (A, u) where $A \leftarrow \mathbb{Z}_q^{n\times m}$, $s \leftarrow \mathbb{Z}_q^n$, $e \leftarrow \chi^m$ and $u \leftarrow \mathbb{Z}_q^m$. Throughout this work, the LWE assumption (without qualification), refers to assuming that there exists some $n = \text{poly}(\lambda)$, some $q \geq 2^{\lambda^{\Omega(1)}}$ and some distribution χ over $\mathbb{Z}$ which is $\text{poly}(\lambda)$ bounded such that the (n, m, q, χ) assumption holds for all $m = \text{poly}(\lambda)$. This is implied by the hardness of worst-case lattice problems with sub-exponential approximation factors.

Definition 2.7 (Pseudorandom Function (PRF) [GGM84])**.** *A polynomial-time function $F : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^m$ with key length $\ell = \ell(\lambda)$, input length $n = n(\lambda)$ and output length $m = m(\lambda)$ is a PRF if for any polynomial-time attacker $\mathcal{A}$ there exists some negligible function $\mu(\lambda) = \text{negl}(\lambda)$ such that*

$$| \Pr[\mathcal{A}^{F(k,\cdot)}(1^\lambda) = 1] - \Pr[\mathcal{A}^{\mathcal{O}(\cdot)}(1^\lambda) = 1] | \leq \mu(\lambda).$$

where we choose $k \leftarrow \{0,1\}^\ell$ and $\mathcal{O} : \{0,1\}^n \to \{0,1\}^m$ is a uniformly random function. We say that the PRF has security level $\sigma = \sigma(\lambda)$ if $\mu(\lambda) \leq 1/\sigma(\lambda)$.

Definition 2.8 (Constrained PRFs (CPRF) [BGI14, KPTZ13, BW13])**.** *A CPRF for a class of constraints $\mathcal{C} = \{\mathcal{C}_\lambda\}$ consists of two polynomial-time algorithms $(F, \mathsf{Constrain})$ where:*

- *$y = F(k, x)$ is a deterministic polynomial-time function that takes as input a key k (either constrained or unconstrained) and a value $x \in \{0,1\}^n$ and outputs $y \in \{0,1\}^m$ for some polynomial length parameters $n = n(\lambda), m = m(\lambda)$ in the security parameter λ.*
- *$k\{C\} \leftarrow \mathsf{Constrain}(k, C)$ takes as input a key $k \in \{0,1\}^\lambda$ and a constraint $C : \{0,1\}^n \to \{0,1\}$ with $C \in \mathcal{C}_\lambda$. It outputs a constrained key, denoted $k\{C\}$.*

We require that the scheme satisfies a correctness and a security property defined below:

Correctness: *We require that no adversary can find an input which is not constrained, yet the constrained key disagrees with the original key. More concretely, consider the following game between a stateful adversary $\mathcal{A}$ and a challenger:*

- *The adversary $\mathcal{A}(1^\lambda)$ chooses $C \in \mathcal{C}_\lambda$*
- *The challenger chooses $k \in \{0,1\}^\lambda$ and $k\{C\} \leftarrow \mathsf{Constrain}(k, C)$.*
- *The adversary $\mathcal{A}^{F(k,\cdot)}(k\{C\})$ gets the constrained key $k\{C\}$ and oracle access to $F(k, \cdot)$. It outputs a value $x \in \{0,1\}^n$.*

We require that, in the context of the above experiment, we have $\Pr[C(x) = 0 \wedge F(k, x) \neq F(k\{C\}, x)] \leq \text{negl}(\lambda)$.

(Adaptive) Security: *Consider the following distinguishing game between an adversary $\mathcal{A}$ and a challenger:*

- *Challenger chooses $k \leftarrow \{0,1\}^\lambda$ and a bit $b \leftarrow \{0,1\}$.*
- *Adversary $\mathcal{A}^{F(k,\cdot)}(1^\lambda)$ gets oracle access to $F(k, \cdot)$ and outputs a constraint $C \in \mathcal{C}_\lambda$ and a values x such that $C(x) = 1$ and x was never queries to the oracle.*
- *If $b = 0$, the challenger sets $r = F(k, x)$ and else it chooses $r \leftarrow \{0,1\}^m$. The challenger also computes $k\{C\} \leftarrow \mathsf{Constrain}(k, C)$.*

- *The adversary $\mathcal{A}$ is given $k\{C\}$ and r. It outputs a bit b'.*

We require that for all polynomial-time adversaries $\mathcal{A}$, we have $|\Pr[b = b'] - \frac{1}{2}| = \mathrm{negl}(\lambda)$.

We also consider several variants of the definition. Firstly, we define the notion of no-constrained-evaluation *security, where we restrict the adversary to never querying the oracle $F(k, \cdot)$ on a point x for which $C(x) = 1$. Secondly, we consider selective security where the adversary chooses $C \in \mathcal{C}_\lambda$ at the beginning of the game before getting oracle access to $F(k, c)$). Lastly, we consider* no-evaluation *security where the adversary does not get oracle access to $F(k, \cdot)$ at all.*

Note that, via a simple guessing argument where we guess the adversary's choice of C, selective security with a sufficiently high security level $\sigma(\lambda) = |\mathcal{C}_\lambda|\omega(\log \lambda)$ implies adaptive security. Furthermore by the same argument, no-evaluation security (which is inherently selective) with a sufficiently high security level $\sigma(\lambda) = |\mathcal{C}_\lambda|\omega(\log \lambda)$ implies no-constrained-evaluation security. This is because, if we guess the adversary's choice of C ahead of time and gets $k\{C\}$, we can answer queries on unconstrained points using $k\{C\}$ rather than calling the PRF oracle.

Definition 2.9 (Shift-Hiding Shiftable Functions [PS18])**.** *A shift-hiding shiftable family of functions (CPRF) for a class of shift functions $\mathcal{S} = \{\mathcal{S}_\lambda\}$ consists of two polynomial-time algorithms $(F, \mathsf{Shift}, \mathsf{Round})$ where:*

- *$y = F(k, x)$ is a deterministic polynomial-time function that takes as input a key k (either "original" or "shifted") and a value $x \in \{0,1\}^n$ and outputs $y \in \{0,1\}^m$ for some polynomial length parameters $n = n(\lambda), m = m(\lambda)$ in the security parameter λ.*
- *$k\{\mathsf{Sh}\} \leftarrow \mathsf{Shift}(k, \mathsf{Sh})$ takes as input a key $k \in \{0,1\}^\lambda$ and a shift function $\mathsf{Sh} : \{0,1\}^n \to \{0,1\}^m$ with $\mathsf{Sh} \in \mathcal{S}_\lambda$. It outputs a shifted key, denoted $k\{\mathsf{Sh}\}$.*

We require that the scheme satisfies a correctness and a security property defined below:

Correctness: *First, we require that there is a round operation Round such that for any x and any shift function $\mathsf{Sh} \in \mathcal{S}_\lambda$, with probability $1 - \mathrm{negl}(\lambda)$ over the choice of the PRF key k,*

$$\mathsf{Round}(F(k\{\mathsf{Sh}\}, x)) = \mathsf{Round}(F(k, x) + \mathsf{Sh}(x))$$

Moreover, we require that no adversary given oracle access to $F(k, \cdot)$, can find a x that violates this equality. More concretely, consider the following game between a stateful adversary $\mathcal{A}$ and a challenger:

- *The adversary $\mathcal{A}(1^\lambda)$ chooses $\mathsf{Sh} \in \mathcal{S}_\lambda$*
- *The challenger chooses $k \in \{0,1\}^\lambda$ and $k\{\mathsf{Sh}\} \leftarrow \mathsf{Shift}(k, \mathsf{Sh})$.*
- *The adversary $\mathcal{A}^{F(k,\cdot)}(k\{\mathsf{Sh}\})$ gets the shifted key $k\{\mathsf{Sh}\}$ and oracle access to $F(k, \cdot)$. It outputs a value $x \in \{0,1\}^n$.*

We require that, in the context of the above experiment, we have

$$\Pr[\mathsf{Round}(F(k\{\mathsf{Sh}\}, x)) \neq \mathsf{Round}(F(k, x) + \mathsf{Sh}(x))] \leq \mathrm{negl}(\lambda)$$

Selective Shift-Hiding Security: *Consider the following distinguishing game between an adversary $\mathcal{A}$ and a challenger:*

- *Adversary chooses two shift functions* $\mathsf{Sh}_0, \mathsf{Sh}_1 \in \mathcal{S}_\lambda$.
- *Challenger chooses* $k \leftarrow \{0,1\}^\lambda$ *and a bit* $b \leftarrow \{0,1\}$ *and sets* $k\{\mathsf{Sh}_b\} \leftarrow \mathsf{Shift}(k, \mathsf{Sh}_b)$.
- *Adversary* $\mathcal{A}^{F(k,\cdot)}(1^\lambda, k\{\mathsf{Sh}_b\})$ *gets oracle access to* $F(k,\cdot)$ *and the shifted key* $k\{\mathsf{Sh}_b\}$.
- *The adversary outputs a bit* b'.

We require that for all polynomial-time adversaries $\mathcal{A}$*, we have* $|\Pr[b = b'] - \frac{1}{2}| = \mathrm{negl}(\lambda)$.

3 Defining ED-Extractors

In this section, we give a formal definition of extractors for extractor-dependent sources (ED-Extractors) and provide some discussion on the various aspects of the definition.

Definition 3.1 (Extractor-Dependent Extraction)**.** *An extractor for* α*-entropy extractor-dependent sources (*α*-ED-Extractor) consists of two polynomial-time algorithms* $(\mathsf{SeedGen}, \mathsf{EDExt})$ *with the following syntax:*

- $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$ *is a randomized algorithm that generates* seed.
- $\mathsf{EDExt}(x, \mathsf{seed})$ *is a deterministic algorithm that takes a sample* $x \in \{0,1\}^n$*, together with* seed *and outputs a value* $y \in \{0,1\}^m$ *for some polynomial length parameters* $n = n(\lambda), m = m(\lambda)$.

Consider an adversarial source/distinguisher pair $(\mathcal{S}, \mathcal{D})$ *and define the following extraction experiment* $\mathsf{EDGame}^{\mathcal{S},\mathcal{D}}(1^\lambda)$*:*

- *Sample a random bit* $b \leftarrow \{0,1\}$ *and a random* $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- *Run* $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- *If* $b = 0$ *set* $r = \mathsf{EDExt}(x, \mathsf{seed})$ *else if* $b = 1$ *set* $r \leftarrow \{0,1\}^m$.
- *Let* $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

We say that $\mathcal{S}$ *is an* α*-legal extractor-dependent source if the following conditions hold:*

1. *The probability that* $\mathcal{S}$ *queries its oracle on the value* x *that it outputs is negligible.*
2. $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}) \geq \alpha(\lambda)$*, where* $X, \mathsf{SEED}, \mathsf{AUX}$ *denotes the joint distribution of the values* $x, \mathsf{seed}, \mathsf{aux}$ *in the above experiment.*

An α*-ED-Extractor is secure if for all* α*-legal polynomial-time sources* $\mathcal{S}$ *and all polynomial-time distinguishers* $\mathcal{D}$*, the above experiment satisfies*

$$\left|\Pr[b = b'] - \frac{1}{2}\right| = \mathrm{negl}(\lambda).$$

We can also define a weaker notion without auxiliary info *by restricting* aux *to be empty. We can also strengthen security to* computationally unbounded sources or distinguishers *by removing the restriction that the source or the distinguisher runs in polynomial time.*

Remark on the Legality Conditions. As we discussed in the introduction, the legality conditions above may not seem entirely intuitive on first look. For example, it may be unclear why we prohibit the source from querying the extractor on the value it outputs. Another undesirable aspect of definition is that the legality conditions are construction-dependent: in other words, a source may be legal for some constructions of the ED-Extractor but illegal for others since the entropy of the output may depend on the oracle queries. Ideally, the legality of the source could be checked independently of the construction. For these reasons, we can also consider an alternate, weaker, definition, which may be more intuitively compelling and does not suffer from the above issue. We say that source $\mathcal{S}$ is *α-super-legal* if:

- It can be written as $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ where $\mathcal{S}_1^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$ gets oracle access to the extractor and outputs some value $\mathsf{state} \in \{0,1\}^{p(\lambda)}$ for some polynomial p, and $\mathcal{S}_2(\mathsf{state})$ outputs x, aux without getting any further access to the extractor.
- For all choices of $\mathsf{state} \in \{0,1\}^{p(\lambda)}$ it holds that $H_\infty(X|\mathsf{AUX}) \geq \alpha(\lambda)$, where (X, AUX) are random variables for the output of $\mathcal{S}_2(\mathsf{state})$.

Note that "super-legality" is only a condition of $\mathcal{S}_2$ which does not have oracle access to the extractor, and is therefore construction-independent.

We claim that for any $\alpha(\lambda) = \omega(\log \lambda)$, every α-super-legal source $\mathcal{S}$ is also α-legal. Firstly, if $\mathcal{S}_1$ only makes polynomially many queries and has a non-negligible probability of querying the oracle on the value x that $\mathcal{S}_2$ outputs then there must be some value of state for which we can predict the value x that $\mathcal{S}_2(\mathsf{state})$ outputs with non-negligible probability. But this contradicts $H_\infty(X) \geq H_\infty(X|\mathsf{AUX}) \geq \omega(\log \lambda)$. Therefore $\mathcal{S}$ satisfies the first legality condition. Secondly, let STATE be a random variable for the value $\mathsf{state} \leftarrow \mathcal{S}_1^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$. Then SEED is independent of (X, AUX) if we condition on STATE. Therefore, $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}) \geq H_\infty(X|\mathsf{AUX}, \mathsf{STATE}) \geq \min_{\mathsf{state}} H_\infty(X_{\mathsf{state}}|\mathsf{AUX}_{\mathsf{state}}) \geq \alpha(\lambda)$ where $X_{\mathsf{state}}, \mathsf{AUX}_{\mathsf{state}}$ is the conditional distribution of X, AUX conditioned on $\mathsf{STATE} = \mathsf{state}$, which is just the distribution of the outputs of $\mathcal{S}_2(\mathsf{state})$. Therefore $\mathcal{S}$ satisfies the second legality condition.

As discussed in the introduction, the super-legality condition can also be interpreted very intuitively: we think of $\mathcal{S}_1$ as capturing all of the influence that prior extractor call can have on nature and $\mathcal{S}_2$ as modeling the entropic process that's responsible for generating x, aux. We chose to use "legality" rather than "super-legality" in our default definition since it makes the definition stronger and thus gives stronger positive results. We mention that (by simple inspection) all of our negative results also hold for the weaker definition using super-legality.

Remark about Conditioning on the Seed. Our legality condition in the formal definition requires that the entropy $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}) \geq \alpha(\lambda)$, where we condition on SEED. Note that we could remove this conditioning and have an alternate, stronger, definition where we only require $H_\infty(X|\mathsf{AUX}) \geq \alpha(\lambda)$. We observe that, assuming one-way functions, any α-ED-Extractor according to our definition can be converted into an $(\alpha' = \alpha + \lambda^\varepsilon)$-ED-Extractor according to the stronger definition for any constant $\varepsilon > 0$. The idea is that we can modify the $\mathsf{SeedGen}$ algorithm to only use λ^ε random bits by expanding them out using a PRG to get as many pseudorandom bits as needed by the original algorithm. By the security of the PRG, this change cannot harm ED-Extractor security. But now SEED comes from a domain of size only 2^{λ^ε} and therefore $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}) \geq H_\infty(X|\mathsf{AUX}) - \lambda^\varepsilon \geq \alpha' - \lambda^\varepsilon \geq \alpha$. Hence the new construction is an α'-ED-Extractor according to

the stronger definition. The take-away is that (as long as we're only considering polynomial-time distinguishers) it does not make much difference whether or not we condition on the seed in the definition.

Remark on Output Size. Note that if we have an α-ED-Extractor with output size $m = \lambda^\varepsilon$ for some constant $\varepsilon > 0$ then, assuming one-way functions, we can also construct an α-ED-Extractor for arbitrarily large output size $m = \lambda^c$ for any constant c just by using a pseudorandom generator (PRG) to expand the output. This holds as long as we're only considering polynomial-time distinguishers.

4 Security without Auxiliary Info

4.1 Construction from any PRF

We first show that every pseudorandom function (PRF) with a sufficiently high level of security is a good ED-Extractor in the setting without auxiliary info.

