Multi-collision Resistance:
A Paradigm for Keyless Hash Functions

Nir Bitansky* Yael Tauman Kalai† Omer Paneth‡

August 6, 2018

Abstract

We introduce a new notion of *multi-collision resistance* for *keyless* hash functions. This is a natural relaxation of collision resistance where it is hard to find multiple inputs with the same hash in the following sense. The number of colliding inputs that a polynomial-time non-uniform adversary can find is not much larger than its advice. We discuss potential candidates for this notion and study its applications.

Assuming the existence of such hash functions, we resolve the long-standing question of the round complexity of zero knowledge protocols — we construct a 3-message zero knowledge argument against arbitrary polynomial-size non-uniform adversaries. We also improve the round complexity in several other central applications, including a 3-message succinct argument of knowledge for **NP**, a 4-message zero-knowledge proof, and a 5-message public-coin zero-knowledge argument. Our techniques can also be applied in the keyed setting, where we match the round complexity of known protocols while relaxing the underlying assumption from collision-resistance to *keyed* multi-collision resistance.

The core technical contribution behind our results is a domain extension transformation from multi-collision-resistant hash functions for a fixed input length to ones with an arbitrary input length and a local opening property. The transformation is based on a combination of classical domain extension techniques, together with new information-theoretic tools. In particular, we define and construct a new variant of list-recoverable codes, which may be of independent interest.

*Tel Aviv University, email `nirbitan@tau.ac.il`. Member of the Check Point Institute of Information Security. Supported by the Alon Young Faculty Fellowship and by Len Blavatnik and the Blavatnik Family foundation. Part of this research was done while at MIT. Supported by NSF Grants CNS-1350619 and CNS-1414119 and DARPA and ARO under Contract No. W911NF-15-C-0236. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA and ARO.

†Microsoft Research, email `yael@microsoft.com`.

‡MIT, email `omerpa@mit.edu` Supported by NSF Grants CNS-1350619 and CNS-1414119, and 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.

Contents

1 Introduction

Collision-resistant hash functions are central to cryptography. They are used everywhere to compress communication and storage, from simple applications such as *hash-and-sign* to advanced applications like reliable delegation of data and computation [Mer89, Dam89, DPP93, BEG$^+$94]. They also have strong implications to foundational concepts in the theory of cryptography and complexity, including succinctness of proofs, non-black-box techniques, and hardness of total search problems [Kil94, Bar01, MP91, KNY17b].

In this work, we study a natural relaxation of collision resistance called *multi-collision resistance*. Roughly speaking, a shrinking hash function is multi-collision-resistant if finding *many* (rather than two) inputs that hash to the same output is intractable. We formalize this notion, explore its applications, and develop techniques for robust composition of multi-collision-resistant hash functions.

Keyless Hash Functions. Our main motivation for studying multi-collision resistance comes from the setting of keyless hash functions. It is well-known that (full) collision resistance cannot be satisfied by any single (fixed) function. Indeed, for any shrinking function, there exist algorithms that can efficiently find collisions, by simply having such collisions hardwired in their code. Accordingly, in the theoretical treatment of collision-resistance, we consider *keyed families of hash functions*, requiring that efficient algorithms cannot find collisions when the key is chosen at random. Due to this modeling, applications often require additional trust assumptions or rounds of communication (to set up the key). Furthermore, this model does not align with practice, where fixed cryptographic hash functions such as SHA-2 are widely used.

In light of the above, a common approach to analyzing keyless hash functions is to consider restricted adversarial models, for example, the class of uniform adversaries whose description is smaller than the size of the hash function's inputs. In practice, however, for any reasonable choice of hash function, the adversary's description may very well be larger than the input size. Other common paradigms for dealing with keyless hash functions include the random oracle methodology [BR93] and the human ignorance approach of Rogaway [Rog06]. (See further details in the related work section.)

We suggest a new paradigm for the treatment of keyless hash functions based on multi-collision resistance. The paradigm aims to guarantee security in the standard model against adversaries with arbitrary (polynomial-size) non-uniform description, based on a well-defined simple hardness assumption on hash functions. We observe that while keyless hash functions cannot be collision resistant, they may satisfy multi-collision-resistance if we only require that the number of collisions that the adversary can find is not much larger the adversary's description (including the size of its non-uniform advice).

More formally, consider any fixed hash function

$$\left\{\mathsf{H} : \{0,1\}^{2\lambda} \rightarrow \{0,1\}^{\lambda}\right\}_{\lambda\in\mathbb{N}} \quad ,$$

where λ is a security parameter, and note that multi-collisions of size 2^λ always exist. We will say that H is *K-collision resistant*, for a polynomial $K(\cdot)$, if for any two polynomials $\zeta(\cdot)$ and $T(\cdot)$, no adversary with non-uniform advice of size $\zeta(\lambda)$ and running-time $T(\lambda)$ can find $K(\zeta)$ distinct inputs:

$$X_1, \dots, X_K \qquad \textit{such that} \qquad \mathsf{H}(X_1) = \dots = \mathsf{H}(X_K) \enspace .$$

Crucially, the same collision bound $K(\zeta)$ holds regardless of the (polynomial) running time T, and depends only on the size ζ of the non-uniform advice. The larger K is the weaker the notion is, and our results throughout can be based on any polynomial K. (For concreteness, the reader may think of K as a specific polynomial, say quadratic.)

The Keyed Setting: Relaxing Collision Resistance. Another motivation for studying multi-collision resistance comes from the setting of keyed hash functions, which is fundamentally different from the keyless setting. In the keyed setting, multi-collision resistance is a natural relaxation of standard collision resistance, and may potentially be based on weaker assumptions. Here the collision bound K does not have to depend on the size of non-uniform advice. It could be a fixed polynomial (in the security parameter independent of the size of the adversary's non-uniform advice) or any constant larger than one, where $K = 2$ corresponds to the standard setting of keyed collision-resistant hashing. (We refer to the discussion in Section 1.3 for more details on the distinction between the keyless and keyed settings.)

Organization. The rest of this introduction is organized as follows. In Section 1.1, we present our main results on the applications of multi-collision-resistant hash functions. In Section 1.2, we discuss possible candidates. In Section 1.3, we discuss connections between multi-collision resistance and other notions in cryptography and complexity theory. In Section 1.4, we give a technical overview of the main ideas behind our results.

1.1 Results

As our main result, based on keyless multi-collision-resistant hashing, we resolve a long standing open question, showing a zero knowledge protocol with the optimal round complexity of three messages. In Section 1.1.1, we elaborate on this result, and present other results that improve the state-of-the-art round complexity in several central applications. In Section 1.1.2, we present additional applications of multi-collision-resistance, beyond round reduction. Section 1.1.3 addresses the core technical result behind our applications, and some of its information-theoretic implications.

1.1.1 Round Reduction

We now present our results regarding round reduction. Some of them rely on quasipolynomial hardness assumptions. Here when we say that a keyless multi-collision-resistant hash function is quasipolynomially hard, we mean that adversaries with polynomial advice ζ and quasi-polynomial running time T fail to find $K(\zeta)$-collisions (again, K can be thought of as any concrete polynomial, e.g. quadratic).

3-Message Zero-Knowledge Arguments. While 4-message zero-knowledge arguments for **NP** are known based on one-way functions [FS89, BJY97],[1] the existence of 3-message zero-knowledge (with negligible soundness error) has been a long standing open problem. We prove:

Theorem 1.1 (Informal)**.** *Assuming keyless multi-collision-resistant hash functions and LWE,* [2]*both quasi-polynomially hard, there exist 3-message zero-knowledge arguments.*

Three messages is the optimal round complexity for zero-knowledge arguments; indeed, 2-message zero-knowledge is impossible (for non-trivial languages) [GO94]. Beyond optimality, the difficulty and importance of this question stem from the fact that proving security (specifically, simulation) of 3-message protocols necessarily requires using the code of the adversary in a non-black-box way [GK96a]. While non-black-box techniques have been known since the work of Barak [Bar01], so far they have fallen short of solving this problem. Indeed, 3-message zero-knowledge arguments were only shown based on auxiliary-input knowledge assumptions [HT98, BP04, CD09, BCC+14], which are false assuming indistinguishability obfuscation exists [BCPR14] (these protocols also do not have an explicit zero-knowledge simulator). Other natural candidates for 3-message zero knowledge, like the parallel repetition of classical protocols with constant soundness (e.g., [GMW91]) are not known to have a simulator, and in fact there is evidence that they cannot be zero knowledge [BLV03, KRR17]. In contrast, 3-message protocols have been shown in restricted adversarial models where either the verifier or the prover is assumed to be uniform [BCPR14, BBK+16], or in models where the simulator can run in super-polynomial time [Pas03].

Our protocol is in the plain model, secure against non-uniform verifiers and provers of arbitrary polynomial size, and has an explicit polynomial-time simulator. The simulator is (inherently) non-black-box in the code of the verifier.

3-Message Succinct Arguments. In succinct argument systems, the verifier can certify the correctness of an **NP** statement, in time that is independent of the witness size. Such arguments have been the subject of a long line of research and are strongly motivated by the problem of delegating computation. Kilian gave a 4-message succinct argument for **NP** based on collision-resistant hashing [Kil92] (later extended to a *universal argument* in [BG08]). So far, this round complexity was only improved in the random oracle model [Mic00], or under strong knowledge assumptions [DCL08, BCC+14, BCCT13]. We prove:

Theorem 1.2 (Informal)**.** *Assuming keyless multi-collision-resistant hash functions with quasi-polynomial hardness, there exist 3-message succinct arguments for* ***NP****.*

[1]We recall the distinction between *arguments* that are only computationally sound and *proofs* which are statistically sound.

[2]In the body, we formulate the theorem based on generic primitives that can all be instantiated from LWE.

The constructed system is, in fact, *universal* for quasi-polynomial computations in the sense of [BG08].[3] Finally, we note that if the hash functions are polynomially compressing, we can get succinct arguments for **NP** without assuming quasi-polynomial hardness. (By polynomially compressing, we mean that they shrink by a factor of $\lambda^{\Omega(1)}$ rather than by a factor of 2).

