Weak Compression and (In)security of Rational Proofs of Storage

Ben Fisch[*] and Shashwat Silas[†]

Stanford University

Abstract

We point out an implicit unproven assumption underlying the security of *rational proofs of storage* that is related to a concept we call *weak randomized compression.* This is a class of interactive proofs designed in a security model with a rational prover who is encouraged to store data (possibly in a particular format), as otherwise it either fails verification or does not save any storage costs by deviating (in some cases it may even increase costs by "wasting" the space). Weak randomized compression is a scheme that takes a random seed r and a compressible string s and outputs a compression of the concatenation $r \circ s$. Strong compression would compress s by itself (and store the random seed separately). In the context of a storage protocol, it is plausible that the adversary knows a weak compression that uses its incompressible storage advice as a seed to help compress other useful data it is storing, and yet it does not know a strong compression that would perform just as well. It therefore may be incentivized to deviate from the protocol in order to save space. This would be particularly problematic for *proofs of replication*, designed to encourage provers to store data in a redundant format, which weak compression would likely destroy. We thus motivate the question of whether weak compression can always be used to efficiently construct strong compression, and find (negatively) that a black-box reduction would imply a universal compression scheme in the random oracle model for all compressible efficiently sampleable sources. Implausibility of universal compression aside, we conclude that constructing this black-box reduction for a class of sources is at least as hard as directly constructing a universal compression scheme for that class.

1 Introduction

The principal question we pose in this work is the following:

> Given an explicit and efficient (polynomial time) compression algorithm that on an input string s from an entropy source $\mathcal{S}$ and a uniformly n-bit random string r outputs a (efficiently decodable) compression of their concatenation $r \circ s$, does this yield an explicit and efficient compression of s itself?

This question has a subtle connection to the security of several cryptographic *proof of storage* protocols, which we will elaborate upon. First, let us address the answer to this

[*]email:bfisch@stanford.edu

[†]email:silas@stanford.edu

question. By intuition, because the string r comes from the uniform distribution U_n and is incompressible, the compression algorithm must only be taking advantage of the compressibility of $\mathcal{S}$. The converse of this question is of course true: given a compression algorithm for the source $\mathcal{S}$, we can compress the direct produce source $U_n \times \mathcal{S}$ by outputting r concatenated with the compression of s. However, can we say that any compression algorithm for the direct product source yields an explicit compression algorithm for $\mathcal{S}$ itself? This requires a certain flavor of a *direct sum theorem* [9,15], extensively studied in the context of communication protocols for proving that lower bounds on the communication complexity of several parallel copies of the same protocol additatively compose. Our question here, on the other hand, is algorithmic and concerns only efficient reductions. An explicit positive answer could be a black-box algorithm that can make black-box queries to a compression scheme for $U_n \times \mathcal{S}$ in order to compress $\mathcal{S}$ directly.

Our answer is in a negative direction: the existence of a black-box reduction of the compression of any source $\mathcal{S}$ to the compression of $U_n \times \mathcal{S}$ implies a universal compression algorithm for all sources that only has access to a random oracle. A universal compression algorithm for a class of sources is a uniform probabilistic polynomial time algorithm that is given samples from any source in the class and outputs a near-optimal compression of that source. In other words, in the random oracle model, there is no black-box algorithm that does this other than one that just universally compresses all sources as optimally as possible (i.e. performs as well as any other efficient compression algorithm for each source). Near-optimal universal compression algorithms are only known for very special classes of sources (e.g. Lempel-Ziv for stationary ergodic processes [24] , also log-space sampleable distributions [21]). Random oracles do increase the power of the compression algorithm significantly by giving it a succinct handle to a large pool of shared entropy. In the random oracle, Alice can send Bob a succinct string (also known as a salt) describing a fresh random function. Nonetheless, it would be quite surprising if random oracles could be used for universal (decodable) compression.[1] Either showing that random oracles give universal compression, or showing that universal compression in the random oracle model implies universal compression in the plain model are both interesting open questions. Regardless, even if universal compression is possible with random oracles, our results still show that coming up with a black-box reduction from strong to weak compression for a given class of sources is at least as hard as constructing a universal compression scheme for that class.

1.1 Strong and weak compression

A bit more formally, an $(\mathcal{S}, k)$-compression scheme for a source $\mathcal{S}$ over $\{0,1\}^m$ is a pair of polynomial time algorithms (E, D) such that $E : \{0,1\}^m \to \{0,1\}^k$ and $D(E(s)) = s$ for all $s \in \mathrm{Sup}(\mathcal{S})$. We may also say that the compression scheme in this case compresses $\mathcal{S}$ to k bits.[2] A source $\mathcal{S}$ is *k-compressible* if there exists an $(\mathcal{S}, k)$-compression scheme. Clearly this is only possible if $\mathrm{Sup}(\mathcal{S}) \leq 2^k$, i.e. the min-entropy of $\mathcal{S}$ is at most k-bits. A *randomized compression* (or seeded compression) takes additionally an n-bit seed $r \overset{\text{R}}{\leftarrow} \{0,1\}^n$ and outputs $E(r, s)$. If D is given r as well to decode then the scheme is said to have *shared*

[1] Considering that hash functions like SHA256 are believed to behave like random oracles, such a reduction might lead to a practical universal compression algorithm using SHA256.

[2] Note that for simplicity we are working with a bit more restrictive definition of compression than [21], as more generally compressing to k bits means that the expected size of the output is a k bit string.

randomness[3], or equivalently E outputs the compressed string t along with the seed r. We can generalize randomized compression to an algorithm $E : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n \times \{0,1\}^m$ that operates on the source $U_n \times \mathcal{S}$. The compression is *lossless* if $D(E(r,s) = (r,s)$ for all r, s, and it is *δ-lossy* if $Pr[D(E(r,s)) = r, s] > 1 - \delta$ over the probability of $r \overset{\text{R}}{\leftarrow} U_n$ and $s \overset{\text{R}}{\leftarrow} \mathcal{S}$.

We will distinguish between two types of this generalized randomized compression. In the special case that E simply outputs the seed r concatenated with the compression of s, whose information is then independent of r, we call this *strong randomized compression.* Otherwise, we will call it *weak randomized compression.* We refer to a strong/weak compression of $\mathcal{S}$ with an n-bit seed to $k+n$ bits as $(\mathcal{S}, n, k)$-strong/weak compression.

Our choice of terminology is due to the connection to *strong condensers.* A *randomness extractor* is an algorithm that extracts k bits of entropy from a source and outputs a distribution statistically close to U_k, whereas a condenser is an algorithm that extracts entropy from a source and is close to injective (an explicit efficient decoder is not required). In essence, condensers bridge the gap between extractors and compression. Like compression algorithms, extractors and condensers may use a seed from the uniform distribution to help with exraction; in general it may infuse the randomness from the seed with its output, however a *strong extractor* (resp. condenser) is one that outputs the extracted entropy independently from the seed. It is known that extractors imply strong extractors via an efficient reduction that incurs only logarithmic increase in the seed length and a small loss in the output distribution's distance from uniform [20].

1.2 Theorem statements

We can rephrase our question now more formally. Given two sources $\mathcal{S}_1, \mathcal{S}_2$, a black-box reduction from $(\mathcal{S}_1, k_1)$-compression to $(\mathcal{S}_2, k_2)$-compression is an algorithm that compresses $\mathcal{S}_1$ to k_1 given black-box access to any $(\mathcal{S}_2, k_2)$-compression scheme. A black-box reduction may apply to two types of compression on the same source, e.g. deterministic to randomized or strong to weak. A black box reduction from strong to weak compression would be a reduction that takes any $(\mathcal{S}, n, k)$-weak compression scheme (for an arbitrary source $\mathcal{S}$) and builds via black-box queries a $(\mathcal{S}, k')$-strong compression scheme, and we say the loss of the reduction is the difference $k' - k$. Finally, a *universal compression* scheme (with e loss) for a class $\mathcal{C}$ of sources is a compression scheme (U_E, U_D) that compresses each source in the class to within e of the optimal (efficient) compression length, i.e. for all $\mathcal{S} \in \mathcal{C}$ such that $\mathcal{S}$ is k-compressible, (U_E, U_D) compresses $\mathcal{S}$ to $k + e$ bits.