Theorem 4.1. *Let $F(\cdot,\cdot) : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^m$ be a pseudorandom function (PRF) with key-length $\ell = \ell(\lambda)$, input length $n = n(\lambda)$ and output length $m = m(\lambda)$, having security level $\sigma(\lambda) = 2^{m(\lambda)}\omega(\log\lambda)$. Define $(\mathsf{SeedGen}, \mathsf{EDExt})$ where $\mathsf{SeedGen}(1^\lambda)$ outputs $\mathsf{seed} \leftarrow \{0,1\}^{\ell(\lambda)}$ and $\mathsf{EDExt}(x, \mathsf{seed}) = F(\mathsf{seed}, x)$. Then $(\mathsf{SeedGen}, \mathsf{EDExt})$ is an α-ED Extractor without auxiliary info for any $\alpha \geq m + \omega(\log\lambda)$. Furthermore, it has security for unbounded distinguishers.*

Proof. Assume that $(\mathcal{S}, \mathcal{D})$ is some α-legal source and distinguisher pair with advantage $\varepsilon = \varepsilon(\lambda)$ in the ED-Exractor security game. Assume that $\mathcal{S}$ runs in polynomial time, but $\mathcal{D}$ can be unbounded. We define a polynomial-time adversary $\mathcal{A}$ that has $(\varepsilon^2 - \mathrm{negl}(\lambda))/2^m$ advantage in the PRF game. In particular, $\mathcal{A}^{\mathcal{O}(\cdot)}$ is given access to an oracle $\mathcal{O}$ and runs $\mathcal{S}^{\mathcal{O}(\cdot)}$ twice with independent randomness to derive two values x, x'. Then $\mathcal{A}^{\mathcal{O}(\cdot)}$ computes $r = \mathcal{O}(x), r' = \mathcal{O}(x')$. If $r = r'$, it outputs 1 else 0.

Firstly, consider the experiment where we sample $k \leftarrow \{0,1\}^\ell, x \leftarrow \mathcal{S}^{F(k,\cdot)}, r = F(k,x)$ and let K, R be the random variables for the values k, r respectively. Then the statistical distance $\mathsf{SD}((\ (K,R)\ ,\ (K,U_m)\ ) \geq \varepsilon$ since $\mathcal{D}$ distinguishes the two distributions with probability ε. Therefore, by Claim 2.6.1, we have $\mathsf{Col}(R|K) \geq \frac{1}{2^m}(1 + 4\varepsilon^2)$ where Col denotes the collision probability (Definition 2.6). It's easy to see that, by the definition of $\mathcal{A}$, we have $\Pr[\mathcal{A}^{F(k,\cdot)} = 1 \ : \ k \leftarrow \{0,1\}^\ell] = \mathsf{Col}(R|K) \geq \frac{1}{2^m}(1 + 4\varepsilon^2)$.

Secondly, consider the experiment where we sample $k \leftarrow \{0,1\}^\ell$ and then sample $x \leftarrow \mathcal{S}^{F(k,\cdot)}, x' \leftarrow \mathcal{S}^{F(k,\cdot)}$ by running $\mathcal{S}$ twice with independent randomness and let K, X, X' be the random variables for the value k, x, x' in the experiment. Since $\mathcal{S}$ is an α-legal source we know that:

- The probability that $\mathcal{S}$ queried the oracle on x during the first run or on x' during the second run is negligible.
- Since $H_\infty(X|K) = H_\infty(X'|K) \geq \alpha \geq m + \omega(\log\lambda)$, the probability that either (1) $\mathcal{S}$ queried the oracle on x' during the first run or (2) $\mathcal{S}$ queried the oracle on x during the second run or (3) $x = x'$ is bounded by $\mathrm{negl}(\lambda)/2^m$.

To summarize, in the above experiment, if we define the "bad event' that $x = x'$ or that the oracle is queried on one of x, x' during the course of the experiment, then the probability of the bad event is

at most $\text{negl}(\lambda)/2^m$. Now, consider the modified experiment where we sample $x \leftarrow \mathcal{S}^{U(\cdot)}, x' \leftarrow S^{U(\cdot)}$ and U is a truly random function. By $\sigma(\lambda)$ security of the PRF, the probability of the bad even occuring in the modified experiment is still be bounded by $\text{negl}(\lambda)/2^m$. If the bad event does not occur, then $r = U(x), r' = U(x')$ are random and independent values and therefore $\Pr[r = r'] = \frac{1}{2^m}$. This shows that $\Pr[\mathcal{A}^{U(\cdot)} = 1] \leq (1 + \text{negl}(\lambda))2^m$.

This shows that the advantage of $\mathcal{A}$ in the PRF security game is $(4\varepsilon^2(\lambda) - \text{negl}(\lambda))/2^m$ which must be $\leq 1/\sigma(\lambda) \leq \text{negl}(\lambda)/2^m$, by the $\sigma(\lambda)$ security of the PRF. Therefore $\varepsilon(\lambda) = \text{negl}(\lambda)$, which concludes the proof of the ED-Extractor security. □

Corollary 4.2. *Assuming the existence of sub-exponentially secure one-way functions, for any polynomial input size $n = n(\lambda)$ the following holds:*

- *For any polynomial output size $m = m(\lambda)$, there exists an α-ED Extractor in the setting without auxiliary info and with security for unbounded distinguishers as long as $\alpha \geq m + \omega(\log \lambda)$.*
- *For any constant $\varepsilon > 0$ and any polynomial output size $m = m(\lambda)$, there exists an α-ED Extractor in the setting without auxiliary info and security for polynomial-time distinguishers as long as $\alpha \geq \lambda^\varepsilon$.*

Proof. We note that sub-exponentially secure one-way functions imply the existence of PRFs with security level $2^{p(\lambda)}$ for any polynomial p (by making the key sufficiency large). Therefore the first part of the corollary follows directly from the preceding Theorem. The second part follows by using a pseudorandom generator (PRG) to expand the output-size of the ED-Extractor as discussed in the Remark on Output Size in Section 3. □

4.2 Necessity of One-Way Functions

Theorem 4.3. *For any input length $n = n(\lambda)$, the existence of an $(\alpha = n - 1)$-ED-Extractor, even without auxiliary info and even with output length $m = 1$, implies the existence of one-way functions. Furthermore, such extractors cannot be secure for computationally unbounded sources, even if we restrict to polynomial-time distinguishers.*

Proof. Let $(\mathsf{SeedGen}, \mathsf{EDExt})$ be an ED Extractor as in the theorem statement. Assume $\mathsf{SeedGen}(1^\lambda)$ uses at most $\ell = \ell(\lambda)$ bits of randomness and let $q = 7\ell + \lambda$. Define the function $f(r, x_1, \ldots, x_q) = (x_1, \ldots, x_q, y_1, \ldots, y_q)$, which takes as input a uniformly random $r \in \{0,1\}^\ell$ and $x_i \in \{0,1\}^n$ and computes $\mathsf{seed} = \mathsf{SeedGen}(1^\lambda; r)$ and $y_i = \mathsf{EDExt}(\mathsf{seed}, x_i)$ for $i \in [q]$. Then we claim that f is a one-way function.

Assume by way of contradiction that a polynomial-size adversary $\mathcal{A}$ breaks the one-wayness of f with non-negligible probability. We define a source $\mathcal{S}^{\mathsf{EDExt}(\mathsf{seed},\cdot)}$ as follows:

1. Choose $x_1, \ldots, x_q$ uniformly at random form $\{0,1\}^n$. Query the oracle to learn $y_i = \mathsf{EDExt}(\mathsf{seed}, x_i)$ for each $i \in [q]$.
2. Run $\mathcal{A}(x_q, \ldots, x_q, y_1, \ldots, y_q)$ and get some value $(r', x'_1, \ldots, x'_q)$.
3. Test if $f(r', x_q, \ldots, x'_q) = (x_1, \ldots, x_q, y_1, \ldots, y_q)$. If not, output a uniformly random $x_0^* \leftarrow \{0,1\}^n$ and halt. Else continue.

4. Compute $\mathsf{seed}' = \mathsf{SeedGen}(1^\lambda; r')$. Choose a random $x_1^* \leftarrow \{0,1\}^n$ and if $\mathsf{EDExt}(\mathsf{seed}', x_1^*) = 0$ output x_1^* and halt. Else continue.

5. Choose a random $x_2^* \leftarrow \{0,1\}^n$ and output it.

We define a corresponding distinguisher $D(\mathsf{seed}, r)$, which outputs r. We claim that the pair $(\mathcal{S}, \mathcal{D})$ breaks the $(\alpha = n-1)$-ED-Extractor security.

Firstly, we claim that $\mathcal{S}$ is an $(\alpha = n-1)$-legal source. It is easy to see that the probability of it outputting a value x that it previously queried is negligible since it outputs one of x_0^*, x_1^*, x_2^* each of which is individually uniformly random and independent of the prior queries. To analyze entropy, let us fix any choice of the values of $\mathsf{seed}, x_1, \ldots, x_q$ and randomness of $\mathcal{A}$ and let X be the random variable for the output of $\mathcal{S}$ in the above experiment. We argue that, even for any choice of the fixed values, it holds that $H_\infty(X) \geq n-1$, which proves the claim. The fixed values determine whether the test in line 3 passes or fails. If it fails, then X is uniformly random and so $H_\infty(X) = n$. If it passes, then let us define the variable V which is 0 if x is output in line 4 and 1 if it is output in line 5. Let us define the value $P_0 = |\{x : \mathsf{EDExt}(x, \mathsf{seed}) = 0\}|$. Then we have

$$\begin{aligned}
\max_x \Pr[X = x] &= \max_x \left(\Pr[X = x | V = 0] \Pr[V = 0] + \Pr[X = x | V = 1] \Pr[V = 1]\right) \\
&\leq \frac{1}{P_0} \cdot \frac{P_0}{2^n} + \frac{1}{2^n}(1 - \frac{1}{P_0}) \\
&\leq 2^{-(n-1)}
\end{aligned}$$

and therefore $H_\infty(X) \geq n-1$.

Next, we analyze the success probability of the pair $(\mathcal{S}, \mathcal{D})$ in the ED-Extractor security game. If the challenger chooses the challenge bit $b = 1$ then, since r is uniformly random, we have $\Pr[b' = 1] = \frac{1}{2}$. On the other hand, let's analyze the security game when the challenge chooses the bit $b = 0$. Assume that the adversary $\mathcal{A}$ breaks the security of the one-way function f with some non-negligible probability $\varepsilon = \varepsilon(\lambda)$. Then $\varepsilon(\lambda) \geq 1/p(\lambda)$ for some polynomial p and for infinitely many values of λ. We define several events in the context of the ED-Extractor security game with the particular sampler defined above:

FAIR: Let's call a seed *biased* if $\Pr_{x \leftarrow \{0,1\}^n}[\mathsf{EDExt}(\mathsf{seed}, x) = 0] \leq \frac{1}{2} - \delta$ where we set $\delta := \frac{1}{20p}$. Let's define the event FAIR to occur if the seed is not biased. Since we assumed that the ED-Extractor is secure, it must be the case that probability that a random seed is biased is negligible (otherwise the sampler that outputs a random x and the distinguisher that tests if the seed is biased and if so outputs r else outputs random would break security). Therefore, $\Pr[\mathsf{FAIR}] = 1 - \mathrm{negl}(\lambda)$.

INV: Let this be the event that the test in line 3 of the execution of $\mathcal{S}$ succeeds, meaning that $\mathcal{A}$ succeeded to invert correctly. By definition $\Pr[\mathsf{INV}] = \varepsilon$.

CLOSE: Let this be the event that for the value seed' computed in line 4, it holds that

$$\Pr_{x \leftarrow \{0,1\}^n}[\mathsf{EDExt}(x, \mathsf{seed}) = \mathsf{EDExt}(x, \mathsf{seed}')] \geq .9$$

where, if the process terminates before line 4, we define $\mathsf{seed}' = \mathsf{seed}$. If CLOSE does not occur, it means that there exists some seed' for which $\Pr_{x \leftarrow \{0,1\}^n}[\mathsf{EDExt}(x, \mathsf{seed}) = \mathsf{EDExt}(x, \mathsf{seed}')] <$

.9 but for all $i \in [q]$ it holds that $\mathsf{EDExt}(x_i, \mathsf{seed}) = \mathsf{EDExt}(x_i, \mathsf{seed}')$. The probability of this happening for any fixed seed' is $.9^q \leq .9^{7\ell+\lambda} \leq 2^{-\ell}\mathrm{negl}(\lambda)$. By taking a union bound over all 2^ℓ values of seed' the probability that some such seed' exists is negligible and therefore $\Pr[\mathsf{CLOSE}] \geq 1 - \mathrm{negl}(\lambda)$.

For simplicity, we also define the event $\mathsf{IFC} = \mathsf{INV} \wedge \mathsf{FAIR} \wedge \mathsf{CLOSE}$. When $b = 0$ we therefore have:

$$\begin{aligned}
\Pr[b' = 0] &\geq \Pr[b' = 0 \wedge \mathsf{INV}] + \Pr[b' = 0 \wedge \neg\mathsf{INV}] \\
&\geq \Pr[b' = 0 \wedge \mathsf{INV} \wedge \mathsf{FAIR} \wedge \mathsf{CLOSE}] + \Pr[b' = 0 \wedge \neg\mathsf{INV} \wedge \mathsf{FAIR}] \\
&\geq \Pr[b' = 0|\mathsf{IFC}](\Pr[\mathsf{INV}] - \Pr[\neg\mathsf{FAIR}] - \Pr[\neg\mathsf{CLOSE}]) \\
&\quad + \Pr[b' = 0|\neg\mathsf{INV} \wedge \mathsf{FAIR}](\Pr[\neg\mathsf{INV}] - \Pr[\neg\mathsf{FAIR}]) \\
&\geq \Pr[b' = 0|\mathsf{IFC}](\varepsilon - \mathrm{negl}(\lambda)) + \Pr[b' = 0|\neg\mathsf{INV} \wedge \mathsf{FAIR}](1 - \varepsilon - \mathrm{negl}(\lambda)) \\
&\geq \Pr[b' = 0|\mathsf{IFC}](\varepsilon - \mathrm{negl}(\lambda)) + \left(\frac{1}{2} - \delta\right)(1 - \varepsilon - \mathrm{negl}(\lambda))
\end{aligned}$$

To analyze $\Pr[b' = 0|\mathsf{IFC}]$ let us fix all randomness z of the experiment except for the choice of x_1^*, x_2^*, such that this fixing makes the event IFC occurs. Let IFC_z be the event that the randomness takes on this value. For any such choice, let E_1 be the event that $\mathsf{EDExt}(x_1^*, \mathsf{seed}) = 0$, let E_1' be the event that $\mathsf{EDExt}(x_1^*, \mathsf{seed}') = 0$, let A be the even that $\mathsf{EDExt}(x_1^*, \mathsf{seed}) = \mathsf{EDExt}(x_1^*, \mathsf{seed}')$ and let E_2 be the event that $\mathsf{EDExt}(\mathsf{seed}, x_2^*) = 0$, where the randomness is only over the choice of x_1^*, x_2^*. Since we conditioned on CLOSE we have $\Pr[A] \geq .9$. Since we conditioned on FAIR we have $\Pr[E_1] \geq (1/2 - \delta), \Pr[E_2] \geq (1/2 - \delta)$. Therefore, for any such choice of randomness z we have:

$$\begin{aligned}
\Pr[b' = 0|\mathsf{IFC}_z] &= \Pr[E_1 \wedge E_1'] + \Pr[E_2 \wedge \neg E_1'] \\
&= \Pr[A \wedge E_1'] + \Pr[E_2]\,(1 - \Pr[E_1']) \\
&\geq \Pr[E_1'] - \Pr[\neg A] + \left(\frac{1}{2} - \delta\right)(1 - \Pr[E_1']) \\
&\geq \frac{1}{2} - \delta - .1 + \frac{1}{2}\Pr[E_1'] \\
&\geq \frac{1}{2} - \delta - .1 + \frac{1}{2}(\Pr[E_1] - \Pr[\neg A]) \\
&\geq \frac{1}{2} - \delta - .1 + \frac{1}{2}(\frac{1}{2} - \delta - .1) \\
&\geq .6 - \frac{3}{2}\delta
\end{aligned}$$

which also implies that $\Pr[b' = 0|\mathsf{IFC}] \geq .6 - \frac{3}{2}\delta$. Combining, we have:

$$\begin{aligned}
\Pr[b' = 0] &\geq \left(.6 - \frac{3}{2}\delta\right)(\varepsilon - \mathrm{negl}(\lambda)) + \left(\frac{1}{2} - \delta\right)(1 - \varepsilon - \mathrm{negl}(\lambda)) \\
&\geq \frac{1}{2} - \delta + \varepsilon(.1 - \delta/2) - \mathrm{negl}(\lambda) \\
&\geq \frac{1}{2} + \varepsilon/10 - (3/2)\delta - \mathrm{negl}(\lambda) \\
&\geq \frac{1}{2} + \frac{1}{10p(\lambda)} - \frac{3}{40p(\lambda)} - \mathrm{negl}(\lambda) \\
&\geq \frac{1}{2} + \frac{1}{40p(\lambda)} - \mathrm{negl}(\lambda)
\end{aligned}$$

for infinitely many values of λ. Therefore $\Pr[b' = b] - \frac{1}{2}$ is non-neglgible, which leads to a contradiction and hence f must be one-way.