4-Message Zero-Knowledge Proofs. When considering zero-knowledge proofs (rather than arguments), the state of the art in terms of round complexity is five messages [GK96a]. Here, similarly to the case of 3-message arguments, Katz showed that 4-message zero-knowledge proofs cannot have a black-box simulator, except for languages in $\mathbf{NP} \cap \mathbf{coAM}$ [Kat12]. Recently, Fleischhacker, Goyal, and Jain gave evidence that 3-message zero-knowledge proofs are impossible altogether [FGJ18]. We prove:

Theorem 1.3 (Informal)**.** *Assuming keyless multi-collision-resistant hash functions, there exist 4-message zero-knowledge proofs for* ***NP****.*

Interestingly, the non-black-box component in our simulation is minimal: the simulator only utilizes the size of the adversary's code, but otherwise uses it as a black-box. Unlike most of the simulators in the literature, our simulator is (slightly) non-uniform (see further discussion in the technical overview section). The protocol is based on a new type of non-interactive statistically-hiding string commitments that are only *weakly binding*. Weak binding is analogous to multi-collision resistance — the adversary can only open the commitment to a few strings proportionally to the size of its non-uniform advice.

5-Message Public-Coin Zero-Knowledge Arguments. Constant-round public-coin zero-knowledge arguments are also subject to black-box lower bounds [GK96b]. Barak gave a 7-message protocol [Bar01], which was later improved to 6 messages [OV12]. We construct a 5-message public-coin zero-knowledge protocol. We prove:

Theorem 1.4 (Informal)**.** *Assuming keyless multi-collision-resistant hash with quasi-polynomial hardness, there exist 5-message public-coin zero-knowledge arguments for* ***NP****.*

A downside of our 5-message protocol is that it requires quasi-polynomial hardness of the multi-collision-resistant hash functions, whereas existing protocols are based on polynomially-hard keyed collision-resistance [BG08]. We can relax the hardness to *any* super-polynomial function, assuming that the hash functions are polynomially compressing.

1.1.2 More Applications

We discuss several applications of multi-collision-resistant hash functions beyond round reduction.

Succinct Arguments with Preprocessing. Various applications require that all parties agree on a common hash function. In settings where they cannot all interact, there may not be a way to jointly choose a trusted key and a keyless hash function is required. While keyless functions cannot be collision resistant, some application are still possible based on multi-collision-resistance. We demonstrate this through a notion that we call *succinct arguments with preprocessing*. Here a single server wishes to prove an **NP** statement of global interest to many clients. The server can preprocess the statement and witness offline. Then when any client requests a proof, the running time of the server (modeled as a random access machine) is required to be much shorter than the possibly long **NP** witness.

Our 3-message succinct arguments, based on keyless multi-collision resistance, give rise to a prover with low online complexity (poly-logarithmic in the witness size). No such protocols were previously known, not even under knowledge assumptions.

Relaxing Assumptions in the Keyed Setting. The results in Section 1.1.1 have analogous versions in the keyed setting. Specifically, all of the above theorems hold when assuming keyed (rather than keyless) multi-collision resistant functions, where each protocol has one extra message (the other conditions in the theorems remain as is). While this is identical to the state of the art in terms of round complexity, we do relax the complexity assumptions from (keyed)

[3]Roughly speaking, plain succinct arguments guarantee soundness only as long as the prover is restricted to **NP** computations (i.e., there is polynomial bound on the time of these computations). Universal arguments provide soundness (against polynomial-size provers) even for the universal language that includes also non-deterministic computations of super-polynomial time. See further details in [BG08].

collision-resistance to (keyed) multi-collision resistance in 4-message succinct arguments, 5-message zero-knowledge proofs, and 6-message public-coin zero-knowledge arguments.[4] Indeed, keyed multi-collision-resistant hash functions have recently been constructed from complexity assumptions that are not known to imply collision-resistance such as average-case hardness of the *Ramsey problem* [KNY17b] and variants of the *entropy approximation problem* [BDRV17]. We do suffer, however, a quasi-polynomial security loss. (This loss can be avoided in certain cases. In particular, assuming that the collision bound K is constant, which is possible in the keyed setting, this loss is avoided in the application of succinct arguments for **NP**.)

Concrete Security of Cryptographic Hash Functions. In practice, cryptographic systems are instantiated with one of a few standard keyless hash functions, for example SHA-2. The notion of multi-collision resistance provides a framework for proving formal security guarantees for such instantiations. Our approach is to express the hardness of the hash function as a concrete (non-asymptotic) bound on the size of a multi-collision that can be generated by any adversary as a function of its description size and running time. Starting from any system based on multi-collision resistance, we can instantiate it with a specific hash function and, based on the hash's hardness, derive bounds on the size and running time of any feasible adversary.

We contrast this with the “human ignorance" approach of Rogaway [Rog06]. There the focus is on establishing an *explicit reduction* from breaking a system to finding a (plain) collision in the underlying hash. Such reductions provide confidence in the system, but only so long that the particular hash function resists humanity's efforts to find even a single collision. Unlike our framework, this approach does not allow to prove security with respect to any well-defined class of adversaries.

1.1.3 The Core Technical Result

The main technical contribution behind our results is a transformation from (keyed or keyless) multi-collision-resistant hash functions for a fixed input length to ones with an arbitrary input length and a *local opening* property. For collision-resistant hash functions such domain extension and local opening properties are quite basic [Mer89, Dam89]. In contrast, for multi-collision resistance, domain extension, and especially local opening, require new ideas.

As first observed by Joux [Jou04], unlike collision-resistance, multi-collision resistance is *not robust under composition*. For instance, in the canonical *hash tree* construction [Mer89], collision-resistance is easily argued by observing that a collision between two long inputs immediately gives rise to a collision in the underlying (fixed length) hash function. This construction also allows to locally open any specific input bit and certify its uniqueness by providing only a small (logarithmic) number of hash values. This construction completely fails when considering multi-collision resistance. Indeed, when instantiating the tree construction with a K-collision resistant function, collisions multiply — the number of potential openings for any specific leaf could be $(K-1)^d$, where d is the depth of the tree. Furthermore, it is possible to “mix-and-match” the values of different leafs so that the overall number of openings for an n-leaf tree may be $(K-1)^{d\cdot n}$. For $K = 2$, corresponding to full collision-resistance, this is not so bad, due to the miraculous fact that $1 \times 1 = 1$. However, for $K > 2$ this is devastating as the number of openings is exponential in the input length!

To overcome the above, we combine known tree hashing techniques, together with a new information-theoretic tool: a variant of list-recoverable codes that we call *collision-free codes*, which may be of independent interest. In particular, these codes give rise to probabilistically checkable proofs (PCPs) with a strengthened soundness guarantee. We refer the reader to the technical overview in Section 1.4 for further details.

1.2 Candidates for Multi-collision-Resistance

Our results rely on the assumption that keyless multi-collision resistant hash functions exist. At this point, we do not know how to reduce this assumption to any other standard cryptographic assumption. We do, however, find this assumption clean and simple. Furthermore, the assumption that a given hash function is K-collision resistance for a given K is a falsifiable assumption [Nao03].[5]

[4] 4-message zero-knowledge arguments are already known from one-way functions [BJY97]; thus, when translated to the keyed setting, our 3-message zero-knowledge protocol, which now has 4 rounds, does not yield a new result.

[5] That is, given an efficient attacker of a given description size ζ, we can efficiently test if it outputs a $K(\zeta)$-collision.

A natural place to start the search for a multi-collision-resistant hash function is existing constructions of cryptographic hash functions. These constructions can be roughly divided into two groups: structured and unstructured ones. By structured hash functions, we mean keyed hash functions for which collision-resistance can be reduced to structured mathematical assumptions such as Discrete Log, Factoring, or LWE. By unstructured hash functions, we mean keyless functions such as SHA-2 or Keccak, which are designed according to heuristic principles guided by cryptanalysis best practice and efficiency considerations.

One approach to designing multi-collision resistant hash functions is by taking a keyed structured hash function and fixing its key in some way (say, by taking the first digits of π). We observe that starting from known structured hash functions, this approach is bound to fail, regardless of how we choose the key. This is due to the fact that for these hash functions there exists a *short trapdoor* for every key that enables to find many collisions, or even sample random ones. While this trapdoor is hard to find given a random key, when fixing the key, this trapdoor can always be taken as non-uniform advice.

In contrast, unstructured hash functions are natural candidates for multi-collision resistance. They are natively unkeyed and their unstructured design deliberately aims to eliminate trapdoors. In the asymptotic setting, we focus on functions that can be naturally scaled to have increasing output and security. For example, hash functions based on the AES cipher use algebraic components that can be scaled to arbitrarily large lengths [DR02, BRS02]. Alternatively, we can make a concrete multi-collision resistance assumption on specific functions such as SHA-2.

Further evidence for the multi-collision resistance of unstructured hash functions comes from the popular random oracle paradigm. We show that random oracles satisfy multi-collision resistance even in the model of *random oracles with auxiliary inputs* of Unruh [Unr07]. That is, we show that even given (short) non-uniform advice that *may depend arbitrarily on the random oracle*, it is hard to find multi-collisions that are larger than this advice. Thus, any attack on the multi-collision resistance of a cryptographic hash function would constitute a strong and natural separation between the hash and random oracles. For several cryptographic hash functions used in practice, the only known separations from random oracles are highly contrived [CGH04].

Another potential candidate for multi-collision resistance is given by Goldreich's one-way functions. In this combinatorial construction every output bit is computed by applying an appropriate predicate to a small number of input bits, where the dependence is chosen by an expander graph. While the security properties of Goldreich's one-way function are the subject of extensive research, its collision resistance is still not well understood [AM13, App16].

1.3 More on Multi-collision Resistance

We further discuss the notion of multi-collision resistance, considering both the case of keyed and keyless functions.

Relations to Existing Primitives. Multi-collision-resistant hash functions (keyed or keyless) imply the existence one-way functions. Specifically a multi-collision-resistant hash may not be one-way function by itself, but it is a *distributional one-way function*. That is, for a random image Y, it is computationally hard to sample (statistically-close-to) uniform preimages of Y.[6] Such distributional one-way functions are known to imply (plain) one-way functions [IL89].