> Is there an efficient black box reduction from strong compression of any source $\mathcal{S}$ to weak compression of $\mathcal{S}$?

Again, our answer is negative, unless there is a universal compression algorithm for all sources in the random oracle model.

Theorem 1. *Any efficient black-box reduction of strong randomized compression to weak randomized compression implies (constructively) universal compression in the random oracle model for the class of all polynomial time sampleable sources. The loss of the universal compression is the loss of the reduction.*

[3]There is also a version of randomized compression with independent randomness where both E and D are randomized algorithms but do not directly share the same random seed.

In building towards this theorem we first prove a slightly weaker statement, which says that any black-box reduction of (deterministic/seedless) compression to weak compression for a particular source (with small loss) can be used to construct a near optimal compression scheme for that source. It turns out that this restriction is seemingly necessary for the analysis of the proposition. We are then able to generalize the result to any reduction from strong randomized compression to weak compression by using the fact that any compression scheme can be derandomized at small loss [21]. We can also make a similar generalization to ϵ-lossy compression (for small ϵ).

Proposition 1. *For any polynomial time sampleable k-compressible source $\mathcal{S}$, integers n and $e < n - \omega(\log n)$, given an explicit black box reduction $\mathcal{B}$ from $(\mathcal{S}, k+e)$-compression to $(\mathcal{S}, n, k)$-weak randomized compression we can construct from $\mathcal{B}$ an $(\mathcal{S}, k+e)$-compression scheme (B_E, B_D) (i.e. without knowing an $(\mathcal{S}, k)$-compression scheme explicitly).*

In light of these results, the most we can realistically conjecture is a certain kind of *knowledge assumption*[4]: to construct a randomized compression scheme for a source $\mathcal{S}$ (i.e. that compresses $U_n \times \mathcal{S}$ to k bits) one must "know" a strong compression scheme that compresses $\mathcal{S}$ itself to k bits.

Assumption 1. *For all sampleable sources $\mathcal{S}$ and a $(\mathcal{S}, n, k)$-weak compression scheme (E, D), there exists a $(\mathcal{S}, k + o(n))$ strong compression scheme (E^*, D^*). Moreover, for any "adversary" $\mathcal{A}$ that on inputs $\mathcal{S}$ and n outputs (E, D), there exists an efficient "extractor" which observes $\mathcal{A}$'s internal state and outputs (E^*, D^*).*

1.3 A connection to proofs of storage

A variety of cryptographic *proofs of storage* have been proposed througout the literature. These are interactive protocols between and prover (e.g. server) and verifier (e.g. client) with different goals relating to statements about the prover's storage. Proofs-of-retrievability (PoR) [13] demonstrate that the prover can retrieve some specified data known to the verifier. Public-key PoR allows for the verification of these proofs to be outsourced to a verifier who only obtains a cryptographic commitment or authentication tag from the client referencing the data. Unless the data is incompressible, a PoR does not prove anything to the verifier about how much space the prover is using in order to retrieve the file, particularly in the public-key setting where the prover may collude with the client. In proofs-of-space (PoS) [8] a time-constrained prover demonstrates that it is storing some incompressible string of $\Omega(N)$ bits, and therefore using at least $\Omega(N)$ bits of space. Proofs-of-secure-erasure (PoSE) [17] are similar to PoS and are used by a storage-bound prover to demonstrate that the prover has erased all of its storage. This is in essense an extremely *tight* PoS that no adversary can pass if it uses strictly less than N total bits of storage, although there is also a weaker form of PoSE that proves erasure of all but at most ϵN bits [14]. They were suggested as a way for embedded devices with small memory to prove to a server that it has erased private data or all prior code before an update (to wash out malware).

Rational proofs of storage Storage enforcing commitments [12] were a precursor to PoR in which the prover must dedicate a minimum amount of storage in order to pass the protocol for a committed file F (preprocessed by a client with a secret key). The protocol

[4]These are common in proofs of knowledge, e.g. knowledge of exponent assumptions [2].

does not guarantee that the prover is actually storing F, however a prover who does not would "waste" the space as it cannot use this space to store any other "meaningful" data. Hourglass schemes [22] were proposed as a generic method to prove that a server is storing data encoded in a specified format, in particular for the use cases of encryption-at-rest and file watermarking. Another special case of this type of protocol is a proof of data replication, which demonstrates the server is storing data encoded in a redundant format. These have been studied in a variety of models. The system Mirror [1] considered proofs of data replication as a special kind of private-verifier PoR in which the server replicates the client's initially preprocessed file. A more recent line of works introduced proofs-of-replication (PoRep) as a hybrid of a PoR and PoS that demonstrates in a publicly verifiable way that the prover is using space to store a unique replica of a file, or even several unique replicas of the same file [3, 10, 18, 19]. The threat model considered in all of these works is a *rational adversary* that acts to minimize its storage costs.

Composable security Formal security models have been proposed for the various types of proofs of storage. However, the matter of *composability* has received sparse treatement. Indeed in some cases, in particular PoR and PoS, composition does not seem difficult to deal with. Moran and Orlov [16] formally addressed amortization-resistance of their proofs-of-spacetime (PoST, a variant on PoS), showing that a prover engaging in k simultaneous copies of the PoST must be using k times as much space. Intuitively, PoS protocols are proved secure in the random oracle (RO) model by showing that adversary who passes the protocol with less space would be able to compress a random function, hence security under composition is achieved by forcing the prover to use different oracles in each instance (e.g. seeding each with a unique identifier). In the case of PoR, security implies the existence of an explicit public extraction algorithm that (given a key from the client) can extract the data itself from repeated protocol interactions with the prover. Therefore, as long as the extractors for each of several protocols operating in parallel on distinct data files are still able to publicly extract the respective files, then security is preserved. A formal security analysis showing that soundness is indeed preserved under composition is warranted, but would not be surprising in a suitable model given the existence of UC-secure proofs of knowledge [5, 11] where the knowledge extractor is weaker.

In this work we focus on a very different and overlooked composition issue, which we suggest is critical to the security definition of a class of proof-of-storage protocols including proofs of secure erasure, storage enforcing commitments, and proofs of replication. A common thread between the security underlying these proofs of storage is the implicit assumption that storing an incompressible random string "wastes" space and therefore cannot serve any other useful purpose than passing the proof, or alternatively that the incompressibility of the data encoding implies the server might as well store it in a particular format. Of course storing a random string could be useful for other protocols in ways that have nothing to do with storage, for example a common reference string (CRS) or password, but we focus specifically on how the incompressible string might actually be used as a seed to compress overall storage.

2 "Weak Compression Attacks" on Proofs of Storage

Before providing the main formal results of this paper, we first provide an overview of the consequences of weak compression for several types of proof of storage.

Proofs of secure erasure A PoSE that guarantees erasure of only $(1-\epsilon)N$ bits may somehow be used to compress the orginally stored data of size $\Omega(N)$ down to less than ϵN bits, which can fit in the remaining storage. This wouldn't prove data erasure at all. For security to make sense the verifier must know a lower bound on the compressibility of the data (e.g. the entropy of its source) and tune the protocol security parameters so that ϵN is below this bound. The entropy of the data might be quite small relative to its best known compression, leading to an inefficient scheme. The verifier might instead tune ϵ based instead on some assumption of the best known compression scheme. Our only point here is that considering best known compression of the source is insufficient: it does not rule out a weak randomized compression that entangles the PoSE randomness together with the original data. The verifier would need to have some approximation of the best known weak randomized compression scheme as well.