For the second part of the theorem, note that we showed how to convert an inverter for f into a source $\mathcal{S}$ together with an efficient disitinguisher $\mathcal{D}$ that break ED-Extractor security. Since an inefficient inverter for f always exists, it means that there exists an inefficient source $\mathcal{S}$ and an efficient distinguisher $\mathcal{D}$ that break the security of the ED-Extractor. □

5 Security with Auxiliary Info

5.1 Construction via Constrained PRFs

We now show how to construct an ED-Extractor in the setting with auxiliary info, using constrained PRFs (Definition 2.8) and standard seeded extractors (Definition 2.2).

Construction. Let $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^\ell$ be an (α', ε)-seeded extractor for some lengths $n = n(\lambda), d = d(\lambda), \ell = \ell(\lambda)$ and some $\alpha' = \alpha'(\lambda), \varepsilon = \varepsilon(\lambda)$. Further let Ext also be a universal hash function. Let $(F, \mathsf{Constrain})$ be a constrained PRF with input length n and output length $m = m(\lambda)$ for the class of constraints $\mathcal{C} = \{C_{s,u}\}_{s \in \{0,1\}^d, u \in \{0,1\}^\ell}$ where $C_{s,u}(x) = 1$ iff $\mathsf{Ext}(x; s) = u$. We construct an ED-Extractor $(\mathsf{SeedGen}, \mathsf{EDExt})$ as follows:

- $\mathsf{SeedGen}(1^\lambda)$: Choose a random $k \leftarrow \{0,1\}^\lambda$. Choose a random $s \leftarrow \{0,1\}^d, u \leftarrow \{0,1\}^\ell$ and let $C_{s,u} \in \mathcal{C}$ be the corresponding constraint. Let $k\{C_{s,u}\} \leftarrow \mathsf{Constrain}(k, C_{s,u})$. Output $\mathsf{seed} = k\{C_{s,u}\}$.
- $\mathsf{EDExt}(x, \mathsf{seed})$: Output $F(k\{C_{s,u}\}, x)$.

Note that F always outputs some value, even if x is in the constrained set. Without loss of generality, we can assume that the constrained key $k\{C_{s,u}\}$ reveals s, u in the clear and that, $F(k\{C_{s,u}\}, x)$ outputs 0^m whenever $C_{s,u}(x) = 1$.

Theorem 5.1. *Assuming the constrained PRF has no-constrained-evaluation security, the construction above is an α-entropy secure ED-Extractor for $\alpha = \alpha' + m$, as long as the parameters satisfy $\ell(\lambda) = \omega(\log \lambda)$, and $\varepsilon(\lambda) = \mathrm{negl}(\lambda)$.*

Proof. Our proof of security follows by a sequence of hybrid games:

Hybrid 0: This is the game $\mathsf{EDGame}^{\mathcal{S},\mathcal{D}}(1^\lambda)$ with a source $\mathcal{S}$ and a distinguisher $\mathcal{D}$ as in Definition 3.1. The game proceeds as follows:

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$. The latter consists of sampling $k \leftarrow \{0,1\}^\lambda, s \leftarrow \{0,1\}^d, u \leftarrow \{0,1\}^\ell$, $k\{C_{s,u}\} \leftarrow \mathsf{Constrain}(k, C_{s,u})$ and setting $\mathsf{seed} = k\{C_{s,u}\}$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrid 1: In this game, instead of giving the source $\mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}$ access to the oracle $\mathsf{EDExt}(\cdot,\mathsf{seed}) = F(k\{C_{s,u}\},\cdot)$, we replace it with the oracle $F(k,\cdot)$ using the unconstrained key k. Furthermore, if $b=0$, instead of setting $r = \mathsf{EDExt}(x,\mathsf{seed}) = F(k\{C_{s,u}\},x)$, we now set $r = F(k,x)$. In detail, the hybrid is defined as follows:

1. Sample a random bit $b \leftarrow \{0,1\}$ and a random $k \leftarrow \{0,1\}^\lambda$.
2. Run $(x,\mathsf{aux}) \leftarrow \mathcal{S}^{F(k,\cdot)}(1^\lambda)$.
3. If $b=0$ set $r = F(k,x)$ else if $b=1$ set $r \leftarrow \{0,1\}^m$. Choose $s \leftarrow \{0,1\}^d, u \leftarrow \{0,1\}^\ell$ and $\mathsf{seed} \leftarrow \mathsf{Constrain}(k, C_{s,u})$.
4. Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrids 0 and 1 are indistinguishable. The only time Hybrid 0 differs from Hybrid 1 is if in Hybrid 0 either: (A) some oracle query or the final output x produced by $\mathcal{S}$ satisfy $\mathsf{Ext}(x;s) = u$, or (B) some oracle query or the final output x produced by $\mathcal{S}$ satisfy $C_{s,u}(x) = 0 \wedge F(k,x) \neq F(k\{C_{s,u}\},x)$. Since u is uniformly random, the probability of (A) happening when $\mathcal{S}$ makes q queries is at most $(q+1)/2^\ell$ which is negligible. By the correctness of the constrained PRF, the probability of (B) happening is also negligible.

Hybrid 2: This is the same as Hybrid 1, except that we give the source access to an oracle $\mathsf{EDExt}(\cdot;\mathsf{seed}')$ where $\mathsf{seed}' = k\{C_{s',u'}\} \leftarrow \mathsf{Constrain}(k, C_{s',u'})$ is a constrained PRF key for random and independent values s', u'. In detail, the hybrid is defined as follows:

1. Sample a random bit $b \leftarrow \{0,1\}$ and a random $k \leftarrow \{0,1\}^\lambda$. Choose $s' \leftarrow \{0,1\}^d, u' \leftarrow \{0,1\}^\ell$ and $\mathsf{seed}' \leftarrow \mathsf{Constrain}(k, C_{s',u'})$.
2. Run $(x,\mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed}')}(1^\lambda)$.
3. If $b=0$ set $r = F(k,x)$ else if $b=1$ set $r \leftarrow \{0,1\}^m$. Choose $s \leftarrow \{0,1\}^d, u \leftarrow \{0,1\}^\ell$ and $\mathsf{seed} \leftarrow \mathsf{Constrain}(k, C_{s,u})$.
4. Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrids 1 and 2 are statistically close. The only time Hybrid 1 differs from Hybrid 2 is if in Hybrid 2 either: (A) some oracle query x_i produced by $\mathcal{S}$ satisfies $\mathsf{Ext}(x_i;s') = u'$, or (B) some oracle query x_i produced by $\mathcal{S}$ satisfy $C_{s',u'}(x) = 0 \wedge F(k,x) \neq F(k\{C_{s',u'}\},x)$. Since u' is uniformly random, the probability of (A) happening when $\mathcal{S}$ makes q queries is at most $q/2^\ell$ which is negligible. By the correctness of the constrained PRF, the probability of (B) happening is also negligible.

Hybrid 3: This is the same as Hybrid 2, except that in step 3, instead of choosing $u \leftarrow \{0,1\}^\ell$ we now set $u = \mathsf{Ext}(x;s)$. In detail, the hybrid is defined as follows:

1. Sample a random bit $b \leftarrow \{0,1\}$ and a random $k \leftarrow \{0,1\}^\lambda$. Choose $s' \leftarrow \{0,1\}^d, u' \leftarrow \{0,1\}^\ell$ and $\mathsf{seed}' \leftarrow \mathsf{Constrain}(k, C_{s',u'})$.
2. Run $(x,\mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed}')}(1^\lambda)$.
3. If $b=0$ set $r = F(k,x)$ else if $b=1$ set $r \leftarrow \{0,1\}^m$. Choose $s \leftarrow \{0,1\}^d$ $u = \mathsf{Ext}(x;s)$ and $\mathsf{seed} \leftarrow \mathsf{Constrain}(k, C_{s,u})$.
4. Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrids 2 and 3 are statistically close if Ext is an (α, ε)-extractor. To argue this, let us use capital letters to denote random variables for the corresponding values in the experiment. Firstly, note that the view of the source $\mathcal{S}$ in hybrid 2 is identically distributed to that of hybrid 0.[3] Therefore, we can rely on the legality to $\mathcal{S}$ (which is defined relative to the distribution of hybrid 0) to argue that $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}') \geq \alpha$. By Lemma 2.1, we also have $H_\infty(X|\mathsf{AUX}, \mathsf{SEED}', R) \geq \alpha - m \geq \alpha'$. Lastly since K is independent of X when conditioned on SEED', R, we also have $H_\infty(X|\mathsf{AUX}, K, R) \geq \alpha'$. Therefore, by the security of the extractor, $U = \mathsf{Ext}(X; S)$ is statistically close to a uniformly random and independent U even given AUX, K, R, S. Lastly, since the view of $\mathcal{D}$ in hybrids 2 and 3 is a function of AUX, K, R, S, U where $U = \mathsf{Ext}(X; S)$ in hybrid 3 and U is uniform/independent in hybrid 2, the two hybrids are statistically close.

Hybrid 4: This is the same as Hybrid 3, except that we switch back from giving $\mathcal{S}$ oracle access to $\mathsf{EDExt}(\cdot, \mathsf{seed}')$ to giving it access to the unconstrained PRF $F(k, \cdot)$. In detail, the hybrid is defined as follows:

1. Sample a random bit $b \leftarrow \{0,1\}$ and a random $k \leftarrow \{0,1\}^\lambda$.
2. Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{F(k,\cdot)}(1^\lambda)$.
3. If $b = 0$ set $r = F(k, x)$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$. Choose $s \leftarrow \{0,1\}^d$ $u = \mathsf{Ext}(x; s)$ and $\mathsf{seed} \leftarrow \mathsf{Constrain}(k, C_{s,u})$.
4. Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrids 3 and 4 are indistinguishable by the same argument as the indistinguishability of hybrid 1 and 2.

Advantage in Hybrid 4: We now claim that in Hybrid 4, the advantage $|\Pr[b = b'] - \frac{1}{2}|$ is negligible by the no-constrained-evaluation security of the constrained PRF. In particular, we define a reduction that runs $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{F(k,\cdot)}(1^\lambda)$ by making queries to its PRF oracle. The reduction chooses $s \leftarrow \{0,1\}^d$, sets $u = \mathsf{Ext}(x; s)$ and gives the constraint $C_{s,u}$ together with the value x to its challenger. Since $\mathcal{S}$ is a legal source, x was never queried by the oracle and, by the definition of the constraint, we have $C_{s,u}(x) = 1$. Secondly, since $\mathsf{Ext}(\cdot; s)$ is a universal hash function, the probability that of any of the previous queries x_i made by $\mathcal{S}$ satisfy $\mathsf{Ext}(x_i; s) = \mathsf{Ext}(x; s)$ is also negligible. Therefore, our reduction makes no constrained-evaluation queries to the PRF.

So, the reduction is a legal attacker in the no-constrained-evaluation security game of constrained PRF. The reduction receives a value r, which is either $F(k, x)$ or uniform, along with a constrained key $k\{C_{s,u}\}$. It sets $\mathsf{seed} = k\{C_{s,u}\}$ and outputs the bit $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$. The advantage of the reduction in the constrained PRF security game is exactly the same as that of the adversary in hybrid 3, and therefore the latter is negligible.

Since the advantage in hybrid 3 is negligible and hybrid 3 is indistinguishable from hybrid 0, the advantage in hybrid 0 must be negligible as well. This proves the theorem. □

Corollary 5.2. *Under the sub-exponential security of either the LWE assumption or the DDHI assumption in an arbitrary prime-order group, there exists an ED-Extractor for α-entropy sources*

[3] This was the reason for introducing hybrid 2 rather than directly going from 1 to 3.

with auxiliary info, for any $\alpha = \lambda^{\Omega(1)}$ and with any polynomial input length n and output length m. Security holds against polynomial-time sources and distinguishers.

Proof. The work of [BV15] construct selectively secure constrained PRFs for all circuits from LWE. We can then use complexity leveraging to get adaptive security by assuming sub-exponential LWE. The results of [AMN+18] constructs no-evaluation secure PRFs for NC1 from the DDHI assumption in arbitrary prime-order groups (the also construct selectively secure PRFs from the DDHI assumption in specific groups). We then use complexity leveraging to get no-constrained-evaluation security under sub-exponential DDHI, as discussed in the remarks after Definition 2.8.

We use an extractor with output length $\alpha/4$ which is secure for entropy $\alpha' = \alpha/2$ with $\varepsilon = 2^{-(\alpha/8)} = \mathrm{negl}(\lambda)$. We combine this with a constrained PRF with output length $m = \alpha/2$ which ensures $\alpha \geq \alpha' + m$. This gives us an ED-Extractor with output length $\alpha/2 = \lambda^{\Omega(1)}$. We can then use a PRG to then get arbitrarily large polynomial output size as discussed in the Remark on Output Size in Section 3. □

5.2 Construction via Shift-Hiding Shiftable Functions

Our second construction builds an ED-extractor in the setting with auxiliary info, using shift-hiding shiftable PRFs (Definition 2.9). Compared to the construction in Section 5.1, the one in this section achieves security under polynomial hardness assumptions, in particular the polynomial hardness of LWE with a subexponential approximation ratio. In addition, the proof of security appears to follow a fundamentally different route, as we explain below.

Construction. Let $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be an (α', ε)-seeded extractor for some lengths $n = n(\lambda), d = d(\lambda), m = m(\lambda)$ and some $\alpha' = \alpha'(\lambda), \varepsilon = \varepsilon(\lambda)$. Let $(F, \mathsf{Shift}, \mathsf{Round})$ be a shift-hiding shiftable PRF with input length n and output length $m = m(\lambda)$ for the class of shift functions $\mathsf{Sh} = \{Sh_s(x) = \mathsf{Ext}(s, x)\}$. We construct an ED-Extractor $(\mathsf{SeedGen}, \mathsf{EDExt})$ as follows:

- $\mathsf{SeedGen}(1^\lambda)$: Choose a random $k \leftarrow \{0,1\}^\lambda$. Let Z denote the zero circuit which on all inputs outputs 0^m. Let $k\{Z\} \leftarrow \mathsf{Shift}(k, Z)$. Output $\mathsf{seed} = k\{Z\}$.
- $\mathsf{EDExt}(x, \mathsf{seed})$: Output $F(k\{Z\}, x)$.

Theorem 5.3. *Assuming that (F, Shift) is a shift-hiding shiftable PRF, the construction above is an α-entropy secure ED-Extractor for $\alpha = \alpha' + m$ as long as $\epsilon(\lambda) = \mathrm{negl}(\lambda)$.*

Proof. Assume for contradiction that the ED-Extractor adversary (A_1, A_2) has a non-negligible success advantage $\epsilon = \epsilon(\lambda)$ in the ED-Extractor game. We proceed in the following sequence of hybrid experiments.

Hybrid 0: This is the ED-extractor game with the construction $(\mathsf{SeedGen}, \mathsf{EDExt})$. In more detail, the game proceeds as follows.

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$. The latter consists of sampling $k \leftarrow \{0,1\}^\lambda$, computing $k\{Z\} \leftarrow \mathsf{Shift}(k, Z)$ and setting $\mathsf{seed} = k\{Z\}$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot, \mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.

- Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

By assumption, then,

$$\Pr[(A_1, A_2) \text{ wins in Hybrid } 0] = 1/2 + \epsilon$$

Hybrid 1: Let $\mathsf{H} = \{h : \{0,1\}^n \to \{0,1\}^\ell\}$ be a family of pairwise independent hash functions. Define the property

$$P_{\ell,\beta,h}(x) = \begin{cases} 1 & \text{if } h(x) = \beta \\ 0 & \text{otherwise} \end{cases}$$

where $\beta \in \{0,1\}^\ell$ and $h \in \mathsf{H}$. When $\beta \leftarrow \{0,1\}^\ell$ and $h \leftarrow \mathsf{H}$ are chosen uniformly at random from the respective sets, the following holds.

Claim 5.3.1. *Assume $\beta \leftarrow \{0,1\}^\ell$ and $h \leftarrow \mathsf{H}$. Denote by Good the event that for all the q queries x_i made by A_1, $P_{\ell,\beta,h}(x_i) = 0$, and for A_1's output x, $P_{\ell,\beta,h}(x) = 1$.*

$$\Pr[\mathsf{Good}] \geq (1 - q/2^\ell) \cdot 1/2^\ell$$

where the probability is over the coins of β and h and those of A_1.