As expected, keyed multi-collision-resistant hash functions follow from any assumption that implies collision-resistance. However, we do not know of any standard cryptographic primitive that implies keyless multi-collision resistance. Indeed, as we explain next, keyless multi-collision-resistance seems to have a different complexity-theoretic nature than that of typical cryptographic primitives. In the converse direction, we do not know if keyless multi-collision resistance implies keyed collision-resistance or any cryptographic primitive other than one-way functions.

Keyless Hashing and Universal Hardness. The notion of hardness encapsulated in keyless multi-collision-resistance is not captured by our standard treatment of hard search problems. Indeed, we usually think of hard search problems as *instance based*. Worst-case hardness says that any algorithm fails to find a solution at least for some instances, whereas cryptography requires a stronger notion of average-case hardness, where all algorithms fail to find solutions for instances from a single distribution. In contrast, in keyless multi-collision-resistance *there are no instances* or,

[6]To see why this is the case, note that any such sampler could be used to find a multi-collision of arbitrary polynomial size, in particular much larger than the polynomial advice required by the sampler. This can be done just by sampling enough preimages for some random image Y.

in other words, the problem given by a hash function H has one *universal* sequence of instances $\left\{1^\lambda : \lambda \in \mathbb{N}\right\}$. The so-called solutions for the instance 1^λ are all the multi-collisions

$$\left\{X_1, \ldots, X_k \in \{0,1\}^{2\lambda} : k \geq 2,\ \ X_i \neq X_j,\ \ \mathsf{H}(X_1) = \cdots = \mathsf{H}(X_k)\right\} \ \ .$$

Clearly, we cannot expect that non-uniform algorithms with arbitrary polynomial advice completely fail to solve such problems. All that we can expect is that they fail to find *solutions that are much longer than their non-uniformity*. In other words these are hard *compression problems* — it is impossible to compress long solutions into short advice that can be efficiently decompressed.

From a complexity-theoretic perspective, such universal problems may be of independent interest. Understanding their implications and feasibility calls for further research.

An Alternative Definition of Multi-collision Resistance. So far we have only considered K-collision resistance where K is a polynomially bounded function of the security parameter (and in the keyless setting, also of the size of the adversary's non-uniform advice ζ). Aiming to rely on the weakest possible notion of multi-collision resistance, we may want to consider also a super-polynomial collision bound $K = \lambda^{-\omega(1)}$. In this case, to make the definition above meaningful, we must also allow the adversary to run in super-polynomial time proportional to K. The resulting definition, therefore, becomes incomparable to the original definition where both the collision bound K and the adversary's running time were polynomial in the security parameter.

We consider an alternative definition that allows capturing super-polynomial values of K while still considering only polynomial-time adversaries. The definition is in the spirit of the *inaccessible entropy* notion of [HRVW09]. Roughly speaking, the definition requires that a polynomial-time adversarial sampler cannot output preimages for any image Y with entropy higher than $\log K$ (in the terminology of [HRVW09], the sample has low accessible entropy). A bit more concretely, we require that for every adversary and hash output Y, there exists a set S_Y of K values, so that the adversary cannot find a preimage of Y that is outside S_Y, except with negligible probability.

Note that in this definition, a successful adversary does not explicitly output a K-collision, but only provides a succinct representation of such a collision. Indeed, this definition is meaningful even when K is super polynomial, but the adversary runs in polynomial time (when K is polynomial, the definition is equivalent to the previous one). As we shall see later, this style of definition is also essential in the setting of multi-collision resistant hashing *with local opening* (as discussed in the next section). Finally, we note that all of our results, excluding 4-message zero-knowledge proofs, can be based on K-collision resistance for super-polynomial K, however, the soundness reduction's loss grows polynomially with K. This loss occurs even if we start with the alternative definition just suggested. Avoiding this loss is an interesting problem.

A Stronger Notion of Multi-collision Resistance. We conclude the discussion by pointing out a seemingly stronger notion of hash functions that we simply call *strong multi-collision resistance*. According to this notion, it is not only hard to find K elements that are hashed to the same value, but rather it is hard to find K distinct colliding pairs $(X_1, X'_1) \ldots (X_K, X'_K)$ where each pair may hash to a different value.

Indeed, this notion clearly implies the previous notion of multi-collision resistance, and seems stronger. In fact, we observe that any sufficiently shrinking strong multi-collision resistant hash (keyed or keyless) implies keyed hash functions that are collision resistant in the standard sense. Specifically, given a strong K-collision resistant hash $\mathsf{H} : \{0,1\}^{3\lambda} \to \{0,1\}^\lambda$ for $K = 2^{\lambda - \omega(\log \lambda)}$, the keyed family $\mathsf{H}'_{X_1} : \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ defined by $\mathsf{H}'_{X_1}(X_2) = \mathsf{H}(X_1 X_2)$ is collision resistant.

While in the keyed setting strong multi-collision resistance is equivalent to full collision resistance, in the keyless setting this does not appear to be the case. Whereas collision resistance is impossible, strong multi-collision resistance seems plausible. In particular, the candidate multi-collision resistant hash functions described above may satisfy this notion. Relying on strong multi-collision resistance, we can avoid the quasi-polynomial loss in some of our reductions (without resorting to polynomially-compressing hash functions). See further details in the next section.

The Compression Rate. Our results were stated in their simplest form where the underlying hash functions compress $\ell = 2\lambda$ bits to λ bits. More generally, the same results hold for any *linear compression*, meaning that $\ell = \lambda \cdot (1 + \Omega(1))$. As already noted, in several cases, we can reduce the security loss in our results (from quasi-polynomial to polynomial or slightly-super-polynomial) by relying on *polynomially compressing* hash functions, meaning that $\ell = \lambda^{1+\Omega(1)}$.[7]

[7]Alternatively, we can keep relying on linear compression, and avoid this loss by assuming strong collision resistance, or in the

Candidates with polynomial compression may be more scarce than ones with linear compression (for instance some existing cryptographic hash functions have linear compression). Applying our domain extension technique (described in the next section) to candidate multi-collision resistant hash functions with linear compression gives a plausible candidate.[8] It is also interesting to note that, unlike linearly-compressing hash functions, polynomially-compressing ones have a *universal construction* in the sense of Levin [Lev87].[9]

1.4 Technical Overview

In this section, we overview the main ideas behind our results based on the existence of multi-collision resistant hash functions. To build intuition, we start with a simple application to commitment schemes. We then move to discuss the main technical challenge centered around the problems of domain extension and local opening. We describe our solution to this problem and how it is applied to obtain our results on round-efficient protocols.

Non-Interactive Commitments with Weak Binding. Commitments are a basic building block in cryptographic protocols such as zero-knowledge proofs. They consist of a commitment phase, where a sender commits to a value, and an opening phase, where the commitment is opened and the value is revealed. The commitment should be *hiding*: the receiver does not learn anything about the committed value before the opening phase, and *binding*: the sender cannot open the commitment to more than one value.

Collision-resistant hashing has been essential in achieving two useful properties for commitment schemes. First, they are used to obtain *shrinking commitments*, where the communication in the commit phase is shorter than the committed value. Second, they are used to construct constant-round *statistically-hiding commitments* where hiding is guaranteed against unbounded receivers [DPP93, HM96]. Since collision-resistant hash functions must be keyed, the corresponding commitments have a two-message commit phase where in the first message, the receiver specifies the hash key.

Replacing keyed collision-resistant hashing with keyless multi-collision-resistant hashing, we get a non-interactive (one commitment message) shrinking and statistically-hiding commitments with *weak binding*: the sender may be able to open the commitment to more than one value, but not to too many values (say, polynomial in the size of its non-uniform advice). We then observe that in many applications of commitments, weak binding is sufficient. We proceed to give examples.

Barak's Protocol. A first example is the public-coin constant-round zero-knowledge protocol of Barak [Bar01]. To prove an **NP** statement $x \in \mathcal{L}$, the protocol proceeds in two phases. In a preamble phase the prover sends a shrinking commitment c to the code of some (potentially long) program Π and the verifier responds with a random string r, much longer than the commitment c. Then, in a proof phase, the prover gives a succinct witness-indistinguishable argument of knowledge proving that either $x \in \mathcal{L}$ or that the committed program $\Pi(c)$ happens to output r. By committing to the code of the verifier itself, a non-black-box simulator is able to produce an accepting transcript without using the witness. Still, a cheating prover, who does not know the verifier's randomness r, can only commit to such a program with negligible probability.

We can shave one message from Barak's protocol by replacing the prover's commitment with a weakly-binding commitment. Roughly speaking, the resulting protocol is still sound because even if the prover can open its commitment to polynomially many programs, with overwhelming probability, none of these programs predicts r. Namely, the soundness error increases by a factor of K, where K is the number of values that the prover can open.

Barak's protocol also relies on collision-resistance to implement (4-message) universal arguments. To avoid collision-resistance altogether, and rely only on multi-collision resistance, we use the universal arguments that we construct from multi-collision resistance (described later in this section).

keyed setting, by assuming a constant collision bound.

[8]The multi-collision resistance of the resulting hash can be formally proved assuming that the basic linearly-compressing function is strongly multi-collision resistant. However, this may not be necessary; namely, even if the underlying hash is not strongly multi-collision resistant the extended construction may still be multi-collision resistant.

[9]Concretely, if there exists a K-collision resistant hash that compresses $\lambda^{1.01}$ to λ bits and can be uniformly computed in polynomial time $t(\lambda)$, then enumerating all such functions, applying them to the input, and concatenating the outputs is a universal construction of a polynomially-compressing hash function.

Domain Extension. The above example is, in fact, oversimplified. To prove that Barak's protocol is zero-knowledge against verifiers of arbitrary (polynomial) size, the simulator must be able to commit to arbitrarily long programs. Therefore, the hash H underlying the commitment scheme must shrink arbitrarily long input strings in $\{0,1\}^*$ to output strings of some fixed length, say λ. Transforming a fixed-domain hash function $\mathsf{H} : \{0,1\}^{\ell(\lambda)} \to \{0,1\}^{\lambda}$ to one over arbitrary inputs in $\{0,1\}^*$ is known as *domain extension* [Dam89, Mer89]. As we have already explained, while existing domain-extension techniques preserve collision resistance, they destroy multi-collision resistance — the bound on the collision size grows exponentially with the input length.