Storage enforcing commitments The possibility of weak compression poses a more fundamental issue for storage enforcing commitments [12]. Security is achieved by arguing that an adversary cannot both pass the proof and deviate from the protocol without "wasting" space, or at least it does not save more than an ϵ fraction of its storage by doing so (with ϵ arbitrarily small). In particular, the prover may either use the space to store some data of interest, or it stores a random "useless" string. However, this security intuition is violated if the prover can use the random string to compress other auxiliary data on its disk.

Hourglass schemes Hourglass schemes force a prover to dedicate storage resources to a particular encoding of a file. In the case of data-encryption, the prover is forced to dedicate resources to retrieving an encryption of the file. If the prover additionally chooses to store a plaintext copy of the file then it must double its resources. This is considered irrational behavior in the threat model considered. Weak compression does not seem to pose any isses for this particular use-case as it would not decrypt the data, and therefore would preserve the intended format. However, this may not be the case for other types of encodings. A prime counterexample is data replication.

Proofs of replication Proofs-of-replication require a security property that no adversary can save storage space by deviating from storing data of interest in a redundant format. More precisely, this format is called an N-replication of a data file D if it is a string s that can be partitioned into N independent substrings, each of which can be independently passed to a universal extraction algorithm that outputs the data D.

Naturally, the adversary could sabotage the replication by using say the first λ bits of s as a key to encrypt the rest, and store the transformed string s' that includes the λ bit key and ciphertext. Since the adversary can efficiently decode s from s' it will still pass the protocol with the same success probability (i.e. it *efficiently* decodes s' and retrieves s, and then follows whatever protocol behavior it would have initially on s). Indeed, such "scrambling" attacks are impossible to prevent as there is always a trivially fast way to encode/decode one's state in a way that destroys the N-replication format.

The best security one could hope for is that there is no sabotage attack that saves the adversary storage costs. In fact, as s itself is incompressible (due to the fact that it embeds data inside a proof of space), no sabotage attack could compress s. However, this rational security argument breaks down in the presence of auxiliary information as there may plausibly exist some weak compression scheme that jumbles s together with auxiliary data. This could of course destroy the N-replication format. To formally prove that N-replication is rational (or at least ϵ-rational, in that the prover cannot save more that $\epsilon|s|$ storage via a sabotage attack) one must at a *minimum* show/assume that any weak compression of s together with auxiliary data can be converted to a strong compression that just compresses the auxiliary data and stores s separately with negligible overall storage compression loss.

3 Preliminaries

3.1 Compression schemes

Definition 1 (Compression scheme). *An $(\mathcal{S}, k)$-compression scheme for a source $\mathcal{S}$ over $\{0,1\}^m$ is a pair of polynomial time algorithms (E, D) such that $E : \{0,1\}^m \rightarrow \{0,1\}^k$ and $D(E(s)) = s$ for all $s \in Sup(\mathcal{S})$.*

Definition 2 (k-Compressible). *A source $\mathcal{S}$ over $\{0,1\}^m$ is k-compressible if there is an efficient compression scheme (E, D) for $\mathcal{S}$ such that $E : \{0,1\}^m \rightarrow \{0,1\}^k$ and $D(E(s)) = s$ for all $s \in Sup(\mathcal{S})$.*

Definition 3 (Randomized compression/weak compression). *An $(\mathcal{S}, n, k)$-randomized compression or weak compression for a source $\mathcal{S}$ over $\{0,1\}^m$ is a compression scheme which uses an n bit seed in order to compress. Formally, $E : \{0,1\}^n \times Sup(\mathcal{S}) \rightarrow \{0,1\}^{n+k}$ (where $m > k$). If $Pr(D(E(r,s)) = r, s) = 1 - \delta$, we say that the scheme is δ-lossy. When $\delta = 0$ we say it is lossless.*

By analogy to strong extractors, we define the case where the output of E in the randomized compression scheme is simply the random string r concatenated with a compression of s.

Definition 4 (Strong compression). *An $(\mathcal{S}, k)$-strong randomized compression scheme is one in which $E(r,s) = \tau$ for $r \circ \tau \in \{0,1\}^k$. If $Pr(D(E(r,s)) = s) = 1 - \delta$, then we say the scheme is δ-lossy. In the special case when r is empty, we call this strong deterministic compression.*

Definition 5 (Universal compression). *A universal compression scheme with e loss for a class $\mathcal{C}$ of functions is a compression scheme (U_E, U_D) that compresses each source in the class to within e of the optimal (efficient) compression length. That is, for all $\mathcal{S} \in \mathcal{C}$ where $\mathcal{S}$ is k-compressible, (U_E, U_D) compresses it efficiently to $k + e$ bits.*

Definition 6 (Black-box reduction). *Let $\mathcal{S}$ be an arbitrary source. A black-box reduction from $(\mathcal{S}, k')$ compression to $(\mathcal{S}, k)$ is an oracle algorithm $\mathcal{B} = (B_E^o, B_D^o)$ that implements an $(\mathcal{S}, k')$-compression scheme making black box queries to any $(\mathcal{S}, k)$-compression scheme. Note that $k' \geq k$ and we call $e = k' - k$ the loss of the reduction.*

Finally, we introduce the notion of a compression game, which will be used in our analysis. Informally, a compression game consists of an encoder and decoder, both of which have access to an oracle. The encoder is given a string s to compress and it has a small (compared to the size of s) channel to communicate with the decoder. The decoder must use the oracle and the small piece of information from the encoder in order to correctly output s. More formally:

Definition 7 (Compression game)**.** *Let $\mathcal{S}$ be a set such that $|\mathcal{S}| > 2^n$. A compression game for A is a tuple $(A_1, A_2, \mathcal{O}, k)$. Here, A_1 and A_2 are randomized algorithms which have access to the oracle $\mathcal{O}$. The compression game proceeds as follows*

1. *(Offline phase) A_1 and A_2 make poly(n) queries to the oracle and save the responses.*
2. *A_1 receives a challenge $s \overset{R}{\leftarrow} \mathcal{S}$. It makes poly$(n)$ queries to the oracle and generates a k-bit message $t \in \{0,1\}^k$ to send to A_2.*
3. *A_2 obtains the input t and makes poly(n) queries to the oracle. It outputs s'.*

The probability of sucess of the compression game is defined as $Succ(A, \mathcal{S}, k) = \Pr_{s \in S}(s' = s)$.

Derandomization and lossless compression The following two facts are from [21]. The first states that randomized compression schemes can be derandomized for a small loss under reasonable assumptions. The complexity assumption that there exists $E = DTIME(2^{O(n)})$ of circuit complexity $2^{\Omega(n)}$ implies that for any $t(n) = O(\text{poly}(n))$ there exists a pseudorandom generator $G : \{0,1\}^{\ell(n)} \to \{0,1\}^{t(n)}$ with $\ell(n) = O(\log n)$ such that no circuit of size $t(n)$ can distinguish the output of G from uniform with probability greater than $1/t(n)$.[5] The existence of this pseudorandom generator is then used to prove Fact 1.

Fact 1. *Suppose there is a function in $E = DTIME(2^{O(n)})$ of circuit complexity $2^{\Omega(n)}$. For every efficient compression scheme (E, D) with shared randomness implies (in a black box manner) a determinisitic compression scheme (E', D') such that if (E, D) compresses $\mathcal{S} \in \{0,1\}^n$ to length m, then (E', D') compress $\mathcal{S}$ to length $m + O(\log(n))$.*

Remark: random oracles In the random oracle (RO) model, derandomization is possible without further complexity assumptions because the RO can be queried on any $O(\log n)$ size seed to obtain a uniform random string. Thus, Fact 1 holds in the RO model without further assumptions.

The second fact shows that in the case of shared randomness, a lossy compression scheme can be made lossless for a small decrease in compression.

Fact 2. *Suppose (E,D) (which share randomness) can efficiently compress $\mathcal{S} \in \{0,1\}^n$ to length m with decoding error ε. Also suppose that for all $x \in Sup(\mathcal{S})$, $|E(s)| \geq m_0$. Then $\mathcal{S}$ is efficiently compressible to lenth $m + \varepsilon(n - m_0) + 1$ with shared randomness and no error.*

We state the following fact from [7].