Proof. We know from the legality condition on A_1 that x is different from all the queries. Then,

$$\begin{aligned} \Pr[\mathsf{Good}] &= \Pr[\text{for all } i \in [q] \;\; h(x_i) \neq \beta \wedge h(x) = \beta] \\ &= \Pr[\text{for all } i \in [q] \;\; h(x_i) \neq \beta] \cdot \Pr[h(x) = \beta] \\ &\geq (1 - q/2^\ell) \cdot 1/2^\ell \end{aligned}$$

where the second equality is because of pairwise indepedence and the inequality follows by an application of the union bound. □

Hybrid 1 proceeds exactly like Hybrid 0 except that if Good does not happen, the game ignores the output of A_2 and replaces it with a random bit. In more detail, the game proceeds as follows.

- Sample a random $\beta \leftarrow \{0,1\}^\ell$ and $h \leftarrow H$. Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$. The latter consists of sampling $k \leftarrow \{0,1\}^\lambda$, computing $k\{Z\} \leftarrow \mathsf{Shift}(k, Z)$ and setting $\mathsf{seed} = k\{Z\}$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$. Let $\{x_i\}$ be the queries generated by $\mathcal{S}$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $P_{\ell,\beta,h}(x_i) = 1$ for some $i \in [q]$ or $P_{\ell,\beta,h}(x) = 0$, set $b' \leftarrow \{0,1\}$ to be a uniformly random bit. Otherwise, let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

We then have the following claim.

Claim 5.3.2. *Set $\ell = \log q + 1$. We have:*

$$\Pr[(A_1, A_2) \text{ wins in Hybrid } 1] \geq 1/2 + \epsilon/4q$$

Proof. First note that $\Pr[\mathsf{Good}] \geq (1 - q/2^\ell) \cdot 1/2^\ell = 1/4q$. Let Win denote the event that (A_1, A_2) wins in Hybrid 1.

$$\begin{aligned}\Pr[\mathsf{Win}] &= \Pr[\mathsf{Win} \mid \mathsf{Good}] \Pr[\mathsf{Good}] + \Pr[\mathsf{Win} \mid \overline{\mathsf{Good}}] \Pr[\overline{\mathsf{Good}}] \\ &\geq (1/2 + \epsilon) \cdot \Pr[\mathsf{Good}] + 1/2 \cdot (1 - \Pr[\mathsf{Good}]) \\ &\geq 1/2 + \epsilon/4q\end{aligned}$$

where the first inequality comes from the fact that $\Pr[\mathsf{Win} \mid \mathsf{Good}] = \Pr[\mathsf{Win}]$ and $\Pr[\mathsf{Win} \mid \overline{\mathsf{Good}}] = 1/2$, and the second comes from the bound on $\Pr[\mathsf{Good}]$. □

Hybrid 2: Let $\mathsf{Sh}_{\ell,\beta,h,s}(x) = P_{\ell,\beta,h}(x) \cdot \mathsf{Ext}(s, x)$ be a shift function where $s \leftarrow \{0,1\}^d$ is chosen uniformly at random. That is,

$$\mathsf{Sh}_{\ell,\beta,h,s}(x) = \begin{cases} 0 & \text{if } P_{\ell,\beta,h}(x) = 0 \\ \mathsf{Ext}(s, x) & \text{otherwise} \end{cases}$$

Instead of $K\{Z\}$, use $K\{\mathsf{Sh}_{\ell,\beta,h,s}\}$ throughout the game. In more detail, the game proceeds as follows.

- Sample a random $s \leftarrow \{0,1\}^d$. Sample random $\beta \leftarrow \{0,1\}^\ell$ and $h \leftarrow \mathsf{H}$. Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$. The latter consists of sampling $k \leftarrow \{0,1\}^\lambda$, computing $k\{\mathsf{Sh}_{\ell,\beta,h,s}\} \leftarrow \mathsf{Shift}(k, \mathsf{Sh}_{\ell,\beta,h,s})$ and setting $\mathsf{seed} = k\{\mathsf{Sh}_{\ell,\beta,h,s}\}$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$. Let $\{x_i\}$ be the queries generated by $\mathcal{S}$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $P_{\ell,\beta,h}(x_i) = 1$ for some $i \in [q]$ or $P_{\ell,\beta,h}(x) = 0$, set $b' \leftarrow \{0,1\}$ to be a uniformly random bit. Otherwise, let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

By shift-hiding we have:

Claim 5.3.3.

$$\Pr[(A_1, A_2) \textit{ wins in Hybrid } 2] = \Pr[(A_1, A_2) \textit{ wins in Hybrid } 1] - \mathrm{negl}(\lambda)$$

Hybrid 3: Use $K\{Z\}$ to answer the queries of A_1 but then use $K\{\mathsf{Sh}_{\ell,\beta,h,s}\}$ to give to A_2 and to compute the output of the extractor. In more detail, the game proceeds as follows.

- Sample a random $s \leftarrow \{0,1\}^d$. Sample random $\beta \leftarrow \{0,1\}^\ell$ and $h \leftarrow \mathsf{H}$. Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$. The latter consists of sampling $k \leftarrow \{0,1\}^\lambda$, computing $k\{\mathsf{Sh}_{\ell,\beta,h,s}\} \leftarrow \mathsf{Shift}(k, \mathsf{Sh}_{\ell,\beta,h,s})$ and setting $\mathsf{seed} = k\{\mathsf{Sh}_{\ell,\beta,h,s}\}$ and $\mathsf{seed}' = k\{Z\}$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed}')}(1^\lambda)$. Let $\{x_i\}$ be the queries generated by $\mathcal{S}$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $P_{\ell,\beta,h}(x_i) = 1$ for some $i \in [q]$ or $P_{\ell,\beta,h}(x) = 0$, set $b' \leftarrow \{0,1\}$ to be a uniformly random bit. Otherwise, let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Claim 5.3.4.

$$\Pr[(A_1, A_2) \textit{ wins in Hybrid } 3] = \Pr[(A_1, A_2) \textit{ wins in Hybrid } 2]$$

Proof. If Good happens, then the view of A_1 is the same in Hybrids 2 and 3. On the other hand, if Good does not happen, then the output of the experiment is a uniformly random bit in both cases. The claim follows. □

To finish the proof, note that in Hybrid 3, the view of A_1 is the same as in the real game so the entropy condition on its output X holds. That is:

$$H(X|\mathsf{SEED}', \mathsf{AUX}) \geq \alpha \text{ and } H(X|\mathsf{SEED}', \mathsf{AUX}, F_K(X)) \geq \alpha - m \geq \alpha'$$

Furthermore, in Hybrids 2 and 3, S is random and independent of X since S is chosen after A_1 outputs X. Therefore, A_2 gets $\mathsf{SEED}, \mathsf{AUX}, \mathsf{Round}(F(K, X) + \mathsf{Ext}(S, X))$ (with high probability), which can be replaced by a uniformly random string by the security of the extractor. Thus,

$$\Pr[(A_1, A_2) \text{ wins in Hybrid } 3] \leq 1/2 + \mathrm{negl}(\lambda)$$

which, combined with the preceding claims, contradicts the assumption that (A_1, A_2) wins in Hybrid 0 with non-negligible advantage. □

Corollary 5.4. *Under the LWE assumption, there exists an ED-Extractor for α-entropy sources with auxiliary info, for any $\alpha = \lambda^{\Omega(1)}$ and with any polynomial input length n and output length m. Security holds against polynomial-time sources and distinguishers.*

Proof. For any $\alpha = \lambda^{\Omega(1)}$, instantiate Ext with optimal seeded extractors with output length $m = \alpha/3$ and entropy requirement $\alpha' \geq m + \omega(\log \lambda) \geq (2/3)\alpha$, eg the leftover hash lemma extractor [ILL89]. Then, using Theorem 5.3, instantiated with the shift-hiding shiftable functions of [PS18] based on LWE with subexponential approximation ratio gives us an ED-extractor for entropy $m + \alpha' = \alpha$ as needed, but the output length is only $m = \alpha/3 = \lambda^{\Omega(1)}$. However, we can then use a PRG to then get arbitrarily large polynomial output size as discussed in the Remark on Output Size in Section 3. □

5.3 Construction via Lossy Functions

We present yet another construction of ED-Extractors in the setting with auxiliary input via *lossy functions* (LFs), which are equivalent to lossy trapdoor functions (LTDFs) [PW08], but without requiring a trapdoor. We give two constructions of ED-Extractors via LFs: the first one is simpler but relies on quasi-polynomial security, while the second one builds on the first, and is more complex, but only needs standard polynomial security. Our first construction of ED-Extractors from LFs is motivated by the works of [BHK11, GKK19], which used a similar construction to solve seemingly unrelated problems. While the construction and some high-level intuition behind it are analogous to those of the above prior works, the proof is fairly different.

We begin by defining the notion of *lossy functions* (LFs). Then, we extend it to a notion of *cumulatively all-lossy-but-one (CALBO)* functions and show how to construct CALBOs from LFs. The notion of CALBOs with a *trapdoor* was also explored in a recent work of [CPW19], where they gave a construction using DDH and indistinguishability obfuscation (iO). In contrast, our notion of

CALBOs without a trapdoor appears to be significantly easier to achieve and we do not rely on iO. Our notion of CALBOs was also implicit in the works of [BHK11, GKK19], where they essentially gave a construction under the DDH assumption in specific groups (the quadratic residues mod a prime p). Here we give show that our notion of CALBOs follows from standard LFs and therefore can be instantiated under many standard assumptions, such as DDH (in arbitrary groups), LWe and DCR.

Definition 5.5 (Lossy Functions (LFs))**.** *An (n, m, ℓ)-family of Lossy-Functions (LFs), for some polynomial input length $n = n(\lambda)$, output length $m = m(\lambda)$ and leakage $\ell = \ell(\lambda)$ consists of two PPT algorithms (KeyGen, F) with the following syntax:*

- $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, b)$*: On input the security parameter λ and $b \in \{0,1\}$, outputs pk. If $b = 0$, we say that pk is generated in "lossy" mode and if $b = 1$ it is generated in "injective" mode.*
- $y = F_{\mathsf{pk}}(x)$*: A deterministic algorithm that takes as input pk along with $x \in \{0,1\}^n$ and outputs $y \in \{0,1\}^m$.*

We require the following two properties:

Injectivity: *With overwhelming probability over the choice of $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, 1)$ the function $F_{\mathsf{pk}} : \{0,1\}^n \to \{0,1\}^m$ is injective.*

Lossyness : *With overwhelming probability over the choice of $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, 0)$ the function F_{pk} has an image of size $|\{F_{\mathsf{pk}}(x) : x \in \{0,1\}^n\}| \leq 2^\ell$.*

Indisitnguishability: *The distributions $\mathsf{KeyGen}(1^\lambda, 0)$ and $\mathsf{KeyGen}(1^\lambda, 1)$ are computationally indistinguishable. (We say that the LF has security level $\sigma = \sigma(\lambda)$ if every PPT distinguisher has distinguishing advantage at most $\mu(\lambda) \leq 1/\sigma(\lambda)$.)*

We extend the notion of LFs to *Cumulatively All-Lossy-But-One (CALBO)* Functions. Such functions also take as input a tag. The public key pk is created with a special tag^* for which the function is "almost" injective, meaning that each output y has at most some small number of pre-images under this tag. On the other hand, for any other $\mathsf{tag} \neq \mathsf{tag}^*$, the function is lossy. Moreover, the leakage does not accumulate – even seeing the output at the function at all lossy tags $\mathsf{tag} \neq \mathsf{tag}^*$ only reveals some small amount of information about the input. Lastly, in the case of CALBO, we insist on the output length being the same as the input length.

Definition 5.6 (Cumulatively All-Lossy-But-One (CALBO) Functions)**.** *An (n, t, ℓ, d)-family of Cumulatively All-Lossy-But-One (CALBO) functions, for some polynomial input length $n = n(\lambda)$, tag length $t = t(\lambda)$, leakage $\ell = \ell(\lambda)$, and max collisions $d = d(\lambda)$ consists of two PPT algorithms (KeyGen, F) with the following syntax:*

- $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$*: On input the security parameter λ and $\mathsf{tag}^* \in \{0,1\}^t$, outputs pk.*
- $y = F_{\mathsf{pk},\mathsf{tag}}(x)$*: A deterministic algorithm that takes as input pk along with $\mathsf{tag} \in \{0,1\}^t$ and $x \in \{0,1\}^n$. It outputs y.*

We require the following two properties:

Almost Injectivity: *For any $\mathsf{tag}^* \in \{0,1\}^t$, with overwhelming probability over the choice of $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$ we have that for all $y \in \{0,1\}^n$, it holds that $|F^{-1}_{\mathsf{pk},\mathsf{tag}^*}(y)| \leq d$.*

Cummulative Lossyness : *For any* $\mathsf{tag}* \in \{0,1\}^t$*, any* $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$*, if we define* $F_{\mathsf{pk}}^{lossy}(x) := \{(\mathsf{tag}, F_{\mathsf{pk},\mathsf{tag}}(x)) : \mathsf{tag} \neq \mathsf{tag}^*\}$ *to be the set of evaluations of all lossy tags cumulatively, then the image of* $F_{\mathsf{pk},lossy}$ *is of size at most* $|\{F_{\mathsf{pk}}^{lossy}(x) : x \in \{0,1\}^n\}| \leq 2^\ell$*.*

Indisitnguishability: *For any* $\mathsf{tag}_0^*, \mathsf{tag}_1^* \in \{0,1\}^t$*, the distributions* $\mathsf{KeyGen}(1^\lambda, \mathsf{tag}_0^*)$ *and* $\mathsf{KeyGen}(1^\lambda, \mathsf{tag}_1^*)$ *are computationally indistinguishable. (We say that the CALBO has security level* $\sigma = \sigma(\lambda)$ *if every PPT distinguisher has distinguishing advantage at most* $\mu(\lambda) \leq 1/\sigma(\lambda)$*.)*

Note that the lossiness property can be equivalently stated as follows: there exists some (potentially inefficient) functions $g : \{0,1\}^n \to \{0,1\}^\ell$ and $h : \{0,1\}^\ell \times \{0,1\}^t \to \{0,1\}^n$ such that for all $x \in \{0,1\}^n$ and $\mathsf{tag} \in \{0,1\}^t$ with $\mathsf{tag} \neq \mathsf{tag}^*$ we have $F_{\mathsf{pk},\mathsf{tag}}(x) = h(\mathsf{tag}, g(x))$.

5.3.1 Constructing CALBOs

We now show how to construct "Cumulatively All-Lossy-But-One Functions" (CALBOs) from arbitrary Lossy Functions. In doing so, we need to solve two issues:

- Firstly, the output size m of an LF can be much larger than the input size n while in a CALBO the output size needs to equal the input size.
- Secondly, a CALBO has a large tag space while an LF can be thought of as only having two tags $b = 0$ and $b = 1$.

We solve the first issue by using k-wise independent hash functions to compress the output. We solve the second issue by using composing many lossy functions together.

Construction. Let (KeyGen, F) be an (n, m, ℓ)-family of Lossy-Functions (LFs), for some polynomial input length $n = n(\lambda)$, output length $m = m(\lambda)$ and leakage $\ell = \ell(\lambda)$. Let $\mathsf{H} = \{h : \{0,1\}^m \times \{0,1\}^n\}$ be a collection of d-wise independent hash functions. For any polynomial tag length t, we define the CALBO (KeyGen', F') as follows:

- $\mathsf{pk} \leftarrow \mathsf{KeyGen}'(1^\lambda, \mathsf{tag}^*)$: For $i = 1, \ldots, t$ generate $h_i \leftarrow \mathsf{H}$, $\mathsf{pk}_{i,\mathsf{tag}_i^*} \leftarrow \mathsf{KeyGen}(1^\lambda, 1)$ and $\mathsf{pk}_{i,1-\mathsf{tag}_i^*} \leftarrow \mathsf{KeyGen}(1^\lambda, 0)$ where tag_i^* denotes the i'th bit of tag^*. Output
$$\mathsf{pk} = (\{\mathsf{pk}_{i,b}\}_{i\in[t],b\in\{0,1\}}, \{h_i\}_{i\in[t]}).$$
- $y = F'_{\mathsf{pk},\mathsf{tag}}(x)$. If we let tag_i denote the i'th bit of tag, we define:
$$F'_{\mathsf{pk},\mathsf{tag}}(x) = h_t \circ F_{\mathsf{pk}_{t,\mathsf{tag}_t}} \circ h_{t-1} \circ F_{\mathsf{pk}_{t-1,\mathsf{tag}_{t-1}}} \cdots \circ h_1 \circ F_{\mathsf{pk}_{1,\mathsf{tag}_1}}(x)$$
To facilitate notation, we define the function $F'_{\mathsf{pk},w}(x)$ inductively for all $w \in \{0,1\}^{\leq t}$. If $|w| = 0$ then $F'_{\mathsf{pk},w}(x) = x$. For any $w = b||w'$ with $|w| = i$ and $b \in \{0,1\}$, we define $F'_{\mathsf{pk},w}(x) = h_i(F_{\mathsf{pk}_{i,b}}(F'_{\mathsf{pk},w'}(x)))$.