We give a new domain extension for multi-collision-resistant hash functions. The construction consists of a cryptographic component (based on multi-collision resistance) and an information-theoretic component. We now proceed to describe each of the two.

Component 1: Hash Trees. The first component is a standard hash tree construction [Mer89], which we now describe more concretely. Given a basic hash H shrinking strings of length 2λ to strings of length λ, we hash a long string by splitting it into blocks of length λ and using H in the form of a binary tree, to hash all blocks down to a single root. We observe that if H is resistant against collisions of size K, then the tree construction provides a *local* multi-collision resistance guarantee: for every index i, it is hard to find many long strings $X \in \left(\{0,1\}^{\lambda}\right)^{n}$ that hash to the same root such that their i-th block $X[i] \in \{0,1\}^{\lambda}$ takes more than $(K-1)^d$ distinct values, where $d = \log n$ is the depth of the tree. We emphasize that the guaranteed local multi-collision resistance is weaker than full multi-collision resistance since it may be possible to "mix-and-match" collisions for different blocks. Accordingly, the bound on the number of global collisions still grows exponentially with the length of the input (rather than the depth of the tree).

The bound $K^{\log n}$ on the collision size for any individual block is the source of the quasi-poly loss in some of our reductions. It can be improved at the cost of starting from a stronger basic hash H:

- Given a polynomially compressing hash H that, say, shrinks strings of length λ^2, to strings of length λ, we can use a tree of arity λ and improve the bound on the collision size: for input strings of polynomial length $n = \lambda^c$, it is hard to find a K^c-collision for any individual block.[10] The construction in the body is described based on such hash functions.

- In the keyed settings, if H is K-collision resistant for a *constant* K (even with linear shrinkage), then it is still hard to find a $\mathrm{poly}(\lambda)$-collision for any individual block for input strings of arbitrary polynomial length.

- If H is *strong* K-collision resistant hash (for any K), the binary tree construction (relying on linear compression) yields tight parameters: it is hard to find more than $\tilde{O}(K)$ collision for any individual block.

Component 2: Rectangle-Evasive Codes. To go from the local multi-collision-resistance guarantee provided by the hash tree to full multi-collision-resistance, we force a certain global structure on the inputs by first encoding them with an appropriate code. Specifically, let $C : \{0,1\}^{\ell} \to \left(\{0,1\}^{\lambda}\right)^{n}$ be a code that maps ℓ bits to n blocks, each of λ bits. We hash inputs by first encoding them, and then hashing the encoded input with a hash tree. We require that the code is *rectangle evasive* in the sense that it has small intersection with any rectangle with relatively small sides. That is, for every sequence of sets $S_1, \dots, S_n \subseteq \{0,1\}^{\lambda}$, each of size K, the number of codewords in the rectangle $S_1 \times \cdots \times S_n$ is at most $K' = \mathrm{poly}(K)$. This construction of a hash tree (with an underlying K-collision-resistant hash) on top of a rectangle-evasive code, gives a K'-collision-resistant hash function for an arbitrary polynomial-size domain.

Such rectangle-evasive codes are a special case of list-recoverable codes [GI01], which more generally bound the number of codewords that are *close* to the rectangle, and can be based on known unbalanced expanders [GUV09]. Such codes have been previously used to obtain strong forms of domain extension for collision-resistant hash functions [MT07, DS11, HIOS15].

Succinct Arguments and Local Opening. Many applications of collision-resistant hashing require domain extension with stronger properties such as efficient local opening. For example, in Kilian's succinct argument system for NP, the prover constructs a PCP proof of the statement and commits to the long proof with a hash tree. The verifier then attempts to verify the PCP by asking the prover to open a few random locations of the committed proof. By exploiting the tree structure of the hash, the prover can open the requested locations and prove succinctly that the values are

[10]Generally, any polynomial compression is sufficient, where a hash that is less compressing yields a larger polynomial loss in the collision bound.

consistent with the committed root without opening the entire tree. It only needs to provide the hash values on the paths from the opened locations to the root (along with their siblings).

Formalizing the local opening in the setting of multi-collision resistance requires more care. Simply requiring that every subset of input locations can be opened in a small number of different ways is not sufficient in applications such as Kilian's arguments. Instead, we make a stronger requirement in the spirit of the inaccessible entropy definition discussed in section Section 1.3. The requirement intuitively says that there exist a small number of global inputs, such that any subset of locations can only be opened consistently with one of them. A bit more accurately, we require that given any (randomized) adversary that produces a hash value and then successfully samples an opening for some subset of locations, it is possible to efficiently extract at most K global inputs, such that any subset of locations opened by the adversary is consistent with one of the global inputs with overwhelming probability. We emphasize that when multiple locations are opened simultaneously they are required to be consistent with the same global input. Therefore, consistency proofs for different locations must be correlated. We cannot simply open every location independently as in the case of collision-resistant hash trees.

We observe that Kilian's argument can be instantiated based on any hash with local opening that satisfies the above notion of multi-collision resistance. Given any cheating prover that tries to prove a false statement, we can extract K full PCP proofs. By the (negligible) soundness of the PCP system, none of the K proofs is convincing with overwhelming probability. Since the prover must always answer any subset of queries consistently with one of these proofs, soundness is guaranteed.

Achieving Local Opening. The domain extension construction for multi-collision resistance, based on rectangle-evasive codes, does not support local opening — to open even a single location of the input, one must first open all the leaves of the hash tree, check that the result is a valid codeword, and only then decode it.

The first idea toward achieving local opening is to use a rectangle-evasive code C that is also *locally decodable*. To open one location of the input, select a small set of locations D that can be used for local decoding and open the hash-tree leaves that contain the locations in D. This approach however does not guarantee (global) multi-collision resistance — even if every rectangle contains at most K' global codewords, there could be much more than K' codewords that are consistent with the rectangle on a small set of locations such as D.

Collision-Free Codes. We design a new type of code that we call *collision free*, which will suffice for constructing a multi-collision-resistant hash with local opening. The code has the following local testing flavor. To decode a location i of the input word, we read a small set of locations D of the codeword together with a small random set of *test locations* T. The test locations are independent of i and intuitively, are used to check consistency of decoding for any location. That is, when decoding the value of the i-th location of the input word, we also verify consistency between the values in the locations given by D and those given by T.

Collision freeness says roughly the following: for every rectangle $\mathbf{S} = S_1 \times \cdots \times S_n$ such that each S_i is of small (say polynomial) cardinality, with high probability over the choice of T, for any location i and set D used to decode the location i, there are no partial codewords C and C' that collide in the following sense:

- Both C and C' are consistent with the rectangle $\mathbf{S}$.[11]
- In both C and C', the values in locations T and D satisfy the consistency test.
- C and C' agree on the locations T, but not on D.

In other words, with high probability, an assignment to the test locations, completely fixes how any location is decoded, provided that the symbols read are always taken from the rectangle $\mathbf{S}$.

Based on collision-free codes, the construction of multi-collision-resistant hash with local opening is as follows: to open a set of locations $i_1, \dots i_k$ of the input, sample a set of test locations T and sets $D_1, \dots, D_k$ for decoding the locations $i_1, \dots i_k$ of the input. Open the leaves of the hash tree that contain these locations and verify the consistency of the values in locations T and D_j for every $j \in [k]$. If all values are consistent, decode the input locations $i_1, \dots i_k$.

By the local multi-collision-resistance of the hash tree, there is some rectangle $\mathbf{S} = S_1 \times \cdots \times S_n \subseteq \{0,1\}^n \times \lambda$ where each S_i is of size at most K, such that every opening for the locations T and $\{D_j\}$ is consistent with $\mathbf{S}$. It follows

[11]In more detail, by a partial codeword C, we mean that $C = (C_i \mid i \in U)$ is a partial assignment for a subset $U \subseteq [n]$ of the blocks. C is consistent with the rectangle $\mathbf{S} = S_1 \times \cdots \times S_n$ if for any $i \in U$, $C_i \in S_i$.

that the locations T can take at most $K^{|T|}$ combinations of values. By the collision freeness of the code, the values for locations T fix some global input word x such that the decoded values in locations $i_1, \dots i_k$ are consistent with x. The collision bound of the final construction is therefore $K^{|T|}$. To minimize this bound, we design a collision-free code where the size of the set T is small (for a natural setting of parameters, where n and K are polynomial in λ, $|T|$ will be a constant).

Collision-Free Polynomial Code. We construct a collision-free code C based on low-degree multivariate polynomials (the code can be seen as an over-redundant variant of Reed-Muller codes). For every m, the code maps strings of length $\ell = \lambda^m$ to a codeword that consists of $n = \mathrm{poly}(\ell)$ blocks. We first describe the code for $m = 2$ and then generalize the construction to arbitrary m. Let $\mathbb{F}$ be a field of size $\mathrm{poly}(\lambda)$, and let $H \subseteq \mathbb{F}$ be a subset of size λ. To encode an input $x \in \{0,1\}^{\lambda^2} \simeq \{0,1\}^{H^2}$, we first compute the low-degree extension P_x of x. That is, $P_x : \mathbb{F}^2 \to \mathbb{F}$ is a bivariate polynomial of (individual) degree $\lambda - 1$ whose evaluations on the square H^2 encode x. The codeword $C(x)$ consists of the restrictions of P_x to all horizontal and vertical lines. That is, the codeword consists of $n = 2|\mathbb{F}|$ blocks, each describing a degree $\lambda - 1$ univariate polynomial.

We consider a rectangle $S_1 \times \cdots \times S_n$ where $|S_i| \leq K$. The set T consists of τ random vertical lines, where τ is chosen such that $(|\mathbb{F}|/\lambda)^\tau > 2K^2 \cdot |\mathbb{F}|$. For example, if $K = \mathrm{poly}(\lambda)$ then $|T| = \tau$ is constant.[12] To decode the i-th input location, the set D includes the horizontal line that encodes x_i. The values in locations D and T are consistent if the restriction of P_x to the horizontal lines in D agrees with its restriction to all the vertical lines in T on their intersection points.