Fact 3. *For any randomized encoding scheme (E, D) where $E : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^m$ and $D : \{0,1\}^\lambda \times \{0,1\}^m \to \{0,1\}^n$ such that $Pr[D(\rho, E(\rho, x)) = x] \geq \delta$ over $x \overset{R}{\leftarrow} \{0,1\}^n$ and $\rho \overset{R}{\leftarrow} \{0,1\}^\lambda$ then $m \geq n - \log(1/\delta)$.*

3.2 Indistinguishability games

We define a generic indistinguishability game for two function families. The game is between a challenger and an adversary who makes a sequence of queries (at most a polynomial number in the size of the domain/range).

[5]The same complexity assumption implies **BPP = P**.

Definition 8. *Let $\mathcal{F}_0$ and $\mathcal{F}_1$ be two function families on finite sets X to Y, where $|X| = m$ and $|Y| = n$. For $\ell \in \mathbb{Z}, b \in \{0,1\}$, the game $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}_1, \mathcal{F}_2, \ell, b)$ is defined as follows:*

1. *The challenger samples $f \overset{\mathrm{R}}{\leftarrow} \mathcal{F}_b$*
2. *$\mathcal{A}$ submits a polynomial number of queries $q_1, ..., q_\ell$ to the challenger where $q_i \in \{0,1\}^m$, and the challenger responds to each ith query with $f(q_i)$. The queries are made adaptively (the adversary receives each response before making the next).*
3. *$\mathcal{A}$ outputs a bit $\hat{b} \in \{0,1\}$ (i.e. a guess as to which function family f was sampled from).*

Let W_0 denote the event $\mathcal{A}$ outputs 1 at the end of the game when $b = 0$ and W_1 the event that it outputs 1 when $b = 1$. We define the aversary's advantage as $\mathsf{Adv}^{\mathcal{A}}(\mathcal{F}_1, \mathcal{F}_2, \ell) = |Pr[W_0] - Pr[W_1]|$.

We cite a well known fact about the indistinguishability of random permutations on a set X of size 2^m from a random function on X for *poly*(m) bounded adversaries (see e.g. [4]).

Fact 4. *Let X be a finite set of size n, let $\mathsf{Perm}(X)$ denote the family of all permutations on $X \to X$, and let $\mathsf{Func}(X)$ denote the family of all functions on $X \to X$. For any $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{A}}(\mathsf{Perm}(X), \mathsf{Func}(X), \ell) < \ell^2/2n$.*

Indistinguishability of two distributions over a larger support can be more generally characterized by the statistical distance of two distributions.

Definition 9. *If X_1, X_2 are two random variables distributed over a common finite support Ω then their statistical distance is defined to be $\Delta(X_1, X_2) = \frac{1}{2}\sum_{s\in\Omega} |Pr[X_1 = s] - Pr[X_2 = s]|$.*

Fact 5. *Let E denote an event over a finite probability space Ω. Let X_1 and X_2 be two random variable distributions over Ω and $X_1|E$ and $X_2|E$ the random variables conditioned on event E. Let $\bar{E}$ denote the complement event. Then:*

$$\Delta(X_1, X_2) \leq P(E)\Delta(X_1|E, X_2|E) + (1 - P(E))\Delta(X_1|\bar{E}, X_2|\bar{E})$$

Proof. By definition $\Delta(X_1, X_2) = \frac{1}{2}\sum_{s\in\Omega} |Pr[X_1 = s] - Pr[X_2 = s]|$. Expand $Pr[X_1 = s] = P(E)Pr[X_1 = s|E] + (1-P(E))Pr[X_1 = s|\bar{E}]$ and likewise $Pr[X_2 = s] = P(E)Pr[X_2 = s|E] + (1 - P(E))Pr[X_2 = s|\bar{E}]$. Using the triangle inequality we upper bound $\Delta(X_1, X_2)$ by:

$$\frac{1}{2}\sum_{s\in\Omega} |P(E)(Pr[X_1 = s|E] - Pr[X_2 = s|E])| + |P(\bar{E})(Pr[X_1 = s|\bar{E}] - Pr[X_2 = s|\bar{E}])|$$

$$= P(E)\Delta(X_1|E, X_2|E) + P(\bar{E})\Delta(X_1|\bar{E}, X_2|\bar{E})$$

□

Generalization to multiple oracles The indistinguishability game in Definition 8 can easily be adapted so that each $\mathcal{F}_b$ is a family of function tuples, where each tuple was sampled according to some constraint (e.g. a pair (f_1, f_2) such that $f_2 = f_1^{-1}$ on a set X).

4 (Im)possibility of strong to weak compression reduction

The bulk of our analysis is in the proof of Proposition 1. The high level structure is as follows. Given an $(\mathcal{S}, n, k)$-weak compression scheme we construct another $(\mathcal{S}, n, k)$-weak compression scheme which "encrypts" its output. Any black box reduction must be able to use this "encrypted" weak compression scheme to give an $(\mathcal{S}, k + e)$-strong compression scheme. That is to say, the black box reduction must be able to play the compression game for $\mathcal{S}$ with oracle access to the "encrypted" weak compression, and compress $\mathcal{S}$ into $k + e$ bits with at most a neglible reduction in the probability of success. We then show successive modifications to this compression game which improve (or at least do not reduce) the probability of success. This sequence of transformations ends with a compression game in which the oracle is simply a random function. In fact, because we are only able to prove that a reduction implies universal compression in the random oracle model, we might as well allow the existential compression scheme we construct to use a random permutation oracle Π and random function oracle H directly.[6] (This allows us to skip several hybrid transformation steps in the proof). The transformation shows that if there is a black box reduction from strong $(\mathcal{S}, n, k)$ compression to weak $(\mathcal{S}, k + e)$ compression, then we can constructively modify this black box reduction algorithm into a strong $(\mathcal{S}, k+e)$ compression scheme that just queries a random oracle (oblivious to any other explicit compression scheme). Before going into the details of the proof, let us restate Proposition 1 and how Theorem 1 follows. The main difference between Proposition 1 and Theorem 1 is that Proposition 1 considers black box reductions from (deterministic) compression to weak compression, whereas Theorem 1 generalizes the result to consider more broadly black box reductions from strong (randomized) compression to weak compression.

Proposition 1 *For any polynomial time sampleable k-compressible source $\mathcal{S}$, integers n and $e < n - \omega(\log n)$, given an explicit black box reduction $\mathcal{B}$ from $(\mathcal{S}, k + e)$-compression to $(\mathcal{S}, n, k)$-weak randomized compression we can construct from $\mathcal{B}$ an $(\mathcal{S}, k + e + \mathsf{negl}(n))$-compression scheme (B_E, B_D) (i.e. without knowing an $(\mathcal{S}, k)$-compression scheme explicitly).*

Theorem 1 *Any efficient black-box reduction of strong (randomized) compression to weak compression implies (constructively) universal compression in the random oracle model for the class of all polynomial time sampleable sources. The loss of the universal compression is the loss of the reduction.*

Proof of Theorem 1. By definition, an efficient black box reduction from strong to weak compression is an oracle algorithm $B = (B_E^o, B_D^o)$ which will implement an $(\mathcal{S}, n, k + e(n))$-strong compression scheme by making black box queries to a $(\mathcal{S}, n, k)$-weak compression scheme for any k, n, function $e(\cdot)$, and source $\mathcal{S}$.