Theorem 5.7. *For any polynomial* t*, if* (KeyGen, F) *is an* (n, m, ℓ)*-LF then the above construction* (KeyGen', F') *with* $d = \max\{2et, n + \lambda\}$ *is a* (n, t, ℓ', d)*-CALBO where* $\ell' = \ell \cdot t$*.*

Proof. The indistinguishability property of the CALBO follows immediately from that of the underlying LF.

To argue the cumulative lossiness of the CALBO, let $\mathsf{pk} \leftarrow \mathsf{KeyGen}'(1^\lambda, \mathsf{tag}^*)$ and, for each $i \in [t]$, let $w_i = (1 - \mathsf{tag}_i^*)||\mathsf{tag}_{i-1}^*||\mathsf{tag}_{i-2}^*||\cdots$ and $z_i = F'_{\mathsf{pk},w_i}(x)$. Then, for any $\mathsf{tag} \neq \mathsf{tag}^*$, the value $F'_{\mathsf{pk},\mathsf{tag}}(x)$ is completely determined $z_1, \ldots, z_t$. In particular, if i is the smallest value such that $\mathsf{tag}_i \neq \mathsf{tag}_i^*$ then $F'_{\mathsf{pk},\mathsf{tag}}(x)$ is completely determined by z_i. On the other hand, with overwhelming probability over the choice of $\mathsf{pk}_{i,1-\mathsf{tag}_i^*}$ there are only 2^ℓ possible values for each z_i and therefore $2^{t\ell}$ possible values for $(z_1, \ldots, z_t)$. Therefore, with overwhelming probability, the image of $F_{\mathsf{pk}}^{lossy}(x) := \left\{(\mathsf{tag}, F'_{\mathsf{pk},\mathsf{tag}}(x)) \ : \ \mathsf{tag} \neq \mathsf{tag}^*\right\}$ is of size at most $2t\ell$.

To argue the almost injectivity of the CALBO, fix some $\mathsf{tag}^* \in \{0,1\}^t$ and let $w_i = \mathsf{tag}_i^*|| \ldots ||\mathsf{tag}_1^*$. Let $\mathsf{pk} = (\{\mathsf{pk}_{i,b}\}_{i\in[t],b\in\{0,1\}}, \{h_i\}_{i\in[t]}) \leftarrow \mathsf{KeyGen}'(1^\lambda, \mathsf{tag}^*)$. First note that, with overwhelming probability, $F_{\mathsf{pk}_{i,\mathsf{tag}_i^*}}$ is injective for each $i \in [t]$. If this happens, we claim that, with overwhelming probability over the choice of $h_1, \ldots, h_t$ for all $y \in \{0,1\}^n$, it holds that $|F^{-1}_{\mathsf{pk},\mathsf{tag}^*}(y)| \leq d$. The only way that the latter may not happen is if there exists some set $S = \{x_1, \ldots, x_d\} \subseteq \{0,1\}^n$ and some values $I = (i_2, \ldots, i_d) \in [t]$ such that each $i_j \leq t$ is the smallest i for which $\widehat{F}_{\mathsf{pk},w_i}(x_j) = \widehat{F}_{\mathsf{pk},w_i}(x_1)$. For any S, I the probability of this happening (by d-wise independence) is $2^{-n(d-1)}$. By the union bound, the probability of such values S, I existing is the

$$\begin{aligned}
&\leq \binom{2^n}{d} t^{d-1} 2^{-n(d-1)} \\
&\leq (e2^n/d)^d t^{d-1} (1/2^n)^{d-1} \leq (e2^n/d)(et/d)^{d-1} \leq 2^n (1/2)^{n+\lambda-1} \\
&\leq 2^{-\lambda+1} = \mathrm{negl}(\lambda).
\end{aligned}$$

□

The works of [PW08, FGK+10, AKPW13] show how to construct "very" lossy functions. In particular, under any of (1) the decision-linear assumption (DLIN) which is implied by DDH, (2) the decisional composite residuosity (DCR) assumption, or the learning with error (LWE) assumptions they achieve the following: for any arbitrarily large input $n = \mathrm{poly}(\lambda)$ and arbitrarily small leakage $\ell = \lambda^{\Omega(1)}$, there exists an (n, m, ℓ)-LF for some polynomial $m = \mathrm{poly}(\lambda)$.

Corollary 5.8. *Under any of (DDH, DLIN, LWE, DCR), the following holds. For any polynomial $n = n(\lambda), t = t(\lambda)$ and any $\varepsilon > 0$ exist (n, t, ℓ, d) CALBOs with $\ell = t \cdot \lambda^\varepsilon$ and $d = \lambda^{O(1)}$.*

5.3.2 ED-Extractor from CALBOs: Quasi-Polynomial Security Loss

Construction. Let (KeyGen, F) be an (n, t, ℓ, d)-CALBO. Let $\mathsf{2Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ be a (e_1, e_2, δ) strong-2-source extractor. Let $f : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^t$ be a PRF family. We construct an ED-Extractor $(\mathsf{SeedGen}, \mathsf{EDExt})$ as follows:

- $\mathsf{SeedGen}(1^\lambda)$: Sample a random $\mathsf{tag}^* \leftarrow \{0,1\}^t$, $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$, $s \leftarrow \{0,1\}^n$, $k \leftarrow \{0,1\}^\lambda$. Output $\mathsf{seed} = (s, \mathsf{pk}, k)$.
- $\mathsf{EDExt}(x, \mathsf{seed})$: Let $\mathsf{tag} = f(k, x)$, $y = F_{\mathsf{pk},\mathsf{tag}}(s)$. Output $\mathsf{2Ext}(x, y)$.

Intuition. Intuitively, the PRF ensures that if the sample x differs from all prior queries x_i to the extractor then the corresponding $\mathsf{tag} = f(k, x)$ differs from all $\mathsf{tag}_i = f(k, x_i)$. Since tag^* is hidden by pk, there is a (roughly) 2^{-t} chance that $\mathsf{tag} = \mathsf{tag}^*$ is the injective tag and, since the adversary cannot tell if this happens, his success probability must remain high even if it does. But in this case the value $y = F_{\mathsf{pk},\mathsf{tag}}(s)$ has a lot of entropy even conditioned on the entire view of the source, which only depends on $F^{lossy}_{\mathsf{pk}}(s)$. Moreover y and x are independent of each other conditioned on $F^{lossy}_{\mathsf{pk}}(s)$. Therefore, we can rely on the strong 2-source extractor security to guarantee

Theorem 5.9. *The above construction is an α-entropy secure ED-Extractor as long as the tag size is $t = \omega(\log \lambda)$, $e_1 = \alpha - 1$, $e_2 = n - \ell - \log d - 1$, $\delta = 2^{-t}\mathrm{negl}(\lambda)$ and the CALBO has security level $2^t\lambda^{\omega(1)}$.*

Proof. The proof follows via a sequence of hybrids.

Hybrid 0. This is the ED-Extractor game with a source/distinguisher $\mathcal{S}, \mathcal{D}$. The game proceeds as follows:

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} = (s, \mathsf{pk}, k) \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Let $\Pr[b = b'] \geq \frac{1}{2} + \epsilon(\lambda)$.

Hybrid 1. This is the same as Hybrid 0 except that, if $\mathcal{S}$ outputs some value x such that $f(k, x) = f(k, x_i)$ for some previous query x_i then output a random bit b'.

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} = (s, \mathsf{pk}, k) \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $f(k, x) = f(k, x_i)$ for some value x_i on which $\mathcal{S}$ previously queried its oracle then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

By the legality condition, we know that the sample x differs from all other queries x_i. Since the PRF has output length $t = \omega(\log \lambda)$, the probability that $f(k, x) = f(k, x_i)$ for some query x_i is negligible (otherwise the sampler $\mathcal{S}$ breaks PRF security). Therefore we have

$\Pr[b = b'] \geq \frac{1}{2} + \epsilon(\lambda) - \mathrm{negl}(\lambda)$.

Hybrid 2. In hybrid 2, we sample random $\mathsf{tag}' \leftarrow \{0,1\}^t$ at the very beginning of the game. (This is done independently of the choice of $\mathsf{tag}^* \leftarrow \{0,1\}^t$ that is used to select $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$ during the run of $\mathsf{SeedGen}$.) If for the value x output by $\mathcal{S}$ we have $f(k, x) \neq \mathsf{tag}'$ then we output a random bit b', and otherwise we proceed as in Hybrid 1. In more detail, the hybrid proceeds as follows.

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} = (s, \mathsf{pk}, k) \leftarrow \mathsf{SeedGen}(1^\lambda)$. Sample a random $\mathsf{tag}' \leftarrow \{0,1\}^t$.

- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $f(k,x) \neq \mathsf{tag}'$ or $f(k,x) = f(k,x_i)$ for some value x_i on which $\mathcal{S}$ previously queried its oracle then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Since tag' is random and independent of other values in the experiment, the probability of it being correct is exactly 2^{-t}. Therefore:

$$\begin{aligned}\Pr[b = b'] &= |\Pr[b = b' \wedge f(k,x) = \mathsf{tag}'] + \Pr[b = b' \wedge f(k,x) \neq \mathsf{tag}'] \\ &\geq (\frac{1}{2} + \epsilon(\lambda) - \mathrm{negl}(\lambda))2^{-t} + (1 - 2^{-t})\frac{1}{2} \\ &\geq \frac{1}{2} + 2^{-t}(\epsilon(\lambda) - \mathrm{negl}(\lambda))\end{aligned}$$

Hybrid 3. In this hybrid, instead of choosing tag' randomly, we set it to be the injective tag tag^*. In particular, the hybrid is defined as follows:

- Sample a random bit $b \leftarrow \{0,1\}$ and $\mathsf{tag}' \leftarrow \{0,1\}^t$. Sample the $\mathsf{seed} = (s, \mathsf{pk}, k)$ by selecting $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ and $s \leftarrow \{0,1\}^n$, $k \leftarrow \{0,1\}^\lambda$ as before.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $f(k,x) \neq \mathsf{tag}'$ or $f(k,x) = f(k,x_i)$ for some value x_i on which $\mathcal{S}$ previously queried its oracle then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrid 3 and Hybrid 2 are indistinguishable by the security of the CALBO. In particular, we can think of both Hybrids 2,3 as choosing two tags $\mathsf{tag}^*, \mathsf{tag}'$ uniformly at random in the beginning of the game, but only tag' is used later in the game to perform the check in the 4th line, while tag^* is not used anywhere else. The only difference between Hybrid 2 and 3 is then that we choose $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*)$ in Hybrid 2 and $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ in Hybrid 3. These are indistinguishable by CALBO security. By choosing the CALBO to have security level $2^t\lambda^{\omega(1)}$, the distinguishing advantage becomes $2^{-t}\mathrm{negl}(\lambda)$ and therefore, in hybrid 3, we have:

$$\Pr[b = b'] \geq \frac{1}{2} + 2^{-t}(\epsilon(\lambda) - \mathrm{negl}(\lambda)) - 2^{-t}\mathrm{negl}(\lambda) \geq \frac{1}{2} + 2^{-t}(\epsilon(\lambda) - \mathrm{negl}(\lambda)).$$

Hybrid 4. In this hybrid, we always choose r as uniform. In particular, the hybrid is defined as follows:

- Sample a random bit $b \leftarrow \{0,1\}$ and $\mathsf{tag}' \leftarrow \{0,1\}^t$. Sample the $\mathsf{seed} = (s, \mathsf{pk}, k)$ by selecting $\mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ and $s \leftarrow \{0,1\}^n$, $k \leftarrow \{0,1\}^\lambda$ as before.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- Set $r \leftarrow \{0,1\}^m$.

- If $f(k,x) \neq \mathsf{tag}'$ or $f(k,x) = f(k,x_i)$ for some value x_i on which $\mathcal{S}$ previously queried its oracle then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Fix any choice of $\mathsf{tag}', k, \mathsf{pk}$ in the experiment for the remainder of the analysis.

Let $S, X, \mathsf{AUX}, \mathsf{SEED} = (S, \mathsf{pk}, k)$ be random variables for the corresponding values in the experiment. Let $Y = F_{\mathsf{pk},\mathsf{tag}'}(S)$. Let E be a random variable which is 0 if "$f(k,X) \neq \mathsf{tag}'$ or $f(k,X) = f(k,X_i)$ for some value X_i on which $\mathcal{S}$ previously queried its oracle" and 1 otherwise. Define $\hat{X}$ and $\hat{Y}$ to be equal to X, Y respectively if $E = 1$ and to be uniformly random and independent over $\{0,1\}^n$ otherwise. Let $R = 2\mathsf{Ext}(\hat{X}, \hat{Y})$.

Claim 5.9.1. *There is a randomized function $\mathcal{A}$ such that* $\Pr[A(S, \mathsf{AUX}, E, R) = 1] = \Pr[b = b' \text{ in hybrid } 3]$ *and* $\Pr[A(S, \mathsf{AUX}, E, U_m) = 1] = \Pr[b = b' \text{ in hybrid } 4]$.

Proof. We define $A(s, \mathsf{aux}, e, z)$ to choose a random $b \leftarrow \{0,1\}$. If $e = 0$, select $b' \leftarrow \{0,1\}$. If $e = 1$ let $b' = \mathcal{D}(1^\lambda, (s, \mathsf{pk}, k), \mathsf{aux}, z')$ where $z' = z$ if $b = 0$ and $z' \leftarrow \{0,1\}^m$ otherwise. Output 1 iff $b = b'$.

It's easy to see the claim follows. In particular, the only time $A(S, \mathsf{AUX}, E, R)$ makes use of $R = 2\mathsf{Ext}(\hat{X}, \hat{Y})$ is when $E = 1$ in which case $R = \mathsf{EDExt}(X, \mathsf{SEED})$. □

Claim 5.9.2. *The statistical distance* $\mathsf{SD}(\ (S, \mathsf{AUX}, E, R), (S, \mathsf{AUX}, E, U_m)) \leq \delta$.

Proof. Let us define an additional random variable $L = F^{lossy}_{\mathsf{pk}}(S)$. First, we claim that

$$\begin{aligned}
\mathsf{SD}(\ (S, \mathsf{AUX}, E, R), (S, \mathsf{AUX}, E, U_m)) &\leq \mathsf{SD}(\ (S, \mathsf{AUX}, L, E, R), (S, \mathsf{AUX}, L, E, U_m)) \\
&\leq \mathsf{SD}(\ (Y, \mathsf{AUX}, L, E, R), (Y, \mathsf{AUX}, L, E, U_m)) \\
&\leq \mathsf{SD}(\ (\hat{Y}, \mathsf{AUX}, L, E, R), (\hat{Y}, \mathsf{AUX}, L, E, U_m))
\end{aligned}$$

The first inequality follows since adding information can only increase the statistical distance. The second inequality follows since S is independent of (AUX, E, R) conditioned on (Y, L) and therefore we can think of S as a randomized function of (Y, L). The last inequality follows since Y is independent of R conditioned on $(\hat{Y}, \mathsf{AUX}, L, E)$ and therefore we can think of Y as a randomized function of $(\hat{Y}, \mathsf{AUX}, L, E)$.

Next, we note that $\hat{X}, \hat{Y}$ are independent conditioned on (AUX, L, E).

Furthermore we have:

$$\begin{aligned}
H_\infty(\hat{X} | L, E, \mathsf{AUX}) &\geq H_\infty(X | L, E, \mathsf{AUX}) \\
&\geq H_\infty(X | L, \mathsf{AUX}) - 1 \\
&\geq H_\infty(X | \mathsf{SEED}, \mathsf{AUX}) - 1 \\
&\geq \alpha - 1
\end{aligned}$$

The first inequality follows since $\hat{X}$ is harder to predict than X, and the third inequality follows since $L = F^{lossy}_{\mathsf{pk}}(S)$ is fully determined by $\mathsf{SEED} + (S, \mathsf{pk}, k)$.

We also have:

$$\begin{aligned} H_\infty(\hat{Y}|L,E,\mathsf{AUX}) &\geq H_\infty(\hat{Y}|L,E) \\ &\geq H_\infty(\hat{Y}) - \ell - 1 \\ &\geq H_\infty(Y) - \ell - 1 \\ &\geq n - \log d - \ell - 1 \end{aligned}$$

The first inequality follows since $\hat{Y}$ is independent of AUX conditioned on L, E and the third inequality follows since $Y = F_{\mathsf{pk},\mathsf{tag}'}(S)$ where S is uniform over $\{0,1\}^n$ and $F_{\mathsf{pk},\mathsf{tag}'}$ is at most d-to-1 so $\max_y \Pr[Y = y] = \max_y \Pr[S \in F^{-1}_{\mathsf{pk},\mathsf{tag}'}(y)] \leq d/2^n$.