To show that the code is collision free, we note that fixing the values on τ random vertical lines fixes the value on τ random points for every horizontal line. We then argue that this is enough to fix a single value for each horizontal line, and thus a unique decoded value. In a bit more detail, let's restrict attention to some specific horizontal line. By consistency with the rectangle $\mathbf{S}$, there are at most K distinct univariate polynomials (each of degree $\lambda - 1$) that the line may take. By Schwartz-Zippel, each pair of these polynomials agrees on the τ random intersection points with the vertical lines in T with probability at most $(\lambda/|\mathbb{F}|)^\tau$. Thus, the probability that any such pair among the K polynomials agrees on these points is at most $K^2 \cdot (\lambda/|\mathbb{F}|)^\tau < 1/2|\mathbb{F}|$. By a union bound, with probability $1/2$ this will be the case for all $|\mathbb{F}|$ restrictions of P_x to horizontal lines. In the actual construction, this collision probability is reduced by standard amplification.

We extend this construction recursively for higher values of m. Specifically to encode an input $x \in \{0,1\}^\ell$ for $\ell = \lambda^m$, we extend x to an m-variate degree-$(\lambda - 1)$ polynomial $P_x : \mathbb{F}^m \to \mathbb{F}$. The codeword contains the restrictions of P_x to all $n = m|\mathbb{F}|^{m-1}$ axis-parallel lines. The test set T is now constructed recursively: we first sample τ random parallel hyperplanes $\mathcal{H}_1, \dots, \mathcal{H}_\tau$ of dimension $m - 1$ (instead of lines as in the two-dimensional case). Then, we recursively sample test locations within every such hyperplane $\mathcal{H}_i$, by sampling τ parallel hyperplanes of one dimension less. The recursion stops once the dimension is reduced to one. The resulting test set T consists of all τ^m axis parallel lines, which is a constant for polynomial-size inputs. To decode the i-th input location, we read one axis-parallel line γ that encodes x_i and is orthogonal to the hyperplanes $\mathcal{H}_1, \dots, \mathcal{H}_\tau$. We then recursively decode its intersection with each of the hyperplanes.

PCPs with Strong Soundness. The above construction can be interpreted as following a common approach in constructing succinct arguments — start with a PCP that is sound in a given adversarial model, and enforce this model using cryptographic tools. For instance, in [Kil94], the standard PCP model where the prover has to fully commit is enforced with a fully collision-resistant tree hash. In [KRR14], being restricted to two messages, they cannot enforce the full commitment model, but can enforce the *no signaling model* based on private information retrieval. To compensate, they construct a stronger PCP (for **P**) that is no signaling.

Viewing our construction through this perspective, based on multi-collision resistance, we get a weak type of commitment that only guarantees that every location can be opened to a small number of symbols K. Using collision-free codes, we can then construct an appropriate PCP (for **NP**) that remains sound even in this setting. That is, we allow the PCP prover to adaptively choose its answer from the corresponding set of symbols, as a function of all queries. To obtain PCPs with such strengthened soundness, we simply encode a standard PCP with the collision-free code, and

[12]When we use the code to construct a hash with local opening, the value of K may depend on the adversary's size, and is therefore not known ahead of time. We thus apply the code for all values $\tau \leq \bar{\tau}$, for a slightly super-constant function $\bar{\tau} = \omega(1)$. Then, only in the analysis we restrict attention to the relevant τ.

read every PCP query using the code's local decoder. We note that we are interested in the parameter setting where the alphabet is of size $n^{\omega(1)}$ and K is an arbitrary polynomial $n^{O(1)}$, and the soundness error is $n^{-\omega(1)}$ (where n is the size of the **NP** instance). We leave as an open question the exploration of different settings of parameters and more applications for such strong PCPs.

3-Message Zero-Knowledge Arguments. Our starting point is the 3-message zero-knowledge argument of Bitansky et al. [BBK+16] in *the global hash model*. In this model, the prover and verifier agree on a hash function before interacting. The soundness of the protocol is reduced to finding collisions in the hash. We get a protocol, in the plain model, relying on an keyless multi-collision-resistant hash.

The protocol of Bitansky et al. relies on a *memory delegation* scheme. In memory delegation schemes, the verifier gets access to a short digest of a long memory string. The verifier then sends the description of a computation to be executed on the memory, together with a cryptographic challenge, and the prover responds with the computation's output and a proof of correctness. The protocol is sound in the sense that the prover cannot convince the verifier that the same computation has two different outputs with respect to the same memory digest. Based on memory delegation, Bitansky et al. give a 3-message private-coin version of Barak's protocol. They essentially use memory delegation construct a succinct 3-message witness-indistinguishable argument.

In their protocol, the collision-resistant hash is used both to instantiate the memory delegation protocol, and in the transformation from memory delegation to zero-knowledge. In this overview, we focus on achieving the former based on multi-collision resistance.[13] We first consider the task of memory delegation in an intermediate model where, instead of sending a digest of the memory, the prover publishes an encoding of the entire memory as an oracle. The verifier delegates a computation to be executed on the memory and can verify the proof of correctness by making only a few queries to the encoded memory given by the oracle. We also require that the verifier's oracle queries depend only on its private coins and not on the proof. We note that such *oracle memory delegation* follows from the protocol of [KRR14].

We then go from oracle-based memory delegation to standard memory delegation in two steps. First, the prover commits to the encoded memory oracle with a multi-collision-resistant hash with local opening. Then, together with its challenge, the verifier sends its queries to the oracle under fully-homomorphic encryption. The prover responds with the proof (in the clear) and the oracle answers (under the encryption). We prove, based on the multi-collision-resistance of the commitment and semantic security of the encryption, that the memory delegation scheme remains sound even when the verifier's encrypted queries are given to the prover.

Since the oracle commitment relies on multi-collision-resistant hash, we are only guaranteed that the oracle's answers are consistent with one of a small number of oracles. We, therefore, get a weaker soundness guarantee from the memory delegation — the prover cannot convince the verifier that the same computation has *many* different outputs with respect to the same memory digest. We show that when instantiating the 3-message zero-knowledge argument of Bitansky et al. with such weak memory delegation soundness is preserved (similarly to the 5-message version of Barak's protocol from weakly binding commitments described above).

4-Message Zero-Knowledge Proofs. We show how to modify the 5-message zero-knowledge proof system of Goldriech and Kahan [GK96a] to get a 4-message zero-knowledge proof based on keyless multi-collision-resistant hash.

In the protocol of [GK96a], the verifier first sends a statistically-hiding commitment to a random string r. The prover and verifier then execute a 3-message public-coin proof with negligible soundness [GMW91], where the verifier opens r as its random challenge. The 3-message proof has the property that for every r, a proof with challenge r can be efficiently simulated given r. Therefore the entire protocol can be simulated as follows: interact with the verifier until it opens the challenge r, and then rewind the verifier and simulate the proof using r. The simulation is successful since the verifier is bound to open the same value of r in every interaction.

We replace the verifier's two-message statistically-hiding commitment, with a non-interactive weakly-binding commitment. If the verifier's commitment is only weakly binding, when rewound, the verifier may open the commitment to a different value. Therefore, we have the simulator repeatedly rewind the verifier until it again opens the commitment to r. The expected time of this naïve simulation strategy is exponential since the verifier may open the commitment to any value r' with some (perhaps negligible) probability. Instead, we change the simulation to abort after roughly $1/\mu$

[13]We can, in fact, avoid the use of collision resistance assuming ZAPs [DN07].

rewinding attempts for some negligible function μ. Using the weak binding of the verifier's commitment, we show that there exists a polynomial-size set, such that the probability of opening a string r outside this set is at most μ. The actual simulation strategy is a bit more complicated and has to deal with similar obstacles to those in [GK96a].

A Note on Non-Uniformity. There are two common definitions of zero knowledge against non-uniform verifiers. The classical definition asks that any non-uniform verifier has a non-uniform simulator. A stronger definition is that of *universal simulation*, which asks that there's a single uniform simulator that can simulate any non-uniform verifier, given its code as auxiliary input. The definition we achieve is stronger than the classical one, but slightly weaker than the universal one. We show a universal simulator that is slightly non-uniform. Alternatively, we can achieve the standard universal definition if we assume that the underlying hash function is slightly super-polynomially secure. We can also achieve a universal simulator if we settle for the notion of ε-zero-knowledge [DNRS03].

1.5 Concurrent and Independent Work

The notion of multi-collision resistance was studied concurrently and independently by Komargodski, Naor, and Yogev [KNY17b, KNY17a] and by Berman et al. [BDRV17]. These works deal only with keyed hash functions, whereas we address both the keyed and keyless settings, focusing on the latter.

These works place the notion of keyed multi-collision-resistent hashing between one-way functions and collision-resistant hashing by proving black-box separations and by constructing such hash function from assumptions that are not known to imply full collision-resistance. In terms of applications, these works show how to use multi-collision-resistent hash functions instead of collision resistent ones in applications such as statistically-hiding commitments and statistical-zero-knowledge arguments. Komargodski, Naor, and Yogev also develop domain extension and local opening techniques. Thier constructions require more rounds of interaction compared to existing constructions based on collision resistance.

In contrast, our main objective is *to reduce round complexity or trust in applications*, which motivates our investigation of keyless hash functions. With this goal in mind, we construct protocols that exactly match the round complexity of existing protocols when relying on keyed multi-collision resistance, but shave a round in the keyless setting. From the point of view of techniques, we avoid the interaction overhead incurred in the above works by diverging from the notion of *fully binding* commitments and aiming only for (different variants of) *weakly binding commitments*. We then develop the required tools for obtaining and utilizing such weakly-binding commitments.

Finally, we observe that our results can also be used generically to achieve local opening for *fully* binding commitments, closing a gap left open in [KNY17a]. Komargodski et al. construct shrinking commitments with local opening from keyed multi-collision resistance. For strings of length λ^c their commitment requires $O(c)$ rounds. They also propose a different 4-message construction that does not support local opening. This leaves open the possibility of constructing constant round shrinking commitments with local opening for strings of arbitrary polynomial length. We show that assuming keyed multi-collision resistant hash functions, any commitment scheme can be transformed into one that supports local opening by adding at most two messages. Applied to the commitment of [KNY17a], the transformation gives a shrinking 5-message commitment with local opening for all polynomial size strings. The transformation is based on our succinct arguments for **NP** and is described in Appendix C. We emphasize that unlike the rest of this work which is independent of [KNY17a], Appendix C follows their work.