As we are in the random oracle model, by Fact 1 any strong compression scheme with an n-bit seed can be derandomized in a black-box way (by querying the oracle) with only $O(\log n)$ compression loss. Therefore, given any $(\mathcal{S}, n, k)$-weak compression, our reduction B' first uses B to construct an $(\mathcal{S}, n, k + e(n))$-strong randomized compression scheme, and then derandomizes this to a $(\mathcal{S}, k + e(n) + O(\log n))$-compression scheme. Therefore, B' is now a reduction from $(\mathcal{S}, k + e(n) + O(\log n))$-compression to $(\mathcal{S}, n, k)$-weak compression. Now we can invoke Proposition 1 to say that if $\mathcal{S}$ is in fact k-compressible, we can construct

[6]Random oracles can be used to implement ideal ciphers [6].

from B' a $(\mathcal{S}, k + e(n) + O(\log n))$-compression scheme B'' (all without knowing explicitly a $(\mathcal{S}, k)$ compression scheme). If for all $k \in \mathbb{N}$ the original reduction B works for any k-compressible polynomial time sampleable source $\mathcal{S}$, then B'' is a universal compression scheme for all polynomial time sources (i.e. it performs almost as well as the optimal efficient compression scheme for each k-compressible source with up to $e(n) + O(\log n)$ loss).

$\square$

Remark A similar argument can be made to generalize Theorem 1 to apply to any black-box reduction of lossy strong compression to lossy weak compression (when the ϵ loss of the compression is small). Given such a reduction we could take any weak compression scheme and artificially add some ϵ error to make it lossy, then use the reduction to construct some ϵ'-error lossy strong compression, and finally using Fact 2 convert this to strong compression scheme with only $\epsilon' n$ loss in compression (i.e. blowup in compression size). Of course as $\epsilon \to 1$ this statement is vacuous (indeed in this case the encoding scheme behaves more like a randomness extractor/condenser, and as mentioned black-box reductions of strong to weak extractors are known).

4.1 Proof of Proposition 1

By hypothesis we suppose $\mathcal{S}$ is a k-compressible source and $\mathcal{B}$ is a black box algorithm that can implement $(\mathcal{S}, k + e)$-compression with oracle queries to any $(\mathcal{S}, k, n)$-weak randomized compression scheme where $e < n - \omega(\log n)$. From here on let us write $k' = k + e$.

Let us recall briefly the high level structure of our argument. By hypothesis, there exists a $(\mathcal{S}, k)$-compression scheme (E, D). We will "weaken" this to a weak randomized compression scheme that randomizes its output with a random permutation, and then argue implausibility of the existence of an algorithm to recover a strong compression from black-box calls to this weak compression. Specifically, we show that given such an explicit black-box algorithm we could directly construct a strong compression of $\mathcal{S}$ itself (in the random oracle model) without knowing (E, D).

Now let us proceed formally. We will weaken (E, D) in two steps. First, we trivially turn (E, D) into a $(\mathcal{S}, n, k)$-strong compression by appending an n-bit seed. Next we construct the weak randomized compression scheme (E', D') as follows. (E', D') will call a random permutation oracle Π for permutations on $\{0,1\}^{n+k}$, its inverse oracle Π^{-1}, and two random function oracles $H_1 : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^{n+k}$ and $H_2 : \{0,1\}^{n+k} \to \{0,1\}^n \times \{0,1\}^m$.

- $E'(r, s)$: On input (r, s) check if $D(E(r, s)) = (r, s)$. If so output $\Pi(E(r, s))$. Otherwise output $H_1(r, s)$.
- $D'(t)$ first computes $(r', s') = D(\Pi^{-1}(t))$. If $E(r', s') = t$, then output (r', s'). Otherwise output $H_2(t)$.

It is easy to see that E', D' compress to k-bits just as well as E, D. Let $IP \subseteq \{0,1\}^n \times \{0,1\}^m$ denote the subset of "invertible points" under (E, D), i.e. the set of points (r, s) such that $D(E(r, s)) = (r, s)$. E' maps IP injectively into $\{0,1\}^{n+k}$, and D' it its inverse on these points. This portion contains $\{0,1\}^n \times S$. The restriction of E' to IP is in fact distributed identically to a random injective function. This is because E' is itself injective on this restriction of its domain, and Π permutes its image to a random subset of $\{0,1\}^{n+k}$. On all other points outside of this domain E' maps to a random point. Likewise, D' implements a random function on all points outside $E'(IP)$ i.e. maps these points to random points in $\{0,1\}^n \times \{0,1\}^m$.

Hybrid compression games Now we introduce a series of hybrid compression games where in each we either replace the oracles given to $\mathcal{B}$ or we restrict the behavior of $\mathcal{B}'$ in a way that does not decrease its probability of success in the compression game. When replacing the oracles (rather than restricting behavior) we argue that the probability $\mathcal{B}$ succeeds in the new game has at most negligible loss due to an indistinguishability argument (i.e. that when $\mathcal{B}$ queries its oracles according to its current restrictions it cannot distinguish between the old/new oracles with non-negligible probability). To make these arguments formal, we generally use $\mathcal{B}$ to construct a distinguisher that leads to some contradiction. *Because the distinguisher must simulate the compression game for $\mathcal{B}$, this is where we need the hypothesis that the source $\mathcal{S}$ is polynomial time sampleable.* Since $\mathcal{S}$ and k' are fixed in each game we simply abbreviate the notation of the compression game with a pair of oracles $(\mathcal{O}_1, \mathcal{O}_2)$ to a tuple $(A_1, A_2, \mathcal{O}_1, \mathcal{O}_2)$. Furthermore from here on we define $S = \mathrm{Sup}(\mathcal{S})$.

Each hybrid game may introduce a negligible loss in the success probability of the compression game, thus the final $\mathcal{B}'$ succeeds in k'-compression of $\mathcal{S}$ with probability at least $1 - \mathsf{negl}(n)$. By Fact 2 this can be converted into a $k' + \mathsf{negl}(n)$-compression that succeeds with probability 1.

Our games are as follows:

1. **Game 1** $= (B_E, B_D, E', D')$ this is the original compression game where B_E and B_D are the components of the black-box reduction $\mathcal{B}$ and E', D' are our constructed oracles. By hypothesis, $Succ(B_E, B_D, E', D') = 1$.

2. **Game 2** $= (B_E, B_D, F, F^{-1})$ this game replaces the oracles with two randomly sampled functions F, F^{-1} subject only to the constraint that F^{-1} is the inverse of F on IP.

3. **Game 3** $= (B'_E, B'_D, F, F^{-1})$ this game restricts the queries of B'_E and B'_D so that both:

   (a) Do not repeat queries to either oracle.

   (b) Do not query an inverse oracle on the output of a query already made, i.e. if a query $F(q) = r$ is made then the query $F^{-1}(r)$ is never made, and $F^{-1}(r') = q'$ is made then $F(q')$ is never made.

   Additionally, both B'_E and B'_D start with a clean state containing no information about the oracle and make all the same (polynomial number) of queries to the oracle in the *offline* phase of the game).

4. **Game 4** $= (B'_E, B'_D, F_1, F_2)$ this game replaces (F, F^{-1}) with independently sampled random functions (F_1, F_2) subject only to the constraint that F_1 is injective on IP and F_2 is injective on $F_1(IP)$ (but they need not be inverses).

5. **Game 5** $= (B'_E, B'_D, H_1, H_2)$ this game finally replaces the partially injective random functions F_1, F_2 with random functions H_1, H_2 subject to no constraints.

We use the notation Game $i \approx$ Game $i+1$ to denote indistinguishability and Game $i \preceq$ Game $i+1$ to denote a non-decreasing success probability.

4.1.1 Game 1 $\approx$ Game 2

Note that as explained above, Game 1 $\approx$ Game 2 follows immediately from how (E', D') were defined. E' is distributed identically to an injective random function on IP, and D' is its inverse on the image $E'(IP)$. On all other points they are identical to random functions.

4.1.2 Game 2 $\preceq$ Game 3

Any time $\mathcal{B}$ would have made a repeat or an inverse query in Game 2, $\mathcal{B}'$ simply ignores making such a query. Instead, in the case for repeats it simply looks up the previous query and for inverses it can predict the the result from a previous query. Finally, the restriction that B'_E and B'_D make all the same offline queries is without loss of generality as these can be preprocessed and hardcoded into both B'_E and B'_D before they enter the "online" phase.