Therefore, by the security of the strong two source extractor, we have

$$\mathsf{SD}(\ (\hat{Y}, \mathsf{AUX}, L, E, R = 2\mathsf{Ext}(\hat{X}, \hat{Y})), (\hat{Y}, \mathsf{AUX}, L, E, U_m)) \leq \delta$$

which concludes the proof of the claim. □

Combining the two claims, we have:

$$\frac{1}{2} = \Pr[b = b' \text{ in hybrid 4}] \geq \Pr[b = b' \text{ in hybrid 3}] - \delta \geq \frac{1}{2} + 2^{-t}(\epsilon(\lambda) - \mathrm{negl}(\lambda))$$

Therefore, we must have $\epsilon(\lambda) = \mathrm{negl}(\lambda)$, which completes the proof. □

Corollary 5.10. *Under the quasi-polynomial security of any of (DDH, DLIN, LWE, DCR), there exists an ED-Extractor for α-entropy sources with auxiliary info, for any $\alpha = \lambda^{\Omega(1)}$ and with any polynomial input length n and output length m. Security holds against polynomial-time sources and distinguishers.*

Proof. We can assume $n >\geq \lambda$ (we can always apply the construction on shorter inputs by appending 0s to the input). We rely on the CALBOs of Corollary 5.8. In particular, we set the tag length $t = \log^2 \lambda$. we can rely on an (n, t, ℓ, d) CALBOs with $\ell = t \cdot \lambda^{.1} \leq \lambda^{.2}$ and $d = \lambda^{O(1)}$ and quasi-polynomial security level $\lambda^{2 \log \lambda}$. We then rely on the Raz 2-source extractor (Theorem 2.5) with $e_1 = \alpha - 1 = \lambda^{\Omega(1)}$, $e_2 = n - \ell - \log d - 1 \geq n - n^{.2} - O(\log n) \geq (1 - o(1))n$, $\delta = 2^{-\lambda^{\Omega(1)}} = 2^{-t}\mathrm{negl}(\lambda)$ and $m = \lambda^{\Omega(1)}$. This satisfies the conditions of the theorem and therefore gives use the claimed result, except that the output size is only $m = \lambda^{\Omega(1)}$. We can then apply a PRG to the output to get arbitrarily large output size. □

5.3.3 ED-Extractor from CALBOs: Polynomial Security Loss

Construction. Let i_{max}, j_{max}, v be some parameters (in the security parameter λ). For $i = 1, \ldots, i_{max}$ and let (KeyGen^i, F^i) be an (n, i, ℓ, d)-CALBO with tags of length i. Let $f : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^v$ be a PRF family and let $G : \{0,1\}^v \to \{0,1\}^w$ be a PRG with input size v output size $w = j_{max} i_{max}(i_{max} + 1)/2$. Let $\mathsf{2Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ be a (e_1, e_2, δ) strong-2-source extractor (see Definition 2.4). We construct an ED-Extractor $(\mathsf{SeedGen}, \mathsf{EDExt})$ as follows:

- $\mathsf{SeedGen}(1^\lambda)$: For $i \in [i_{max}]$ and $j \in [j_{max}]$: sample a random $\mathsf{tag}^*_{i,j} \leftarrow \{0,1\}^i$, $\mathsf{pk}_{i,j} \leftarrow \mathsf{KeyGen}^i(1^\lambda, \mathsf{tag}^*_{i,j})$, $s_{i,j} \leftarrow \{0,1\}^n$, $k \leftarrow \{0,1\}^\lambda$. Output $\mathsf{seed} = (\{s_{i,j}, \mathsf{pk}_{i,j}\}, k)$.

- $\mathsf{EDExt}(x, \mathsf{seed})$: Let $z = f(k, x)$. Parse $\{\mathsf{tag}_{i,j}\} = G(z)$ as a collection of tags for $i \in [i_{\max}], j \in [j_{max}]$ with $|\mathsf{tag}_{i,j}| = i$. For $i \in [i_{max}]$ and $j \in [j_{max}]$: let $y_{i,j} = F^i_{\mathsf{pk}_{i,j},\mathsf{tag}_{i,j}}(s_{i,j})$. Let $y := \bigoplus_{i,j} y_{i,j}$. Output $\mathsf{2Ext}(x, y)$.

Intuition. Assume the source makes q queries to the oracle. Our reduction will then target the value $i^* = \lceil \log q \rceil + 1 \in [i_{max}]$. There are $2^{i^*} \geq 2q$ possible tags of length i^*. For each $j \in [j_{max}]$, the source's queries to EDExt produce at most q different tags in position (i^*, j), and therefore at most only $1/2$ of all possible tags that could be in that position. This means that, with overwhelming probability, there is some j^* such that the tag in position (i^*, j^*) produced by the computation of EDExt on the actual sample x^* was never produced in that position by any prior query. We can then guess j^* as well as the tag of length i^* that's produced in position (i^*, j^*) on the sample x^*. This guess is correct with inverse polynomial probability $1/(j_{max}2^{i^*}) \leq 1/(4j_{max}q)$. Now we choose the public key pk_{i^*,j^*} so that the guessed tag is the injective one. This is indistinguishable by the CALBO. The rest of the proof proceeds analgously to the previous one, where we rely on the fact that, if our guess is correct, then s_{i^*,j^*} (and hence also y_{i^*,j^*}) have very high entropy, even conditioned on the view of the source.

Theorem 5.11. *The above construction is an α-entropy secure ED-Extractor as long as $i_{max}, j_{max} = \omega(\log \lambda)$, $e_1 = \alpha - v - 1$, $e_2 = n - \ell - v - \log d - 1$, $\delta = \mathrm{negl}(\lambda)$.*

Proof. Let $\mathcal{S}, \mathcal{D}$ be some adversarial source/distinguisher pair. Assume that the sampler $\mathcal{S}$ makes at most q queries to its oracle and let $i^* = \lceil \log q \rceil + 1$. The proof follows via a sequence of hybrids.

Hybrid 0. This is the ED-Extractor game with a source/distinguisher $\mathcal{S}, \mathcal{D}$. The game proceeds as follows:

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- Let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Let $\Pr[b = b'] \geq \frac{1}{2} + \epsilon(\lambda)$ for some ϵ.

Hybrid 1. We define the event BAD to occur if, for all $j \in [j_{max}]$, the value of $\mathsf{tag}_{i^*,j}$ produced in position (i^*, j) during the computation $\mathsf{EDExt}(x, \mathsf{seed})$ already also appeared in position (i^*, j) during the computation of $\mathsf{EDExt}(x_a, \mathsf{seed})$ for some prior query x_a. In Hybrid 1, we choose b' uniformly at random if BAD occurs.

- Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If BAD occurs, then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

We claim that the probability of BAD occurring is negligible. By the legality condition, we know that the sample x differs from all prior queries x_a made by $\mathcal{S}$. Furthermore, $\mathcal{S}$ only made q queries to the oracle before outputting a sample. Therefore, these queries produced

at most q distinct tags in each position (i^*, j), out of a total of $2^{i^*} \geq 2q$ possible tags. If the values $\mathsf{tag}_{i^*,j}$ corresponding to the sample x were truly random, the probability that for each $j \in [j_{max}]$ such tag also already appeared in position (i^*, j) for a previous query would therefore be at most $(q/2^{i^*})^{-j_{max}} \leq 2^{-j_{max}} = \mathrm{negl}(\lambda)$. Therefore, if $\mathcal{S}$ can trigger the event BAD with non-negligible probability it can break the security of the PRF or the PRG.

This shows that, in Hybrid 1, we have

$\Pr[b = b'] \geq \frac{1}{2} + \epsilon(\lambda) - \mathrm{negl}(\lambda)$.

Hybrid 2. In hybrid 2, we sample random value $j^* \leftarrow [j_{max}], \mathsf{tag}' \leftarrow \{0,1\}^{i^*}$ at the very beginning of the game. If $\neg\mathsf{BAD}$ occurs then there exists some $j \in [j_{max}]$, such that the value of $\mathsf{tag}_{i^*,j}$ produced in position (i^*, j) during the computation $\mathsf{EDExt}(x, \mathsf{seed})$ never appeared in position (i^*, j) during the computation of $\mathsf{EDExt}(x_a, \mathsf{seed})$ for any prior query x_a. Let j' be the minimal j for which the above holds. We define the event GUESS to occur if $\neg\mathsf{BAD}$ occurs and $j^* = j'$ and $\mathsf{tag}' = \mathsf{tag}_{i^*,j^*}$. We define Hybrid 2 to choose b' uniformly at random if either BAD or $\neg\mathsf{GUESS}$ occurs.

- Sample a random $j^* \leftarrow [j_{max}], \mathsf{tag}' \leftarrow \{0,1\}^{i^*}$. Sample a random bit $b \leftarrow \{0,1\}$ and a random $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $\mathsf{BAD} \vee \neg\mathsf{GUESS}$ occurs, then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

If we let $b'_0 = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$ in the above experiment and set $p = j_{max}2^{i^*}$, then we have:

$$
\begin{aligned}
\Pr[b = b'] &\geq \Pr[b = b' \wedge \neg\mathsf{BAD}] \\
&\geq \Pr[b = b' \wedge \neg\mathsf{BAD} \wedge \mathsf{GUESS}] + \Pr[b = b' \wedge \neg\mathsf{BAD} \wedge \neg\mathsf{GUESS}] \\
&\geq \Pr[b = b'_0 \wedge \neg\mathsf{BAD} \wedge \mathsf{GUESS}] + \frac{1}{2}\Pr[\neg\mathsf{BAD} \wedge \neg\mathsf{GUESS}] \\
&\geq \Pr[b = b'_0 \wedge \neg\mathsf{BAD}]\Pr[\mathsf{GUESS} \mid b = b'_0 \wedge \neg\mathsf{BAD}] \\
&\quad + \frac{1}{2}(1 - \Pr[\mathsf{GUESS} \mid \neg\mathsf{BAD}])\Pr[\neg\mathsf{BAD}] \\
&\geq (\Pr[b = b'_0] - \Pr[\mathsf{BAD}])(1/p) + \frac{1}{2}(1 - 1/p)(1 - \Pr[\mathsf{BAD}]) \\
&\geq (\frac{1}{2} + \epsilon(\lambda) - \mathrm{negl}(\lambda))(1/p) + \frac{1}{2}(1 - 1/p)(1 - \mathrm{negl}(\lambda)) \\
&\geq \frac{1}{2} + \epsilon(\lambda)/p - \mathrm{negl}(\lambda)
\end{aligned}
$$

We rely on the fact that, if we condition on $\neg\mathsf{BAD}$ then, $\Pr[\mathsf{GUESS}] = 1/p$ where the probability is only over the choice of j^*, tag' and independent of any other randomness in the game.

Hybrid 3. In this hybrid, we set the injective tag $\mathsf{tag}^*_{i^*,j^*}$ to be tag'. In particular, the hybrid is defined as follows:

- Sample a random $j^* \leftarrow [j_{max}], \mathsf{tag}' \leftarrow \{0,1\}^{i^*}$. Sample a random bit $b \leftarrow \{0,1\}$. Sample the $\mathsf{seed} = (\{s_{i,j}, \mathsf{pk}_{i,j}\}, k)$ by selecting $\mathsf{pk}_{i^*,j^*} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ and choosing all other components in the same manner as previously.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- If $b = 0$ set $r = \mathsf{EDExt}(x, \mathsf{seed})$ else if $b = 1$ set $r \leftarrow \{0,1\}^m$.
- If $\mathsf{BAD} \vee \neg\mathsf{GUESS}$ occurs then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Hybrid 3 and Hybrid 2 are indistinguishable by the security of the CALBO. In particular, we can think of both Hybrids 2,3 as choosing two tags $\mathsf{tag}^*_{i^*,j^*}, \mathsf{tag}'$ uniformly at random in the beginning of the game, but only tag' is used later in the game to determine if the event GUESS occurred. The only difference between Hybrid 2 and 3 is then that we choose $\mathsf{pk}_{i^*,j^*} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}^*_{i^*,j^*})$ in Hybrid 2 and $\mathsf{pk}_{i^*,j^*} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ in Hybrid 3. These are indistinguishable by CALBO security and therefore, in hybrid 3, we have:

$$\Pr[b = b'] \geq \frac{1}{2} + \epsilon(\lambda)/p - \mathrm{negl}(\lambda).$$

Hybrid 4. In this hybrid, we always choose r as uniform. In particular, the hybrid is defined as follows:

- Sample a random $j^* \leftarrow [j_{max}], \mathsf{tag}' \leftarrow \{0,1\}^{i^*}$. Sample a random bit $b \leftarrow \{0,1\}$. Sample the $\mathsf{seed} = (\{s_{i,j}, \mathsf{pk}_{i,j}\}, k)$ by selecting $\mathsf{pk}_{i^*,j^*} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{tag}')$ and choosing all other components in the same manner as previously.
- Run $(x, \mathsf{aux}) \leftarrow \mathcal{S}^{\mathsf{EDExt}(\cdot,\mathsf{seed})}(1^\lambda)$.
- Choose $r \leftarrow \{0,1\}^m$.
- If $\mathsf{BAD} \vee \neg\mathsf{GUESS}$ occurs then set $b' \leftarrow \{0,1\}$, else set $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r)$.

Fix any choice of $\mathsf{tag}', j^*, k, \{\mathsf{pk}_{i,j}\}, \{s_{i,j} \ : \ (i,j) \neq (i^*,j^*)\}$ in the experiment for the remainder of the analysis.

Let $S_{i^*,j^*}, X, \mathsf{AUX}, \mathsf{SEED}$ be random variables for the corresponding values in the experiment. Let $Y_{i^*,j^*} = F^{i^*}_{\mathsf{pk}_{i^*,j^*},\mathsf{tag}'}(S_{i^*,j^*})$. Let $Z = f_k(X)$. The seed Z determines all the values $\{\mathsf{TAG}_{i,j}\} = G(Z)$. Let $Y_{i,j} = F^{i}_{\mathsf{pk}_{i,j},\mathsf{TAG}_{i,j}}(s_{i,j})$ for $(i,j) \neq (i^*,j^*)$. Let $Y = \bigoplus_{i,j} Y_{i,j}$. Let E be a random variable which is 0 if $\mathsf{BAD} \vee \neg\mathsf{GUESS}$ and 1 otherwise. Define $\hat{X}$ and $\hat{Y}$ to be equal to X, Y respectively if $E = 1$ and to be uniformly random and independent over $\{0,1\}^n$ otherwise. Let $R = 2\mathsf{Ext}(\hat{X}, \hat{Y})$.

Claim 5.11.1. *There is a randomized function $\mathcal{A}$ such that* $\Pr[\mathcal{A}(S_{i^*,j^*}, \mathsf{AUX}, E, R) = 1] = \Pr[b = b' \text{ in hybrid } 3]$ *and* $\Pr[\mathcal{A}((S_{i^*,j^*}, \mathsf{AUX}, E, U_m) = 1] = \Pr[b = b' \text{ in hybrid } 4]$.

Proof. We define $A(s_{i^*,j^*}, \mathsf{aux}, e, r)$ to choose a random $b \leftarrow \{0,1\}$. Let $\mathsf{seed} = (\{s_{i,j}, \mathsf{pk}_{i,j}\}, k)$ where s_{i^*,j^*} is given as an input and the remaining values are already fixed. If $e = 0$, select $b' \leftarrow \{0,1\}$. If $e = 1$ let $b' = \mathcal{D}(1^\lambda, \mathsf{seed}, \mathsf{aux}, r')$ where $r' = r$ if $b = 0$ and $r' \leftarrow \{0,1\}^m$ otherwise. Output 1 iff $b = b'$.

It's easy to see the claim follows. In particular, the only time $\mathcal{A}(S, \mathsf{AUX}, E, R)$ makes use of $R = 2\mathsf{Ext}(\hat{X}, \hat{Y})$ is when $E = 1$ in which case $R = \mathsf{EDExt}(X, \mathsf{SEED})$. □

Claim 5.11.2. *The statistical distance* $\mathsf{SD}(\ (S_{i^*,j^*}, \mathsf{AUX}, E, R), (S_{i^*,j^*}, \mathsf{AUX}, E, U_m)) \leq \delta$.