2 Preliminaries

We rely on the standard computational concepts:

- A PPT is a probabilistic polynomial-time algorithm.
- We model efficient adversaries as PPTs with arbitrary non-uniform advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$.
- We say that a function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for all constants $c > 0$, there exists $N \in \mathbb{N}$ such that for all $n > N$, $f(n) < n^{-c}$. We sometimes denote negligible functions by negl.
- We say that a function $f : \mathbb{N} \to \mathbb{R}$ is noticeable if there exists a constant $c > 0$ and $N \in \mathbb{N}$ such that for all $n > N$, $f(n) \geq n^{-c}$.

- If $\mathcal{X}^{(b)} = \{X_\lambda^{(b)}\}_{\lambda\in\mathbb{N}}$ for $b \in \{0,1\}$ are two ensembles of random variables indexed by $\lambda \in \mathbb{N}$, we say that $\mathcal{X}^{(0)}$ and $\mathcal{X}^{(1)}$ are ε-computationally indistinguishable for a function $\varepsilon(\lambda)$, if for all polynomial-size distinguishers $\mathcal{D}$, and all large enough λ,

$$\left|\Pr[\mathcal{D}(X_\lambda^{(0)}) = 1] - \Pr[\mathcal{D}(X_\lambda^{(1)}) = 1]\right| \leq \varepsilon(\lambda).$$

  We denote this by $\mathcal{X}^{(0)} \approx_\varepsilon \mathcal{X}^{(1)}$. We say that the ensembles are simply **computational indistinguishable** if they are ε-indistinguishable for every noticeable function $\varepsilon(\lambda) = \lambda^{-O(1)}$.

Fact 2.1. *Let D be a distribution, π be a predicate, f be a function on the support of D, and $t \in \mathbb{N}$ be an integer. Let $S_0 = \emptyset$, and consider a random process where for every $i \in [t]$, we sample $x_i \leftarrow D$ and if $\pi(x_i) = 1$, add $f(x_i)$ to the previous set $S_i := S_{i-1} \cup \{f(x_i)\}$. Let p be the probability that an additional sample $x_{t+1} \leftarrow D$ satisfies the predicate and is not covered by S_n, namely, $\pi(x_{t+1}) = 1$ but $f(x_{t+1}) \notin S_t$. Then,*

$$p \leq \frac{\mathbb{E}\left[|S_t|\right]}{t} \quad .$$

Proof. The fact that S_t grows with probability p given a random sample x_{t+1}, implies that for every i, the probability that S_{i-1} grows in step i is at least p. Thus, the expected number size of S_t is at least tp. $\square$

2.1 Zero-Knowledge Protocols

In what follows, we denote by $\langle \mathsf{P} \leftrightarrows \mathsf{V}\rangle$ a protocol between two parties P and V. For input w for P, and common input x, we denote by $\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}\rangle(x)$ the output of V in the protocol. For honest verifiers this output will be a single bit indicating acceptance (or rejection), whereas we assume (without loss of generality) that malicious verifiers outputs their entire view. Throughout, we assume that honest parties in all protocols are uniform PPT algorithms.

Definition 2.1 ([GMR89])**.** *A protocol $\langle \mathsf{P} \leftrightarrows \mathsf{V}\rangle$ for an **NP** relation $\mathcal{R}(x, w)$ is a zero-knowledge proof if it satisfies:*

Completeness: *For any $\lambda \in \mathbb{N}, x \in \mathcal{L}(\mathcal{R}) \cap \{0,1\}^\lambda$, $w \in \mathcal{R}(x)$:*

$$\Pr\left[\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}\rangle(x) = 1\right] = 1 \quad .$$

Soundness: *For any prover P^* there exists a negligible function μ, such that for any $x \in \{0,1\}^\lambda \setminus \mathcal{L}(\mathcal{R})$,*

$$\Pr\left[\langle \mathsf{P}^* \leftrightarrows \mathsf{V}\rangle(x) = 1\right] \leq \mu(\lambda) \quad .$$

The system is **computationally sound** *if the above is restrict to PPT provers P^* with polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$. In this case, the protocol is said to be* **an argument***.*

Zero Knowledge: *There exists a PPT simulator S such that for every polynomial-size circuit family $\mathsf{V}^* = \{\mathsf{V}^*_\lambda\}_\lambda$:*

$$\{\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}^*_\lambda(x)\rangle\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \approx_c \{\mathsf{S}(x; \mathsf{V}^*_\lambda)\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \quad .$$

3 Multi-collision-resistant Hash Functions

In this section, we define the notion of multi-collision-resistant hash functions. We start with the standard formulation of this notion, and then define a relaxed version geared toward the setting of fixed (keyless) hash functions.

Syntax: A hash function is associated with an input length function $\ell(\lambda) > \lambda$ and polynomial-time algorithms $\mathsf{H} = (\mathsf{H.Gen}, \mathsf{H.Hash})$ with the following syntax:

- $\mathsf{H.Gen}(1^\lambda)$ is a probabilistic algorithm that takes the security parameter 1^λ and outputs a key hk. In the keyless setting, this algorithm is deterministic and outputs a fixed key $\mathsf{hk} \equiv 1^\lambda$.

- $\mathsf{H.Hash}(\mathsf{hk}, X)$ is a deterministic algorithm that takes the key hk and an input $X \in \{0,1\}^{L(\lambda)}$ and outputs a hash $Y \in \{0,1\}^{\lambda}$.

Definition 3.1 (K-Collision Resistance)**.** *Let $K(\cdot)$ be a function. We say that H is K-collision-resistant if for any PPT $\mathcal{A}$ and sequence of polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$, there is a negligible function μ, such that for any $\lambda \in \mathbb{N}$, letting $K = K(\lambda)$,*

$$\Pr\left[\begin{array}{c} Y_1 = \cdots = Y_K \\ \forall i \neq j : X_i \neq X_j \end{array} \middle| \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{H.Gen}(1^\lambda) \\ (X_1, \ldots, X_K) \leftarrow \mathcal{A}(\mathsf{hk}; z_\lambda) \\ \forall i : Y_i = \mathsf{H.Hash}(\mathsf{hk}, X_i) \end{array}\right] \leq \mu(\lambda) \ .$$

Remark 3.1 (Compression). We note that for the above definition to be non-trivial it must be that there necessarily exists K-collisions, which requires that the hash function is sufficiently compressing . We will consider two types of compression. We will say that the hash function is *polynomially compressing* if the input size $L = \lambda^{1+\Omega(1)}$ is polynomially larger than the output size λ. We will say that it is *linearly compressing* if $L = \lambda \cdot (1 + \Omega(1))$. Note that in either case we are guaranteed that there exists a $2^{\Omega(\lambda)}$-collision. We call $\alpha(\lambda) = L(\lambda)/\lambda$ *the compression rate*.

The above definition of multi-collision-resistance is clearly unachievable in the keyless setting where H is a fixed hash function. Indeed, the advice z_λ may always include some K-collision in the function H. This motivates our relaxed definition, where the adversary may be able to find collisions of size proportional to its advice, but not significantly larger. That is, the number of collisions K now depends on (and could be bigger than) the size of the advice z_λ.

We note that while the definition is geared toward the keyless setting it also serves as a meaningful relaxation in the keyed setting. We accordingly formulate it in the more general keyed setting.

Definition 3.2 (Weak K-Collision Resistance)**.** *Let $K(\cdot,\cdot)$ be a function. We say that H is weakly K-collision-resistant if for any PPT $\mathcal{A}$ and any sequence of polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$, there is a negligible function μ, such that for any $\lambda \in \mathbb{N}$, letting $K = K(\lambda, |z_\lambda|)$,*

$$\Pr\left[\begin{array}{c} Y_1 = \cdots = Y_K \\ \forall i \neq j : X_i \neq X_j \end{array} \middle| \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{H.Gen}(1^\lambda) \\ (X_1, \ldots, X_K) \leftarrow \mathcal{A}(\mathsf{hk}; z_\lambda) \\ \forall i : Y_i = \mathsf{H.Hash}(\mathsf{hk}, X_i) \end{array}\right] \leq \mu(\lambda) \ .$$

Remark 3.2 (Super-Polynomial Running Time). For some of our applications, we will require a strengthening of the above two definitions that allows the adversary $\mathcal{A}$ to run in some (usually slight) super-polynomial time. Accordingly, for a function $\gamma(\lambda)$ (possibly $\gamma = \lambda^{\omega(1)}$), we will say that H is (weakly) (K,γ)-collision resistant if the guarantee of Definitions 3.1 and 3.2 holds against any probabilistic $\gamma^{O(1)}$-time $\mathcal{A}$ with arbitrary polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$.

Remark 3.3 (The Collision Size K). Naturally, we shall think of the collision size parameter K as polynomial in the security parameter λ. In some cases, however, we may want to think also of super-polynomial K. In those cases, for the above definition to be meaningful, we consider (K,γ)-collision resistance for γ that may be large than K, e.g. an arbitrary polynomial $\mathrm{poly}(K)$.

As discussed in the introduction, through the approach of inaccessible entropy [HRVW09], it is possible to capture super-polynomial collision bounds without resorting to super-polynomial running time. However, in our application, even if we start from such a definition, our reductions take a $\mathrm{poly}(K)$ loss. We thus stick to the above simple definition of K-collision resistance also for super polynomial values of K.

4 Multi-collision-resistant Hash with Local Opening

In this section, we define and construct a multi-collision-resistant hash with local opening, which is an analog of the concept of hash tress from the literature [Mer89], with a relaxed collision resistance requirement.