4.1.3 Game 3 $\approx$ Game 4

Step 1 Consider any finite sets X, Y where $|X| = m$ and $|Y| = n$ for $m > n$, and a fixed subset $S \subseteq X$ where $|S| \leq n$. Define $\mathcal{F}$ to be the family of function pairs $F_1 : X \to Y$ and $F_2 : Y \to X$ such that F_1 is injective on S and F_2 is injective on $F_1(S)$. Define $\mathcal{G}$ to be the family of function pairs $F_1 : X \to Y$ and $F_2 : Y \to X$ such that F_1 is injective on S and $F_2 = F_1^{-1}$ on $F_1(S)$.

Clearly, any adversary that can distinguish between Game 3 and Game 4 with $\ell = poly(n)$ queries can be used to win the game $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, b)$, as the only difference between these two games is the sampling of its oracles from these two different distributions. Our first objective is to show that any adversary that has non-negligible probability of distinguishing these games must make what we will call a *witness query pair*, which is a query q to F_1 and r to F_2 such that $r = F_1(q)$ or $q = F_2(r)$. Intuitively, this is the only way to distinguish the distributions as they only differ in the fact that one pair are inverses on the set S.

Claim 1. *If $\mathsf{Adv}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell) > \epsilon(n, m)$ for non-negligble ϵ and $\ell < poly(n, m)$, then either $\mathcal{A}$ makes a witness query in $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, 0)$ or in $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, 1)$ with non-negligible probability.*

Proof. We define several random variables and events over the randomness of $\mathcal{A}$, F_1, and F_2 sampled in both experiments. Let $\mathbf{q} = (\hat{q}_1, ..., \hat{q}_{\ell_1})$ denote a vector of random variables representing the values of $\mathcal{A}$'s sequence of queries to F_1 and let $\mathbf{r} = (\hat{r}_1, ..., \hat{r}_{\ell_2})$ denote $\mathcal{A}$'s sequence of queries to F_2.

Let W_0 denote the event that in the experiment with $b = 0$ the queries $\mathbf{q}$ and $\mathbf{r}$ contain a witness pair, i.e. $F(\hat{q}_i) = \hat{r}_j$ or $F(\hat{r}_j) = \hat{q}_i$ for some i, j. Let W_1 denote the same event in the experiment with $b = 1$. Finally let $W = W_0 \vee W_1$.

Let $View_b$ denote $\mathcal{A}$'s view of all queries and oracle responses in the experiments with $b = 0$ and $b = 1$ respectively.

We show that $View_0|\bar{W} \approx View_1|\bar{W}$, i.e. the two views are identically distributed conditioned on the event that neither sets of queries in each game contain a witness pair. We can show this by finite induction on the number ℓ of queries. For $\ell = 0$, no queries have been made so the distributions of the views are vacuously identical. Now assume that for any $\ell = i$ sets of queries the two views are identical. Consider $\ell = i + 1$. The partial views in both experiments after the first i queries are identical by assumption, therefore the $i + 1$st query will be sampled identically in both cases. This is either a query q to F_1 or r to F_2. Conditioned on $\bar{W}$, q was never the output of some query to F_2 (in either experiment). Therefore conditioned on the current view it is subject to the same constraints in either experiments. In both experiments, $F_1(q)$ is distributed uniformly subject only to the constraint that if $q \in S$ then $F_1(q)$ cannot collide with the outputs of any previous queries to F_1 in the current view. A symmetric argument can be applied to a query r to F_2. Since ℓ is finite we conclude that the two views $View_0|\bar{W} \approx View_1|\bar{W}$ are identical.

Finally, by Fact 5:

$$\Delta(View_0, View_1) \leq P(W)\Delta(View_0|W, View_1|W)$$

because $\Delta(View_0|\bar{W}, View_1|\bar{W}) = 0$. If $\Delta(View_0, View_1) > \epsilon$ for non-negligible ϵ, then by the above inequality $P(W) > \epsilon$. This means that either $\mathcal{A}$ makes a witness query in experiment 0 or in experiment 1 with non-neglible probability. □

Claim 2. *Given $\mathcal{B} = (\mathcal{B}_E, \mathcal{B}_D)$ that has non-negligible difference in its probability of success between Game 3 and Game 4, we can design (Enc, Dec) which can with non-negligible probability $\delta > 1/poly(n, k)$ compress the function tables of one of the random functions F_1 or F_2 by at least $n + k - k' - O(\log n)$ bits in either Game 3 or Game 4.*

Proof. Since $\mathcal{B} = (\mathcal{B}_E, \mathcal{B}_D)$ has a non-negligible difference in its probability of success between Game 3 and Game 4, it achieves a non-negligible $\mathsf{Adv}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell)$ where $\mathcal{F}$ and $\mathcal{G}$ are defined as in Claim 1 on set X, Y of sizes $n + m$ and $n + k$ respectively. Hence, we know that with a non-negligible probability B makes a "witness" query pair to either F, F^{-1} in Game 3 or F_1, F_2 in Game 4. *Enc* will start by playing both Game 3 and Game 4 in parallel as the challenger with $\mathcal{B}$, sampling F_1, F_2, F, F^{-1} accordingly in each, and also choosing $\mathcal{B}$'s randomness. If in neither of these games $\mathcal{B}$ makes a witness query then *Enc* aborts. This happens with probability at most $(1 - \delta)$ for some $\delta > 1/poly(n, m, k)$. If a witness query is made in any of these games, *Enc* writes down which game and saves the randomness ρ it used to run $\mathcal{B}$ in the game. We show the case where this happens in Game 3, and design (Enc, Dec). The case for Game 4 is analogous as our proof will not at all use the fact that F^{-1} is the inverse of F. Importantly, the randomness ρ sampled by *Enc* will be shared by *Dec* (i.e. it is not considered as part of the compression output). (Recall Fact 3 says that random functions are incompressible even for such randomized compression schemes with shared randomness).

Enc Continue from above, save the randomness ρ that was used to run $\mathcal{B}$ and replay the game with $\mathcal{B}$ on the same ρ, F, F^{-1}. *Enc* starts with the entire function tables T_F and $T_{F^{-1}}$. Note that T_F is a list of outputs in the set $Y = \{0,1\}^{n+k}$ sorted by inputs in $X = \{0,1\}^n \times \{0,1\}^m$ and $T_{F^{-1}}$ is a list of outputs in X sorted by inputs in Y. As *Enc* is running the game with $\mathcal{B}$, it observes the queries made by $\mathcal{B}$. It records the k' bits (i.e. the value t) communicated by $\mathcal{B}_E$ to $\mathcal{B}_D$, and also increments a counter that counts the queries made by $\mathcal{B}_D$ after receiving t. By assumption at some point $\mathcal{B}$ makes a witness query pair q to F and $F(q)$ to F^{-1}. By the restriction on the queries, we also know one of the queries in the witness pair was made by $\mathcal{B}_E$ and the other was made by $\mathcal{B}_D$. (Otherwise this would violate the restriction that neither can query $\mathcal{F}^{-1}$ on the output of a query to $\mathcal{F}$ and vice-versa). We also know that the witness query pair happens during the *online phase* because all offline queries are shared by both $\mathcal{B}_E$ and $\mathcal{B}_D$. More specifically, one of two cases occur:

(a) In the online phase, first $\mathcal{B}_E$ queried $F(q) = r$ and second $\mathcal{B}_D$ queried $F^{-1}(r)$. *Enc* now records the index i of this query in the order queries were made by $\mathcal{B}_D$ (which requires on $O(\log(n))$ bits to store) and writes this at the index q of T_F. Specifically, *Enc* replaces the entry r with (t, i).

(b) In the online phase, first $\mathcal{B}_E$ queried $F^{-1}(r)$ and second $\mathcal{B}_D$ queried $F(q)$. In this case *Enc* similarly records the index i at which this occurs in the queries made by $\mathcal{B}_D$ and stores at index r in $T_{F^{-1}}$ the value (t, i).