Proof. Let us define an additional random variable $L = F^{lossy}_{\mathsf{pk}_{i^*,j^*}}(S_{i^*,j^*})$ and recall that $Z = f_k(X)$. First, we claim that

$$\begin{aligned}
\mathsf{SD}((S_{i^*,j^*}, \mathsf{AUX}, E, R), (S_{i^*,j^*}, \mathsf{AUX}, E, U_m)) &\leq \mathsf{SD}(S_{i^*,j^*}, \mathsf{AUX}, L, Z, E, R), (S_{i^*,j^*}, \mathsf{AUX}, L, Z, E, U_m)) \\
&\leq \mathsf{SD}((Y_{i^*,j^*}, \mathsf{AUX}, L, Z, E, R), (Y_{i^*,j^*}, \mathsf{AUX}, L, Z, E, U_m)) \\
&\leq \mathsf{SD}((\hat{Y}, \mathsf{AUX}, L, Z, E, R), (\hat{Y}, \mathsf{AUX}, L, Z, E, U_m))
\end{aligned}$$

The first inequality follows since adding information can only increase the statistical distance. The second inequality follows since S_{i^*,j^*} is independent of (AUX, E, Z, R) conditioned on (Y_{i^*,j^*}, L) and therefore we can think of S_{i^*,j^*} as a randomized function of (Y_{i^*,j^*}, L). The last inequality follows since Y_{i^*,j^*} is independent of R conditioned on $(\hat{Y}, \mathsf{AUX}, L, Z, E)$ and therefore we can think of Y_{i^*,j^*} as a randomized function of $(\hat{Y}, \mathsf{AUX}, L, Z, E)$.

Next, we note that $\hat{X}, \hat{Y}$ are independent conditioned on (AUX, L, Z, E).

Furthermore we have:

$$\begin{aligned}
H_\infty(\hat{X}|L, Z, E, \mathsf{AUX}) &\geq H_\infty(X|L, Z, E, \mathsf{AUX}) \\
&\geq H_\infty(X|L, \mathsf{AUX}) - v - 1 \\
&\geq H_\infty(X|\mathsf{SEED}, \mathsf{AUX}) - v - 1 \\
&\geq \alpha - v - 1
\end{aligned}$$

The first inequality follows since $\hat{X}$ is harder to predict than X, and the third inequality follows since L is fully determined by SEED.

We also have:

$$\begin{aligned}
H_\infty(\hat{Y}|L, Z, E, \mathsf{AUX}) &\geq H_\infty(\hat{Y}|L, Z, E) \\
&\geq H_\infty(Y|L, Z, E) \\
&\geq H_\infty(Y_{i^*,j^*}|L, Z, E) \\
&\geq H_\infty(Y_{i^*,j^*}) - \ell - v - 1 \\
&\geq n - \log d - \ell - v - 1
\end{aligned}$$

The first inequality follows since $\hat{Y}$ is independent of AUX conditioned on L, Z, E and the third inequality follows since Y is completely determined by Z, Y_{i^*,j^*}.The last inequality follows since $Y_{i^*,j^*} = F^{i^*}_{\mathsf{pk}_{i^*,j^*},\mathsf{tag}'}(S_{i^*,j^*})$ where S_{i^*,j^*} is uniform over $\{0,1\}^n$ and $F^{i^*}_{\mathsf{pk}_{i^*,j^*},\mathsf{tag}'}$ is at most d-to-1 so $\max_y \Pr[Y = y] = \max_y \Pr[S_{i^*,j^*} \in F^{i^*,-1}_{\mathsf{pk}_{i^*,j^*},\mathsf{tag}'}(y)] \leq d/2^n$.

Therefore, by the security of the strong two source extractor, we have

$$\mathsf{SD}(\ (\hat{Y}, \mathsf{AUX}, L, Z, E, R = 2\mathsf{Ext}(\hat{X}, \hat{Y})), (\hat{Y}, \mathsf{AUX}, L, Z, E, U_m)) \leq \delta$$

which concludes the proof of the claim. □

Combining the two claims, we have:

$$\frac{1}{2} = \Pr[b = b' \text{ in hybrid 4}] \geq \Pr[b = b' \text{ in hybrid 3}] - \delta \geq \frac{1}{2} + \epsilon(\lambda)/p - \text{negl}(\lambda)$$

where $p = j_{max}2^{i^*} \leq j_{max}(4q) = \text{poly}(\lambda)$.

Therefore, we must have $\epsilon(\lambda) = \text{negl}(\lambda)$, which completes the proof. □

Corollary 5.12. *Assuming the security of any of (DDH, DLIN, LWE, DCR), there exists an ED-Extractor for α-entropy sources with auxiliary info, for any $\alpha = \lambda^{\Omega(1)}$ and with any polynomial input length n and output length m. Security holds against polynomial-time sources and distinguishers.*

Proof. We can assume $n >\geq \lambda$ (we can always apply the construction on shorter inputs by appending 0s to the input). We set $i_{max}, j_{max} = \lambda^{.1}$. We set $v = \min\{\alpha/2, \lambda^{.1}\}$. We rely on the CALBOs of Corollary 5.8, so that for any i we have an (n, i, ℓ, d) CALBOs with $\ell = i \cdot \lambda^{.1} \leq \lambda^{.2}$ and $d = \lambda^{O(1)}$. We then rely on the Raz 2-source extractor (Theorem 2.5) with $e_1 = \alpha - v - 1 \geq \alpha/2 - 1 = \lambda^{\Omega(1)}$, $e_2 = n - \ell - v - \log d - 1 \geq n - 2n^{.2} - O(\log n) \geq (1 - o(1))n$, $\delta = 2^{-\lambda^{\Omega(1)}}$ and $m = \lambda^{\Omega(1)}$. This satsifies the conditions of the Theorem and therefore gives use the claimed result, except that the output size is only $m = \lambda^{\Omega(1)}$. We can then apply a PRG to the output to get arbitrarily large output size. □

5.4 Negative Results for ED Extractors with Auxiliary Info

Our constructions of ED-Extractors in the auxiliary info setting have several disadvantages compared to our construction in the setting without auxiliary info. Firstly, in the auxiliary info setting we needed complex constructions based on "cryptomania" assumptions (LWE and DDHI), whereas in the setting without auxiliary info, we showed that any sufficiently secure PRF is a good ED-Extractor. Secondly, in the auxiliary info setting we only achieved security for polynomial-time distinguishers while in the setting without auxiliary info we got security even for computationally unbounded distinguishers. In this section, we give some evidence that the two setting are substantially different and that we indeed need to work harder and cannot hope for as much in the setting with auxiliary info.

5.4.1 Not All PRFs are ED-Extractors with Aux Info

Firstly, we show that *not* every PRF is a good ED-Extractor in the setting with auxiliary info. We give two variants of this result. The first is based on collision-resistant hash functions (CRHFs) and gives a PRF that is not an α-ED-Extractor for entropy $\alpha = n - \lambda^{\varepsilon}$. The second one is based on fully homomorphic encryption and gives a PRF that is not an α-ED Extractor even for entropy $\alpha = n - 1$. In both cases, the result holds even if the PRF/ED-Extractor only outputs 1 bit.

CRHF-based Construction. Let $F' : \{0,1\}^{\ell} \times \{0,1\}^{n'} \to \{0,1\}$ be a PRF with key-length $\ell = \ell(\lambda)$, input length $n' = n'(\lambda)$ and output length 1. Let $H : \{0,1\}^{d} \times \{0,1\}^{n} \to \{0,1\}n'$ be a collision-resistant hash function (CRHF) with seed length $d = d(\lambda)$, input length $n = n(\lambda)$ and output length $n' = n'(\lambda)$. We define a PRF $F : \{0,1\}^{\ell+d} \times \{0,1\}^{n} \to \{0,1\}$ as follows. Parse the key $k = (k', s)$ with $k' \in \{0,1\}^{\ell}, s \in \{0,1\}^{d}$. Define $F(k, x)$:

- If $x \leq d$ output $s[x]$, where we interpret x as an integer in the range $[2^n]$ and $s[x]$ denotes the x'th bit of s.
- Else output $F(k', H(s, x))$.

It is easy to see that F is a PRF if F' is a PRF and H is a CRHF. On the other hand it is not an $\alpha = (n - n')$-ED-Extractor. In particular, consider the source that queries the oracle on values $1, \ldots, d$ to learn the CRHF seed s. It then chooses a random $x \leftarrow \{0,1\}^n$ and outputs $x, \mathsf{aux} = H(s, x)$. It is clearly an α legal source. Yet we can define a distinguisher D that gets $k = (k', s)$, aux, r and outputs 1 iff $r = F(k', \mathsf{aux})$. Then D always outputs 1 if r is the outputs of the ED-Extractor on x but only outputs 1 with probability $1/2$ if r is truly random, giving it a non-negligible advantage of $1/2$. For parameters, we note that the existence of CRHFs implies the existence of a CRHF with arbitrary polynomial input size $n = n(\lambda)$ and output size λ^ε for any constant $\varepsilon > 0$. Therefore, we get a PRF with arbitrary polynomial input size $n = n(\lambda)$ and output size $m = 1$, which is not an α-ED-Extractor for $\alpha = n - \lambda^\varepsilon$.

Theorem 5.13. *Assuming the existence of collision-resistant hash functions, for every polynomial $n = n(\lambda)$ and every constant $\varepsilon > 0$ there exists a PRF with n-bit input and 1-bit output which is not a secure α-ED-Extractor with axuliary input for $\alpha = n - \lambda^\varepsilon$.*

FHE-based Construction. Let $F' : \{0,1\}^\ell \times \{0,1\}^{n'} \to \{0,1\}$ be a PRF with key-length $\ell = \ell(\lambda)$, input length $n' = n'(\lambda)$ and output length 1. Let $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an FHE scheme capable of evaluating the PRF F'. Furthermore assume that the ciphertexts are pseudorandom and that the Eval procedure is statistically circuit private. Assume that the key-generation algorithm and the encryption algorithm each use at most $d = d(\lambda)$ bits of randomness, and that the encryption of an ℓ-bit message produces an ℓ'-bit ciphertext. Define the the PRF $F : \{0,1\}^{\ell+2d} \times \{0,1\}^n \to \{0,1\}$ as follows. Parse the key $k = (k', s_1, s_2)$ with $k' \in \{0,1\}^\ell, s_1, s_2 \in \{0,1\}^d$. Define $F(k, x)$:

- Check if $x \leq \ell'$(where we interpret x as an integer in the range $[2^n]$). If so let $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda; s_1)$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, k; s_2)$. Output the x'th bit of ct denoted by $\mathsf{ct}[x]$.
- Else output $F(k, x)$.

It is easy to see that F is a secure PRF: by the security of the FHE with pseuduorandom ciphetexts, we can replace ct by a uniformly random value independent of k, and by the security of the PRF F' the above is then a good PRF. On the other hand it is not an $\alpha = (n - 1)$-ED-Extractor. In particular, consider the source that queries the oracle on values $1, \ldots, \ell'$ to learn the the ciphertext ct. It then chooses a random $x \leftarrow \{0,1\}^n$ and outputs $x, \mathsf{aux} = \mathsf{Eval}(F'(\cdot, x), \mathsf{ct})$ so that aux is an FHE encryption of $F'(k, x)$. Since Eval is circuit private aux does not reveal anything about x beyond $F(k, x)$ and therefore is an $\alpha = n - 1$ legal source. Yet we can define a distinguisher D that gets $k = (k', s_1, s_2)$, aux, r and outputs 1 iff $\mathsf{Dec}(\mathsf{sk}, \mathsf{aux}) = r$ where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda; s_1)$. Then D outputs 1 with probability 1) if r is the outputs of the ED-Extractor on x, but only outputs 1 with probability $1/2$ if r is truly random, giving it a non-negligible advantage of $1/2 - \mathrm{negl}(\lambda)$. Therefore, we get a PRF with arbitrary polynomial input size $n = n(\lambda)$ and output size $m = 1$, which is not an α-ED-Extractor for $\alpha = n - 1$.

Theorem 5.14. *Assuming the existence of Fully Homomorphic Encryption (FHE) with statistical circuit privacy and pseudorandom ciphertexts, for every polynomial $n = n(\lambda)$ there exists a PRF with n-bit input and 1-bit output which is not a secure α-ED-Extractor with axuliary input for $\alpha = n - 1$.*

5.4.2 Black-Box Separations

We now show give two black-box separation results, showing that certain types of ED-Extractors cannot be proven secure via a black-box reduction from virtually any "standard" computational assumption (e.g.,including general assumptions such as the existence of one-way functions or public-key encryption, as well as specific assumptions such as DDH, LWE, RSA, etc., even if we assume (sub-)exponential security). In particular, we show two results of this type. Firstly, we show that one cannot prove the security of any ED-Extractor in the auxiliary info setting against computationally unbounded distinguishers (and polynomial-time sources) under such assumptions. This is contrast to the setting without auxiliary info, where we were able to do so. Secondly, we show that one cannot prove security in the auxiliary input setting (even for polynomial-time sources and distinguisher) of any ED-Extractor that has a certain type of *seed-committing* property: if you query the extractor EDExt on some polynomial set of values $x_1, \ldots, x_q$ then the output uniquely fixes a single possible seed that could have produced it. This is true for many natural constructions, such as the Naor-Reingold PRF or most block-cipher and hash-function based constructions. (But is crucially not true for our constructions based on constrained PRFs.) We view this as partial evidence that more complex constructions are necessary in the setting with auxiliary info.

Note that these results do not show that ED-Extractors with such properties cannot be constructed; in fact the work of Coretti et al. [CDKT19] in the random-oracle model can be interpreted as showing that "good" hash functions are heuristically likely to be good ED-Extractors in the auxiliary info setting with security even against computationally unbounded distingusihers, *and* they are also likely to be seed-committing. However, our results show that we cannot prove security under standard assumptions.

Our results are of the same flavor as the work of Wichs [Wic13]. They define the class of (single-stage) cryptographic game assumptions, which are modeled via a game between a challenger and a stateful adversary. They require that any polynomial-time (or sub-exponential time) attacker has at most a negligible (or inverse sub-exponential) success probability in winning the game. This captures essentially all standard assumptions used in cryptography. However, the security definition of ED-Extractors is *not* a single-stage game since it involves two separate entities (the source and the distinguisher) who cannot share state.

We use the "simulatable attacker" paradigm (also called a meta-reduction) to prove our black box separations. This paradigm is formalized in [Wic13] and we give a high-level overview. To prove a separation, we design a class of inefficient attackers $\mathcal{A}_h$ indexed by some h that break the security property but otherwise satisfy any strucutral/legality conditions (e.g., being multi-stage, entropy conditions etc.). However we also design an efficient simulator $\mathcal{A}'$ that may not satisfy such conditions, such that one cannot distinguish between black-box access to $\mathcal{A}_h$ for a random h versus $\mathcal{A}'$. Therefore if some reduction can break an assumption given black-box access to every $\mathcal{A}_h$ it would also be able to do so given access to $\mathcal{A}'$. If for any polynomial ℓ we can further show such a simulatable attack which is $2^{-\ell(\lambda)}$ indistinguishable, then we also rule out black-box reductions under sub-expoential or even exponential assumptions.

Unbounded Distinguishers. We first give a black-box reduction for ED-Extractors in the auxiliary info setting with security against unbounded distinguishers. Since the distinguisher can be computationally unbounded, a black-box reduction cannot call it. Therefore it suffices to construct a class of simulatable inefficient sources $\mathcal{A}_h$ tha satisfy the legality conditions and ensure that for the output (x, aux) it holds that $\mathsf{seed}, \mathsf{aux}, \mathsf{EDExt}(x, \mathsf{seed})$, is statistically far from $\mathsf{seed}, \mathsf{aux}, u$

where u is uniform. Our a high level, the source $\mathcal{A}_h$ that we construct makes oracle queries and inefficiently learns the function $\mathsf{EDExt}(\cdot, \mathsf{seed})$ sufficiently well to predict $\mathsf{EDExt}(x, \mathsf{seed})$ for a random x with high accuracy without querying it. It chooses such random x and sets aux to be a "statistically binding commitment" of its prediction for $\mathsf{EDExt}(x, \mathsf{seed})$. This ensures that the distribution of $(\mathsf{seed}, \mathsf{aux}, \mathsf{EDExt}(x, \mathsf{seed}))$ is statistically far from $(\mathsf{seed}, \mathsf{aux}, uniform)$. The commitment is generated using an exponentially large random function h and can therefore be simultaneously statistically hiding and binding. Therefore this attack is simulatable by an efficient simulator that chooses a random x and outputs a commitment to a random value.