Syntax: A multi-collision-resistant hash with local opening is associated with polynomial-time algorithms

$$\mathsf{HLO} = (\mathsf{HLO.Gen}, \mathsf{HLO.Hash}, \mathsf{HLO.Chal}, \mathsf{HLO.Auth}, \mathsf{HLO.Ver}) \ ,$$

with the following syntax:

- $\mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda)$ is a probabilistic algorithm that takes the security parameter 1^λ and outputs a key hk. In the keyless setting, this algorithm is deterministic and outputs a fixed key $\mathsf{hk} \equiv 1^\lambda$.
- $\mathsf{dig} \leftarrow \mathsf{HLO.Hash}(\mathsf{hk}, X)$ is a deterministic algorithm that takes the key hk and an input $X \in \{0,1\}^L$ of length $L \leq 2^\lambda$. It outputs a digest $\mathsf{dig} \in \{0,1\}^\lambda$.
- $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$ is a probabilistic algorithm that takes the security parameter 1^λ and an opening-size parameter 1^ρ. It outputs a challenge ch.
- $\Pi \leftarrow \mathsf{HLO.Auth}(\mathsf{hk}, X, I, \mathsf{ch})$ is a deterministic algorithm that takes the key hk, input $X \in \{0,1\}^L$, an index set $I \subseteq [L]$ and a challenge ch. It outputs a proof Π that $X|_I = (X_i \mid i \in I)$ is consistent with the digest dig.
- $b \leftarrow \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi)$ is a deterministic algorithm that takes the key hk, an input length $L \in \mathbb{N}$, the digest $\mathsf{dig} \in \{0,1\}^\lambda$, the index set I, and an assignment $A : I \to \{0,1\}$, as well as a challenge ch and a corresponding proof Π. It outputs a bit.

Remark 4.1 (The Challenge). Differently from the standard notion of Merkle tree, where the opening phase is non-interactive and includes a single message from the opening party, in the definition considered here the opening phase consists of a random challenge from the receiver, followed by the response message. The challenge itself depends on the security parameter λ as well as an opening size ρ, which intuitively specifies the maximal number of locations that can be simultaneously opened, while guaranteeing consistency with the digest (in terms of completeness, any number of locations may be opened regardless of ρ).

In the above definition, verification is public in the sense that the challenge algorithm does not produce any private state. We may further require a *public-coin* challenge algorithm, where the challenger simply outputs its random coins. Indeed, our construction will satisfy this stronger requirement.

Definition 4.1 (Multi-collision-resistant Hash with Local Opening)**.** *A multi-collision-resistant hash with local opening* $\mathsf{HLO} = (\mathsf{HLO.Gen}, \mathsf{HLO.Hash}, \mathsf{HLO.Chal}, \mathsf{HLO.Auth}, \mathsf{HLO.Ver})$ *satisfies:*

Correctness: *The verifier accepts in any honest execution. That is, for every parameters $\lambda, \rho \in \mathbb{N}$, integer $L \leq 2^\lambda$, string $X \in \{0,1\}^L$, and set $I \subseteq [L]$,*

$$\Pr\left[\mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, X|_I, \mathsf{ch}, \Pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ \mathsf{dig} = \mathsf{HLO.Hash}(\mathsf{hk}, X) \\ \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ \Pi = \mathsf{HLO.Auth}(\mathsf{hk}, X, I, \mathsf{ch}) \end{array}\right] = 1 \enspace .$$

Succinctness: *There exists a fixed polynomial* poly*, such that the length of the proof in the above (honest) experiment is* $|\Pi| = \mathrm{poly}(\lambda, \rho, |I|)$.

K-Collision Resistance for Length Bound $\bar{L}(\lambda)$: *There exists a PPT extractor* Ext *with the following guarantee. For any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, any polynomial-size advice sequence $\{z_\lambda\}_{\lambda \in \mathbb{N}}$, any noticeable function $\varepsilon(\lambda)$, any length $L(\lambda) = \bar{L}^{O(1)}$, for all but finitely many security parameters λ and every opening size $\rho \leq L$, letting $K = K(\lambda, |z_\lambda|, L)$,*

$$\Pr\left[\begin{array}{l} |I| \leq \rho \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ A \notin \{X|_I : X \in S\} \end{array} \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ (\mathsf{dig}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ (I, A, \Pi) \leftarrow \mathcal{A}_2(\mathsf{ch}; \mathsf{st}) \\ \hline S \leftarrow \mathsf{Ext}^{\mathcal{A}_2(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{1/\varepsilon}) \end{array}\right] \leq \varepsilon \enspace .$$

Furthermore, in the above experiment Ext *always outputs a set S of size at most K.*

Remark 4.2 (The Extracted-List Size K). While it is natural to think of the size K of the extracted list as polynomial in the security parameter λ, we will also consider a weaker guarantee where K is super polynomial. We note that even when K is super polynomial we may address polynomial-size adversaries $\mathcal{A}$, and only the size of the extractor scales with K.

Remark 4.3 (The Input-Length Bound $\bar{L}$). We note that while in terms of functionality, we will always support hashing strings of any length $\leq 2^{\lambda}$, in terms of security we may be restricted to smaller values of $\bar{L}$. Indeed, in our constructions the achieved parameters will depend on $\bar{L}$, and out scheme will be able to tolerate a bound $\bar{L}$ of at most $2^{o(\lambda)} \ll 2^{\lambda}$ (assuming appropriate collision-resistance).

We now state the main theorems proved in this section regarding the existence of multi-collision-resistant hash with local opening (according to the above definition) based on weak multi-collision-resistant hash functions (Definition 3.2). Theorem 4.1 is a polynomial version that guarantees security for inputs of arbitrary polynomial length, based on polynomial assumptions. Theorem 4.2 is a super-polynomial version that guarantees security even for slightly super-polynomial input length, relying on slightly super-polynomial assumptions. This second theorem will be useful in applications where an apriori polynomial bound on the input length may not be known. Both theorems are stated for the case that the hash function is polynomially compressing. We add a third theorem that captures both in the case of linear compression.

Theorem 4.1 (Multi-collision-resistant Hash with Local Opening for Polynomial-Length Input)**.** *Assuming a polynomially-compressing weakly K-collision-resistant hash for $K(\lambda,\zeta) = \mathrm{poly}(\lambda,\zeta)$, there exists a $K^{O(1)}$-collision-resistant hash with local opening for any input-length bound $\bar{L}(\lambda) = \lambda^{O(1)}$.*

Theorem 4.2 (Multi-collision-resistant Hash with Local Opening for Super-Polynomial-Length Input)**.** *For any (arbitrarily small) $\tau(\lambda) = \omega(1)$, there exists $\bar{L}(\lambda) = \lambda^{\omega(1)}$ such that assuming a polynomially-compressing weakly (K,γ)-collision-resistant hash for $K(\lambda,\zeta) = \mathrm{poly}(\lambda,\zeta)$ and $\gamma(\lambda) = \lambda^{\tau}$, there exists a K^{τ}-collision-resistant hash with local opening for input-length bound $\bar{L}$.*

Theorem 4.3 (Linear Compression Version)**.** *For any (arbitrarily small) $\tau(\lambda) = \omega(\log\lambda)$, there exists $\bar{L}(\lambda) = \lambda^{\omega(1)}$ such that assuming a linearly-compressing weakly (K,γ)-collision-resistant hash for $K(\lambda,\zeta) = \mathrm{poly}(\lambda,\zeta)$ and $\gamma(\lambda) = \lambda^{\tau}$, there exists a K^{τ}-collision-resistant hash with local opening for input-length bound $\bar{L}$.*

Remark 4.4 (Super-Polynomial Collision Bound). In the above, we concentrate on a natural setting of parameters, where we assume the collision bound K is polynomial. We can consider a collision parameter that may be some super-polynomial function at the cost of assuming (K,γ)-collision-resistance for an appropriate $\gamma \gg K$ hardness parameter.

In the next sections, we describe the construction behind the theorems, and its building blocks. Eventually, the theorems above are derived as corollaries of a more general Theorem 4.6 proven in Sections 4.3,4.4.

4.1 Ingredient I: Hash Trees

As a building block toward the construction of hash with local opening, we define and construct a hash tree. Roughly speaking, our hash tree allows to commit to a string $X \in \{0,1\}^{\lambda \times L}$ consisting of L blocks in $\{0,1\}^{\lambda}$. Unlike multi-collision-resistant hash with local opening, the hash tree is only *locally binding* in the sense that for any single location $i \in [L]$, specifying *a block of bits* the adversary may successfully open only a small number of values K from $\{0,1\}^{\lambda}$ for this block. However, when opening many locations $I = \{i_1,\ldots,i_{|I|}\}$ simultaneously, the adversary can "mix-and-match" values. That is, the number of possible openings overall may be as large as $K^{|I|} \gg \mathrm{poly}(K)$.[14]

Syntax: A hash tree is associated with polynomial-time algorithms

$$\mathsf{HT} = (\mathsf{HT.Gen}, \mathsf{HT.Hash}, \mathsf{HT.Auth}, \mathsf{HT.Ver})$$

with the following syntax:

- $\mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^{\lambda})$: is a probabilistic algorithm that takes the security parameter 1^{λ} and outputs a key hk. In the keyless setting, the algorithm is deterministic and outputs a fixed key $\mathsf{hk} \equiv 1^{\lambda}$.

[14]Note that for such a definition, considering blocks rather than individual bits is necessary. Indeed, we consider blocks that may potentially have an exponential number of values, making the restriction to polynomial meaningful, whereas for an individual bit a restriction to any more than a single value becomes meaningless.

- $\mathsf{dig} \leftarrow \mathsf{HT.Hash}(\mathsf{hk}, X)$: is a deterministic algorithm that takes a key $\mathsf{hk} \in \{0,1\}^\lambda$ and an input $X \in \{0,1\}^{\lambda \times L}$ where $L \leq 2^\lambda$. It outputs a digest $\mathsf{dig} \in \{0,1\}^\lambda$.
- $\Pi \leftarrow \mathsf{HT.Auth}(\mathsf{hk}, X, i)$: is a deterministic algorithm that takes a key hk, an input $X \in \{0,1\}^{\lambda \times L}$ and an index $i \in [L]$. It outputs a proof Π that X_i is consistent with the digest dig.
- $b \leftarrow \mathsf{HT.Ver}(\mathsf{hk}, L, \mathsf{dig}, i, A, \Pi)$: is a deterministic algorithm that takes the key hk, an input length $L \in \mathbb{N}$, the digest $\mathsf{dig} \in \{0,1\}^\lambda$, the index i, a block assignment $A \in \{0,1\}^\lambda$, and a proof Π. It outputs a bit.