Finally, *Enc* outputs either its modified T_F^* or $T_{F^{-1}}^*$.

To see that *Enc* has indeed compressed the table of either T_F or $T_{F^{-1}}$ simply note that $|r| = n + k < |q|$ whereas $|(t, i)| = k' + O(\log(n)) < n + k$. It can output one more bit to indicate to *Dec* which table it has compressed.

Dec Now we describe how Dec decodes T_F^* (resp. $T_{F^{-1}}^*$) received from *Enc*. It obtains the randomness ρ as its first input and the encoding T_F^* (resp. $T_{F^{-1}}^*$) from *Enc*. Then it replays the game with $\mathcal{B}$ on randomness ρ, using the table T_F^* (resp. $T_{F^{-1}}^*$) to answer queries until it reaches the index at which $\mathcal{B}_E$ queries $F(q)$ on a missing entry at index q in T_F^* (resp. missing $F^{-1}(r)$ at index r in T_F^*). It finds in this location the value (t, i), skips the rest of the online phase for B_E, and sends t immediately to $\mathcal{B}_D$. Now it counts the queries until the ith query $\mathcal{B}_D$ makes is a value r to F^{-1}. It adds (q, r) back to the table and outputs the completed T_F (thereby recovering (q, r). (It would operate on $T_{F^{-1}}$ in the analogous way). □

Given the hypothesis (in Proposition 1) that $k' - k < n - \omega(\log n)$, Claim 2 implies that the difference in B's success probability between Game 3 and Game 4 must be negligible. Otherwise, we could use B to compress a random function by more than $n + k - k' - O(\log n) > \omega(\log n)$ bits with success $\delta > 1/poly(n)$. This a contradiction to Fact 3 which says that compression cannot exceed $\log(1/\delta) = O(\log n)$ bits.

4.1.4 Game 4 $\approx$ Game 5

We will argue indistinguishability of Game 4 and Game 5 by using the fact that if $\mathcal{B}$ has a non-negligible difference in success between Game 4 and Game 5 then this would be a contradiction to Fact 4. Such a $\mathcal{B}$ is a distinguisher that achieves a non-negligible advantage in distinguishing random functions from random partially injective functions. To make the argument cleaner, we break the argument up into two partial hybrid steps, the first is Game 4.5 where we replace the partially injective F_1 with a random function H_1 and the second is Game 5 where we replace F_2 with H_2 as well.

Game 4 $\approx$ 4.5 (replace F_1 with H_1) Let $\mathcal{F}$ be the family of all functions from $X = 0, 1^n \times \{0,1\}^m$ to $Y = \{0,1\}^{n+k}$ and $\mathcal{G}$ be the subset of $\mathcal{F}$ that are also injective on $IP \subseteq X$ (recall IP is the set of points fixed by $D \circ E$). Then $\mathcal{B}$ that distinguishes Game 4 and Game 4.5 can be used to construct $\mathcal{A}$ for $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, b)$ with $\ell = \text{poly}(n)$ such that $\mathsf{Adv}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell)$ is non-negligible in n. ($\mathcal{A}$ just plays the compression game with $\mathcal{B}$ such that if it receives $g \overset{\text{R}}{\leftarrow} \mathcal{G}$ it perfectly simulates Game 4 and if it receives $f \overset{\text{R}}{\leftarrow} \mathcal{F}$ it perfectly simulates Game 4.5).

We now argue that such a distinguisher $\mathcal{A}$ for $\mathcal{F}$ and $\mathcal{G}$ is impossible. Since $f \overset{\text{R}}{\leftarrow} \mathcal{F}$ and $g \overset{\text{R}}{\leftarrow} \mathcal{F}$ are identical outside IP, it suffices to argue indistinguishability of the restriction of f, g to IP. Thus consider $\texttt{Func}(IP, Y)$ to be all functions from IP into Y and $\texttt{Inj}(IP, Y)$ to be all injective functions from IP into Y. First we formally show that $\mathcal{A}$ can indeed be used to build an $\mathcal{A}'$ that achieves non-negligible advantage in the modified game $\mathsf{IND}^{\mathcal{A}'}(\mathtt{Func}(IP, Y), \mathtt{Inj}(IP, Y), \ell, b)$. We use the fact that IP is polynomial time testable (given a point in X we can check if $D \circ E$ fixes the point or not).[7]$\mathcal{A}'$ plays the role of the challenger simulating the game $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, b)$ for $\mathcal{A}$ (oblivious to the value of b). Whenever $\mathcal{A}$ queries a point in IP, $\mathcal{A}'$ queries its own challenger in $\mathsf{IND}^{\mathcal{A}'}(\mathtt{Func}(IP, Y), \mathtt{Inj}(IP, Y), \ell, b)$ and gives the response to $\mathcal{A}$. $\mathcal{A}'$ also records all queries that $\mathcal{A}$ makes on points in $X \setminus IP$ in table L. If $\mathcal{A}$ queries a point $q \in X \setminus IP$ for the first time then $\mathcal{A}'$ chooses a random $r \xleftarrow{R} Y$ and stores (q, r) in L. If an entry (q, r') already exists in L then $\mathcal{A}'$ returns r' to $\mathcal{A}$. Since $\ell < \text{poly}(|X|)$ the list never grows too large. It is easy to see that in case $\mathcal{A}'$ is playing the game with $b = 0$ and its challenger has sampled $f \xleftarrow{R} \mathtt{Func}(IP, Y)$ then it perfectly simulates $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, 0)$, and in the case that $b = 1$ and its challenger has sampled $g \xleftarrow{R} \mathtt{Inj}(IP, Y)$ then it perfectly simulates $\mathsf{IND}^{\mathcal{A}}(\mathcal{F}, \mathcal{G}, \ell, 1)$.

Now we assume we have an adversary $\mathcal{A}$ that distinguishes $\mathtt{Func}(IP, Y)$ and $\mathtt{Inj}(IP, Y)$ to construct $\mathcal{A}'$ that achieves $\mathsf{Adv}^{\mathcal{A}}(\mathtt{Func}(Y), \mathtt{Perm}(Y), \ell)$ non-negligible in n, which contradicts Fact 4. $\mathcal{A}'$ will use the polynomial time computable function E. When $h \xleftarrow{R} \mathtt{Func}(Y)$, the function $h \circ E : IP \to Y$ is identically distributed to $f \xleftarrow{R} \mathtt{Func}(IP, Y)$ (because E is injective and then h maps to a random point in Y). Likewise, when $\pi \xleftarrow{R} \mathtt{Perm}(Y)$, the function $\pi \circ E : IP \to Y$ is identically distributed to $g \xleftarrow{R} \mathtt{Inj}(IP, Y)$. Thus, $\mathcal{A}'$ simulating the challenger for $\mathcal{A}$ will respond to a query $q \in IP$ by first computing $y = E(q)$ and then querying y to its own challenger in $\mathsf{IND}^{\mathcal{A}}(\mathtt{Func}(Y), \mathtt{Perm}(Y), \ell, b)$. The simulation is perfect in either case $b = 0$ or $b = 1$. This proves that $\mathcal{A}'$ achieves non-negligible advantage, which is a contradiction.

Game 4.5 $\approx$ Game 5 (replace F_2 with H_2) The analysis is symmetric to the analysis above for Game 4 $\approx$ Game 4.5. There are two main differences. First, we replace IP with its image $E(IP) \subseteq Y$. Note that $E(IP)$ is polynomial time testable just like IP. Second, we will call D instead of E in the reduction that builds the distinguisher $\mathcal{A}'$ for function familes $\mathtt{Func}(X)$ and $\mathtt{Perm}(X)$ from a distinguisher $\mathcal{A}$ for the function families $\mathtt{Func}(E(IP), X)$ and $\mathtt{Inj}(E(IP), X)$.