Theorem 5.15. *For any candidate ED-Extractor* $(\mathsf{SeedGen}, \mathsf{EDExt})$ *with* $n(\lambda)$*-bit input and* 1 *bit output and for any polynomial* $\ell = \ell(\lambda)$ *there exists a* $2^{-\ell(\lambda)}$*-simulatable attack against the* $\alpha = (n-1)$*-ED-Extractor security of the candidate in the setting with auxiliary info and unbounded distinguishers.*

In particular, if there is a black-box reduction showing this type of security for the candidate based on the security of some cryptographic game $\mathcal{G}$*, then* $\mathcal{G}$ *is not secure. If the reduction is based on the* $2^{\ell(\lambda)}$*-security of the game* $\mathcal{G}$ *then* $\mathcal{G}$ *is not* $2^{\ell(\lambda)}$ *secure.*

Proof. Assume that the length of $\mathsf{seed} \leftarrow \mathsf{SeedGen}(1^\lambda)$ is bounded by $|\mathsf{seed}| \leq p(\lambda)$ for some polynomial p. Let $q = q(\lambda) = 3p(\lambda) + \lambda$. Let H_λ be the set of all functions from $\{0,1\}^{\ell(\lambda)}$ to $\{0,1\}$. For any $h \in \mathsf{H}_\lambda$, consider the inefficient source $\mathcal{S}_{\lambda,h}$ that chooses $x_1, \ldots, x_q$ uniformly at random and queries its oracle on them, gets back $y_1, \ldots, y_q$, and finds the (lexiographically first) value seed' such that $\mathsf{EDExt}(x_i, \mathsf{seed}') = y_i$ for all $i \in [q]$. It chooses a random x, computes $z' = \mathsf{EDExt}(x, \mathsf{seed}')$ and sets $\mathsf{aux} = (r, h(r) \oplus z')$ where $r \leftarrow \{0,1\}^\ell$.

First we claim that for any $h \in \mathsf{H}_\lambda$, the above source $\mathcal{S}_{\lambda,h}$ breaks the security of the ED-Extractor with auxiliary info and an unbounded distinguisher. It's easy to see that $\mathcal{S}_{\lambda,h}$ is a legal source with entropy $n-1$ since x is uniformly random and aux can reveal at most 1-bit of information z' about x. Secondly, we claim that if $\mathcal{S}_{\lambda,h}$ has oracle access to $\mathsf{EDExt}(\cdot, \mathsf{seed})$, then with overwhelming probability the value seed' that it finds must agree with seed on at least $3/4$ of all inputs. Otherwise there exists some seed' that agrees with seed on $< 3/4$ inputs yet agrees with it on $x_1, \ldots, x_q$ which occurs with probability at most $2^p(3/4)^q = \mathrm{negl}(\lambda)$. This also implies that if we let $z' = \mathsf{EDExt}(x, \mathsf{seed}'), z = \mathsf{EDExt}(x, \mathsf{seed})$ in the experiment, then $z' = z'$ with probability $3/4 - \mathrm{negl}(\lambda)$. But this shows that the distribution $(\mathsf{seed}, \mathsf{aux}, u = \mathsf{EDExt}(\mathsf{seed}, x))$ is statistically far from $(\mathsf{seed}, \mathsf{aux}, u \leftarrow \{0,1\})$ since in the first case, if we let $\mathsf{aux} = (r, v)$ then $h(r) \oplus v = u$ with probability at least $3/4 - \mathrm{negl}(\lambda)$ while in the second case this happens with probability at most $1/2$.

Secondly, we claim that for a random $h \leftarrow \mathsf{H}_\lambda$, the above source $\mathcal{S}_{\lambda,h}$ can be simulated by an efficient $\mathcal{S}'_\lambda$ that runs in time $\mathrm{poly}(\lambda)$. We define $\mathcal{S}'_\lambda$ which chooses $x_1, \ldots, x_q$ uniformly at random and queries its oracle on them, gets back $y_1, \ldots, y_q$, and outputs a uniformly random $(r, v) \leftarrow \{0,1\}^\ell \times \{0,1\}$.

The only way that $\mathcal{S}_{\lambda,h}$ for a random h can be distinguished from $\mathcal{S}'_\lambda$ using black-box access is if two different executions of $\mathcal{S}$ use the same randomness r. Given Q queries to $\mathcal{S}$, this happens with probability at most $\mathrm{poly}(Q)2^\ell$. □

Seed-Committing Extractors. We show that one cannot prove security in the auxiliary input setting (even for polynomial-time sources and distinguisher) of any ED-Extractor that has a certain type of *seed-committing* property.

Definition 5.16. *An ED-Extractor is* seed-committing *if there exist some polynomial $q = q(\lambda)$ and some inputs $x_1, \ldots, x_q \in \{0,1\}^{n(\lambda)}$ such that for any $\mathsf{seed}, \mathsf{seed}'$ for which $\mathsf{EDExt}(x_i, \mathsf{seed}) = \mathsf{EDExt}(x_i, \mathsf{seed}')$ for all $i \in [q]$ it must hold that for all x^* we have $\mathsf{EDExt}(x^*, \mathsf{seed}) = \mathsf{EDExt}(x^*, \mathsf{seed}')$.*

For example, if we use the Naor-Reingold PRF [NR97] as an ED-Extractor then it is seed-committing. Moreover, we believe that ED-Extractor constructions using standard hash-functions and block-cipher will be seed-committing.

Theorem 5.17. *For any candidate seed-committing ED-Extractor $(\mathsf{SeedGen}, \mathsf{EDExt})$ with $n(\lambda)$-bit input and $m(\lambda)$ bit output and for any polynomial $\ell = \ell(\lambda)$ there exists a $2^{-\ell(\lambda)}$-simulatable attack against the $\alpha = (n-1)$-ED-Extractor security of the candidate in the setting with auxiliary info.*

In particular, if there is a black-box reduction showing this type of security for the candidate based on the security of some cryptographic game $\mathcal{G}$, then $\mathcal{G}$ is not secure. If the reduction is based on the $2^{\ell(\lambda)}$-security of the game $\mathcal{G}$ then $\mathcal{G}$ is not $2^{\ell(\lambda)}$ secure.

Proof. Let H_λ be the set of all pairs of functions $h_1 : \{0,1\}^\ell \to \{0,1\}^{q\ell+1}, h_2 : \{0,1\}^{q\ell+1} \to \{0,1\}^\ell$. First we define $(\mathsf{Enc}_{h_1,h_2}, \mathsf{Dec}_{h_1,h_2})$ to be an information-theoretic authenticated encryption scheme whose key is h_1, h_2. In particular, $\mathsf{Enc}_{h_1,h_2}(m) = (r, h_1(r) \oplus m, h_2(r, h_1(r) \oplus m))$ where $r \leftarrow \{0,1\}^\ell$ is uniformly random and $\mathsf{Dec}_{h_1,h_2}(r, c, \sigma) = h_1(r) \oplus c$ if $h_2(r, c) = \sigma$ and $\perp$ otherwise.

For any $h = (h_1, h_2) \in \mathsf{H}_\lambda$, consider an inefficient source/distinguisher pair $\mathcal{A}_{\lambda,h} = (\mathcal{S}_{\lambda,h}, \mathcal{D}_{\lambda,h})$ defined as follow. The source $\mathcal{S}_{\mathrm{sec},h}$ chooses $x_1, \ldots, x_q$ as given by the seed-committing definition and queries its oracle on them, gets back $y_1, \ldots, y_q$, and finds the (lexiographically first) seed' such that $\mathsf{EDExt}(x_i, \mathsf{seed}') = y_i$ for all $i \in [q]$. It chooses a random x, computes z' to be the first bit of $\mathsf{EDExt}(x, \mathsf{seed}')$ and sets $\mathsf{aux} \leftarrow \mathsf{Enc}_h(y_1, \ldots, y_q, z')$. The distinguisher $\mathcal{D}_{\lambda,h}$ gets $(\mathsf{seed}, \mathsf{aux}, u)$, it computes z to be the first bit of u. It sets $\mathsf{Dec}_h(\mathsf{aux}) = (y_1, \ldots, y_q, z')$. If $\mathsf{EDExt}(\mathsf{seed}, x_i) = y_i$ for all $i \in [q]$ and $z' = z$ it outputs 0 else 1.

It is easy to see that, for any h, the adversary $\mathcal{A}_{\lambda,h}$ is an $\alpha = (n-1)$-legal adversary and breaks ED-Extractor security with advantage $1/4$: If the challenge bit is $b = 0$, the distinguisher always outputs 0 and if the challenge bit is $b = 1$ the distinguisher only outputs 1 with probability $> 1/2$.

Secondly, for a random $h = (h_1, h_2)$ the adversary $\mathcal{A}_{\lambda,h}$ can be efficiently simulated by a stateful adversary $\mathcal{A}' = (\mathcal{S}', \mathcal{D}')$ that acts as both the source and the distinguisher but allows them to share state. On input $y_1, \ldots, y_q$ to $\mathcal{S}'$, it chooses a random x, aux and remembers the tuple $(\mathsf{aux}, y_1, \ldots, y_q, x)$. On input $(\mathsf{seed}, \mathsf{aux}, u)$ to $\mathcal{D}'$ it checks if it stores a tuple of the form $(\mathsf{aux}, y_1, \ldots, y_q, x)$. If it does store such a tuple and $\mathsf{EDExt}(\mathsf{seed}, x_i) = y_i$ for all $i \in [q]$ and u is equal to the first bit of $\mathsf{EDExt}(x, \mathsf{seed})$ it outputs 0 else 1.

To show that one cannot distinguish between black-box access to $\mathcal{A}$ vs $\mathcal{A}'$ we define an intermediate $\mathcal{A}^*$ which is inefficient but also stateful. In particular, $\mathcal{A}^* = (\mathcal{S}^*, \mathcal{D}^*)$ acts just like $\mathcal{A}$, but instead of encrypting, the source $\mathcal{S}$ sets aux to be uniformly random and stores the tuple $(\mathsf{aux}, y_1, \ldots, y_q, z')$ and instead of decrypting $\mathcal{D}^*$ retrieves the tuple indexed by aux to uses the corresponding $(y_1, \ldots, y_q, z')$.

Firstly, we claim that $\mathcal{A}$ and $\mathcal{A}^*$ are indistinguishable by any (comp. unbounded) distinguisher that makes Q queries with probability better than $\mathrm{poly}(Q) \cdot 2^{-\ell}$. This essentially follows by the authenticated-encryption security of the encryption scheme.

Secondly, we claim that $\mathcal{A}^*$ and $\mathcal{A}'$ are perfectly indistinguishable. The only difference between them is that $\mathcal{A}^*$ compares u against the first bit of $\mathsf{EDExt}(\mathsf{seed}', x)$ while $\mathcal{A}'$ compares it against $\mathsf{EDExt}(\mathsf{seed}, x)$. But since $\mathsf{seed}, \mathsf{seed}'$ agree on $x_1, \ldots, x_q$, the seed-committing property ensures that $\mathsf{EDExt}(\mathsf{seed}', x) = \mathsf{EDExt}(\mathsf{seed}, x)$. □

Acknowledgements. YD was partially supported by gifts from VMware Labs, Facebook and Google, and NSF grants 1314568, 1619158, 1815546. VV was supported in part by NSF Grants CNS-1350619 and CNS-1414119, an NSF-BSF grant CNS-1718161, the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226 and W911NF-15-C-0236, an IBM-MIT grant and a Microsoft Trustworthy and Robust AI grant. DW was supported by NSF grants CNS-1314722, CNS-1413964, CNS-1750795 and the Alfred P. Sloan Research Fellowship.

References

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 57–74. Springer, Heidelberg, August 2013.

[AMN+18] Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Constrained PRFs for NC^1 in traditional groups. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 543–574. Springer, Heidelberg, August 2018.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519. Springer, Heidelberg, March 2014.

[BH05] Boaz Barak and Shai Halevi. A model and architecture for pseudo-random generation with applications to /dev/random. In Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels, editors, *ACM CCS 2005: 12th Conference on Computer and Communications Security*, pages 203–212. ACM Press, November 2005.

[BHK11] Mark Braverman, Avinatan Hassidim, and Yael Tauman Kalai. Leaky pseudo-entropy functions. In Bernard Chazelle, editor, *ICS 2011: 2nd Innovations in Computer Science*, pages 353–366. Tsinghua University Press, January 2011.

[BHK13] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. Instantiating random oracles via UCEs. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 398–415. Springer, Heidelberg, August 2013.

[BIW04] Boaz Barak, Russell Impagliazzo, and Avi Wigderson. Extracting randomness using few independent sources. In *45th Annual Symposium on Foundations of Computer Science*, pages 384–393. IEEE Computer Society Press, October 2004.

[Blu86] Manuel Blum. Independent unbiased coin flips from a correlated biased source-a finite stae markov chain. *Combinatorica*, 6(2):97–108, 1986.

[BST03] Boaz Barak, Ronen Shaltiel, and Eran Tromer. True random number generators secure in a changing environment. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 166–180. Springer, Heidelberg, September 2003.

[BV15] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 1–30. Springer, Heidelberg, March 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300. Springer, Heidelberg, December 2013.

[CDKT19] Sandro Coretti, Yevgeniy Dodis, Harish Karthikeyan, and Stefano Tessaro. Seedless fruit is the sweetest: Random number generation, revisited. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 205–234. Springer, Heidelberg, August 2019.

[CG85] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity (extended abstract). In *26th Annual Symposium on Foundations of Computer Science*, pages 429–442. IEEE Computer Society Press, October 1985.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, 1988.

[CGH+85] Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem of t-resilient functions (preliminary version). In *26th Annual Symposium on Foundations of Computer Science*, pages 396–407. IEEE Computer Society Press, October 1985.

[CPW19] Suvradip Chakraborty, Manoj M Prabhakaran, and Daniel Wichs. Witness maps and applications. 2019. Manuscript.

[CZ16] Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In Daniel Wichs and Yishay Mansour, editors, *48th Annual ACM Symposium on Theory of Computing*, pages 670–683. ACM Press, June 2016.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[DPR+13] Yevgeniy Dodis, David Pointcheval, Sylvain Ruhault, Damien Vergnaud, and Daniel Wichs. Security analysis of pseudo-random number generators with input:

/dev/random is not robust. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013: 20th Conference on Computer and Communications Security*, pages 647–658. ACM Press, November 2013.

[DRV12] Yevgeniy Dodis, Thomas Ristenpart, and Salil P. Vadhan. Randomness condensers for efficiently samplable, seed-dependent sources. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 618–635. Springer, Heidelberg, March 2012.

[DSSW14] Yevgeniy Dodis, Adi Shamir, Noah Stephens-Davidowitz, and Daniel Wichs. How to eat your entropy and have it too - optimal recovery strategies for compromised RNGs. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 37–54. Springer, Heidelberg, August 2014.

[FGK+10] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 279–295. Springer, Heidelberg, May 2010.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th Annual Symposium on Foundations of Computer Science*, pages 464–479. IEEE Computer Society Press, October 1984.

[GKK19] Ankit Garg, Yael Tauman Kalai, and Dakshita Khurana. Computational extractors with negligible error in the CRS model. *IACR Cryptology ePrint Archive*, 2019:1116, 2019.

[GT16] Peter Gazi and Stefano Tessaro. Provably robust sponge-based PRNGs and KDFs. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 87–116. Springer, Heidelberg, May 2016.

[Hut16] Daniel Hutchinson. A robust and sponge-like PRNG with improved efficiency. Cryptology ePrint Archive, Report 2016/886, 2016. http://eprint.iacr.org/2016/886.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 12–24. ACM, 1989.

[IZ89] Russell Impagliazzo and David Zuckerman. How to recycle random bits. In *30th Annual Symposium on Foundations of Computer Science*, pages 248–253. IEEE Computer Society Press, October / November 1989.

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013: 20th Conference on Computer and Communications Security*, pages 669–684. ACM Press, November 2013.

[KRVZ11] Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *J. Comput. Syst. Sci.*, 77(1):191–220, 2011.

[LLS89] David Lichtenstein, Nathan Linial, and Michael E. Saks. Some extremal problems arising form discrete control processes. *Combinatorica*, 9(3):269–287, 1989.

[NR97] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudorandom functions. In *38th Annual Symposium on Foundations of Computer Science*, pages 458–467. IEEE Computer Society Press, October 1997.

[NZ93] Noam Nisan and David Zuckerman. More deterministic simulation in logspace. In *25th Annual ACM Symposium on Theory of Computing*, pages 235–244. ACM Press, May 1993.

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

[PS18] Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 675–701. Springer, Heidelberg, March 2018.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 187–196. ACM Press, May 2008.

[Raz05] Ran Raz. Extractors with weak random seeds. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 11–20. ACM Press, May 2005.

[ST17] Pratik Soni and Stefano Tessaro. Public-seed pseudorandom permutations. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 412–441. Springer, Heidelberg, April / May 2017.

[TV00] Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *41st Annual Symposium on Foundations of Computer Science*, pages 32–42. IEEE Computer Society Press, November 2000.

[von51] John von Neumann. Various techniques used in connection with random digits. In A.S. Householder, G.E. Forsythe, and H.H. Germond, editors, *Monte Carlo Method*, pages 36–38. National Bureau of Standards Applied Mathematics Series, 12, Washington, D.C.: U.S. Government Printing Office, 1951.

[Wic13] Daniel Wichs. Barriers in cryptography with weak, correlated and leaky sources. In Robert D. Kleinberg, editor, *ITCS 2013: 4th Innovations in Theoretical Computer Science*, pages 111–126. Association for Computing Machinery, January 2013.