5 Weak compression resistant PoS?

We have pointed out that at a *minimum* we need to assume some form of strong to weak compression reduction in order to achieve meaningful rational security in proof of storage protocols, particularly those like proof-of-replication that aim to incentivize a certain storage format. Theorem 1 suggests a black-box reduction from strong to weak compression is unlikely, but we can consider using a plausible *knowledge of strong compression* assumption (like Assumption 1) in order to prove that an adversary cannot gain any storage advantage by using its PoS advice string as a seed to weakly compress auxiliary data. Assumption 1 provides a minimum requirement: if the adversary is able to weakly compress auxiliary data using a *uniformly sampled random seed* then it could "extract" from itself a strong compression that compresses the axulliary data and outputs the seed independently. But a PoS advice string isn't a uniformly random seed persay—indeed proving a lower bound on the storage requirements of PoS advice is in general highly non-trivial. Moreover, we

[7]The adversary in the proof $\mathcal{A}'$ is allowed to use the polynomial time computable functions E, D, which by hypothesis exist. Note that $\mathcal{B}$ is still oblivious to E, D. We are only using E, D inside this proof of indistinguishability to argue the existence of a polynomial time algorithm $\mathcal{A}'$.

know from proof-of-replication that PoS *can* be used to embed *specific* data of interest. Is Assumption 1 sufficient?

> For what types of PoS protocols does Assumption 1 imply that a strong compression of auxiliary data z can always be extracted from a weak compression of z seeded by the PoS advice string?

If a PoS satisfies this property we will say it is *weak compression resistant.* We identify sufficient properties of a PoS for which this is true. It was recently proven that a PoS construction from *depth robust graphs* (DRG) has these properties [18]. First we briefly review the syntax of a generalized PoS protocol, where additionally a data commitment D can be specified.

A proof of space is an interactive protocol with two phases:

- Initialization is an interactive protocol between P and V that run on shared input (id, N) and P is additionally given data D and auxiliary input z. P outputs Φ and S, where S is its storage advice and Φ is a compact string given to the verifier.

- Execution is an interactive protocol between P and V where P runs on input S and V runs on input Φ. V sends challenges to P, obtains back a proof π, and outputs accept or reject.

The PoS protocol has perfect completeness if the verifier always outputs accept with an honest prover. It is (ϵ, N, T) sound if no P that implements a depth at most T circuit in the execution phase can pass verification with probability non-neglible in N and storage S where $|S| < \epsilon N$. We additionally say the PoS is *data commiting* if the output Φ is a binding cryptographic commitment (not necessarily hiding) to the data input D.

The DRG PoS was proven (in the random oracle model) to have the property that there is a bound T such that any depth T prover storing S can be used to compress the function table $T_{\mathcal{H}}$ of a random oracle $\mathcal{H}$ by at least ϵN bits, and this is where the storage lower bound on S comes from as $T_{\mathcal{H}}$ is incompressible (with some $\log(1/\delta)$ loss for the prover's failure probability δ). We claim that Assumption 1 is sufficient to say that any data commiting PoS protocol with this specific property is weak compression resistant.

Proposition 2. *Assumption 1 implies weak compression resistance for any data committing PoS with the additional property that its storage advice S of size $\Omega(N)$ can be used to compress the function table of a random oracle table T_H by $|S|$ bits. That is, we can extract from the PoS prover a strong compression of any auxiliary data $z \neq d$, where d is the committed data.*

Proof Sketch. Suppose there exists a weak compression $Enc(S, z) = \tau$ such that $Dec(\tau) = (S, z)$ and $|\tau| = |S| + k < |S| + |z|$. Then since S can be used to compress a random oracle table T_H by $|S| > \epsilon N - \log(1/\delta)$ bits, we can form $Enc'(T_H, z) = (T_H^*, Enc(S, z)) = (T_H^*, \tau)$ such that $Dec'(T_H^*, \tau)$ can first run $Dec(\tau) = (S, z)$ and then use S and T_H^* to recover T_H. Hence $Dec'(T_H^*, \tau) = (T_H, z)$ where $|T_H^*| + |\tau| \leq |T_H| + k + \log(1/\delta)$. This is a weak randomized compression with a uniform random seed T_H. Hence by Assumption 1 there exists a strong compression of z to $k + O(\log N)$ bits when δ is non-negligible in N. □

6 Conclusion

We have introduced the notion of weak compression and pointed out its relevance to defining the security of rational proofs of storage, particularly storage enforcing commitments like proof of replication that are intended to incentivze a particular format of file storage. We have shown the implausibility of a black-box reduction of strong compression to weak compression, which is of independent interest. Finally, we suggested a minimal knowledge assumption that could be used as the basis for analyzing the composable security of rational proofs of storage in presence of auxiliary data, in lieu of an unconditional proof.

References

[1] Frederik Armknecht, Ludovic Barman, Jens-Matthias Bohli, and Ghassan O. Karame. Mirror: Enabling proofs of data replication. In *25th USENIX Security Symposium*, 2016.

[2] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In *CRYPTO*, 2004.

[3] Dan Boneh, Joseph Bonneau, Benedikt Bunz, and Ben Fisch. Verifiable delay functions. 2018. To appear in CRYPTO 2018.

[4] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography.* 2017. Version 0.4.

[5] Jan Camenisch, Stephan Krenn, and Victor Shoup. A framework for practical universally composable zero-knowledge protocols. In *Asiacrypt*, 2011.

[6] Jean-Seabastien Coron, Jacques Patarin, and Yannick Seurin. the random oracle model and the ideal cipher model are equivalent. *CRYPTO*, 2008.

[7] Anindya De, Luca Trevisan, and Madhur Tulsiani. Time space tradeoffs for attacks against one-way functions and prgs. In *CRYPTO*, 2010.

[8] Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. Proofs of space. In *CRYPTO*, 2015.

[9] Tomas Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM Journal of Computing*, 1995.

[10] Ben Fisch, Joseph Bonneau, Juan Benet, and Nicola Greco. Proofs of replication using depth robust graphs. In *Blockchain Protocol Analysis and Security Engineering 2018*, 2018. `https://cyber.stanford.edu/bpase2018`.

[11] J. A Garay, P. Mackenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. In *Journal of Cryptology*, 2006.

[12] Philippe Golle, Stanislaw Jarecki, and Ilya Mironov. Cryptographic primitives enforcing communication and storage complexity. In *Financial Cryptography*, 2002.

[13] Ari Juels and Burton S Kaliski Jr. Pors: Proofs of retrievability for large files. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 584–597. Acm, 2007.

[14] Nikolaos P. Karvelas and Aggelos Kiayias. Efficient proofs of secure erasure. In *SCN*, 2014.

[15] Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* 1997. Cambridge University Press.

[16] Tal Moran and Ilan Orlov. Rational Proofs of Space-Time. Cryptology ePrint Archive # 2016/035, 2016.

[17] Daniele Perito and Gene Tsudik. Secure code update for embedded devices via proofs of secure erasure. In *ESORICS*, 2010.

[18] Krzysztof Pietrzak. Proofs of Catalytic Space. Cryptology ePrint Archive # 2018/194, 2018.

[19] Protocol Labs. Proof of replication, 2017. https://filecoin.io/proof-of-replication.pdf.

[20] Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. *SIAM Journal of Computing*, page 11851209, 2006.

[21] Luca Trevisan, Salil Vadhan, and David Zuckerman. Compression of samplable sources. *computational complexity*, 14(3):186–227, 2005.

[22] Marten van Dijk, Ari Juels, Alina Oprea, Ronald L. Rivest, Emil Stefanov, and Nikos Triandopoulos. Hourglass schemes: how to prove that cloud files are encrypted. In *ACM CCS*, 2012.

[23] Gaven J. Watson, Reihaneh Safavi-Naini, Mohsen Alimomeni, Michael E. Locasto, and Shivaramakrishnan Narayan. Lost: location based storage. In *CCSW, year = 2012*.

[24] Jacob Ziv and Abraham Lempel. Compression of individual sequences by variable rate coding. *IEEE Transactions on Information Theory 24*, page 1530536, 1978.