Definition 4.2 (Multi-collision-resistant Hash Tree)**.** *A multi-collision-resistant hash tree* $\mathsf{HT} = (\mathsf{HT.Gen}, \mathsf{HT.Hash}, \mathsf{HT.Auth}, \mathsf{HT.Ver})$ *satisfies:*

Correctness: *The verifier accepts in any honest execution. That is, for every security parameter $\lambda \in \mathbb{N}$, integer $L \leq 2^\lambda$, string $X \in \{0,1\}^{\lambda \times L}$, and index $i \in [L]$,*

$$\Pr\left[\mathsf{HT.Ver}(\mathsf{hk}, L, \mathsf{dig}, i, X_i, \Pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda) \\ \mathsf{dig} = \mathsf{HT.Hash}(\mathsf{hk}, X) \\ \Pi = \mathsf{HT.Auth}(\mathsf{hk}, X, i) \end{array}\right] = 1 \ .$$

Succinctness: *There exists a fixed polynomial* poly*, such that the length of the proof in the above (honest) experiment is* $|\Pi| = \mathrm{poly}(\lambda)$.

*K**-Collision Resistance for Input-Length Bound $\bar{L}(\lambda)$ and Accuracy Bound $\underline{\varepsilon}(\lambda)$:*** *There exists a PPT extractor* Ext *with the following guarantee. For any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, any polynomial-size advice sequence $\{z_\lambda\}_{\lambda \in \mathbb{N}}$, any $\varepsilon(\lambda) > \underline{\varepsilon}^{\Omega(1)}$, any length $L \leq \bar{L}^{O(1)}$, for all but finitely many security parameters λ, and for every $i \in [L]$, letting $K = K(\lambda, |z_\lambda|, L)$,*

$$\Pr\left[\begin{array}{l} \mathsf{HT.Ver}(\mathsf{hk}, L, \mathsf{dig}, i, A, \Pi) = 1 \\ A \notin S \end{array} \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda) \\ (\mathsf{dig}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ (A, \Pi) \leftarrow \mathcal{A}_2(\mathsf{st}) \\ \hline S \leftarrow \mathsf{Ext}^{\mathcal{A}_2(\mathsf{st})}(i, 1^K, 1^{1/\varepsilon}) \end{array}\right] \leq \varepsilon \ .$$

Furthermore, in the above experiment Ext *always outputs a set S of size at most K.*

Remark 4.5. In the above definition, $\mathcal{A}_2(\mathsf{st})$ has no further input. The second part of the experiment is thus defined over the coins tosses of $\mathcal{A}_2(\mathsf{st})$.

The Construction. The construction of HT:

$$\mathsf{HT} = (\mathsf{HT.Gen}, \mathsf{HT.Hash}, \mathsf{HT.Auth}, \mathsf{HT.Ver})$$

is based on a standard hash-tree with arity $\alpha = \alpha(\lambda)$ for $2 \leq \alpha(\lambda) \leq \lambda$. The basic building block is accordingly a hash function $\mathsf{H} = (\mathsf{H.Gen}, \mathsf{H.Hash})$ shrinking $\lambda \times \alpha$ bits to λ bits.

- $\mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda)$**:**
  Runs $\mathsf{H.Gen}(1^\lambda)$ and outputs the corresponding key hk.
- $\mathsf{dig} \leftarrow \mathsf{HT.Hash}(\mathsf{hk}, X)$**:**
  Constructs a hash tree as follows. Let L be the number of blocks in the input $X \in \{0,1\}^{\lambda \times L}$ and assume w.l.o.g that $L = \alpha^d$ for some $d \in \mathbb{N}$. We consider a corresponding α-array, depth-d, tree where each node is indexed by string $\sigma \in [\alpha]^i$ for $0 \leq i \leq d$, and is associated with a label $X_\sigma \in \{0,1\}^\lambda$ computed as follows:
  - For each leaf $\sigma \in \{0,1\}^d$,
    $$X_\sigma = X_\sigma \ .$$
    Namely, the label is the σ-th input block, where we naturally interpret σ as an integer in $[L] \cong [\alpha]^d$.

– For each intermediate node $\sigma \in \{0,1\}^i$, where $0 \leq i < d$,

$$X_\sigma = \mathsf{H.Hash}(\mathsf{hk}, X_{\sigma 1} \dots X_{\sigma \alpha}) \ .$$

Namely, the label corresponding to σ is the hash of the parent labels.

The output digest dig is then set to be the root label X_ε, where ε is the empty string.

- $\Pi \leftarrow \mathsf{HT.Auth}(\mathsf{hk}, X, \sigma)$**:**
interprets $\sigma \in [L]$ as a string in $[\alpha]^d$, and outputs as the proof Π all the nodes on the path from σ to the root along with their siblings:
$$\Pi = (X_{\sigma_1 \dots \sigma_i 1}, \dots, X_{\sigma_1 \dots \sigma_i \alpha} \mid 0 \leq i < d-1) \ .$$

- $b \leftarrow \mathsf{HT.Ver}(\mathsf{hk}, L, X_\varepsilon, \sigma, A, \Pi)$**:**
given a proof
$$\Pi = \left(X^*_{\sigma_1 \dots \sigma_i 1}, \dots, X^*_{\sigma_1 \dots \sigma_i \alpha} \mid 0 \leq i < d-1\right) \ ,$$
and letting $X^*_\varepsilon := X_\varepsilon$, it checks the consistency of the path:
$$X^*_{\sigma_1 \dots \sigma_i} = \mathsf{H.Hash}(\mathsf{hk}, X^*_{\sigma_1 \dots \sigma_i 1}, \dots, X^*_{\sigma_1 \dots \sigma_i \alpha}) \text{ for all } 0 \leq i < d \ ,$$
and the consistency with the given assignment:
$$X^*_{\sigma_1 \dots \sigma_d} = A \ .$$

The following proposition shows that if the underlying hash function is K-collision resistant, then the above tree is locally K^d-collision resistant, where d is the depth of the tree.

Theorem 4.4. *Let:*

- *$K(\lambda, \zeta)$ be polynomial in ζ.*
- *$\bar{L} = \bar{L}(\lambda)$ be an input-length bound and let $\underline{\varepsilon} = \underline{\varepsilon}(\lambda)$ be an accuracy bound.*
- *H be a weak (K, γ)-collision resistant hash where $\gamma(\lambda) = K^{\bar{d}}(\lambda, \lambda)/\underline{\varepsilon}$, and $\bar{d} = \log_\alpha \bar{L}$.*
- *$K'(\lambda, \zeta, L) = K^d(\lambda, \zeta)$ be a collision bound where $d = \log_\alpha L$.*

Then HT, with arity α is K'-collision-resistant with input-length bound $\bar{L}$ and accuracy bound $\underline{\varepsilon}$.

Remark 4.6 (Parameters). Throughout most of this work, we will set $\alpha = \lambda$, which will ensure that when $\bar{L}$ is polynomial so is K'. This comes at the account of assuming that the underlying hash function is polynomially compressing, i.e. maps λ^2 (or more generally $\lambda^{1+\Omega(1)}$) bits to λ bits. We can also deal with say $\alpha = 2$, which will result in weaker collision-resistance of the hash tree $K' \approx \mathrm{quasipoly}(K)$ and require stronger collision-resistance from the underlying hash (K, γ)-resistance for $\gamma \gg \mathrm{quasipoly}(K)$.

Also, the assumption above that the dependence of $K(\lambda, \zeta)$ on ζ is polynomial is for the sake of simplicity. The same statement also holds for any dependence $2^{\zeta^{o(1)}}$. (Even worst dependence can be tolerated, at the account of degrading the resulting collision resistance.)

Proof. The correctness and succinctness properties hold readily (just as in the basic construction of hash trees from the literature [Mer89]). We focus on proving multi-collision-resistance.

We describe the extractor $\mathsf{Ext}^{\mathcal{A}_2(\mathsf{st})}(i, 1^{K'}, 1^{1/\varepsilon})$ and prove that it satisfies the required guarantee.

The extractor does the following:

- **Initialize:** Creates an (initially empty) list of blocks S.
- **Sample:** Obtains from $\mathcal{A}_2(\mathsf{st})$ a total of $2K'/\varepsilon$ samples of the form (A, Π). For every such sample where Π is valid add it: $S = S \cup \{A\}$.

- **Output:** If $|S| > K'$ output `fail`, otherwise output S.

We first note that the running time of the extractor is $\mathrm{poly}(K', \varepsilon^{-1})$. Next, we prove that the extractor satisfies the local K'-collision-resistance guarantee. For this purpose, fix any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, accuracy $\varepsilon(\lambda) > \underline{\varepsilon}^{\Omega(1)}$, input length $L(\lambda) \le \bar{L}^{O(1)}$, and $\sigma \in [L] \cong [\alpha]^d$. We first bound the failure probability.

Claim 4.1. *There exists a negligible μ, such that for all $\lambda \in \mathbb{N}$*

$$\Pr\left[\text{Ext } \textit{outputs } \texttt{fail} \;\middle|\; \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda) \\ (\mathsf{dig}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ \hline S \leftarrow \mathsf{Ext}^{\mathcal{A}_2(\mathsf{st})}(\sigma, 1^K, 1^{1/\varepsilon}) \end{array} \right] \le \mu(\lambda) \ .$$

Proof. We prove this based on the (K, γ)-collision resistance of H. For this, it is enough to show that whenever the exactor fails, we can efficiently extract a K-collision in H from its transcript. Indeed, whenever the extractor fails, the list S contains at least $K' = K^d$ distinct block assignments A for σ, where each was sampled with a valid proof Π_A of consistency.

We argue that for some node $\sigma_1 \dots \sigma_j$, along the path from the root to $\sigma = \sigma_1 \dots \sigma_d$, the extractor obtains a label $X^*_{\sigma_1 \dots \sigma_j}$ together with K distinct preimages:

$$\left\{ X^*(t) := \left(X^*_{\sigma_1 \dots \sigma_j 1}(t), \dots, X^*_{\sigma_1 \dots \sigma_j \alpha}(t) \right) \;\middle|\; t \in [K] \right\}$$
$$\textit{such that}$$
$$X^*_{\sigma_1 \dots \sigma_j} = \mathsf{H.Hash}(\mathsf{hk}, X^*(1)) = \dots = \mathsf{H.Hash}(\mathsf{hk}, X^*(K)) \ .$$

Indeed, if for all exhibited label $X^*_{\sigma_1 \dots \sigma_j}$, there are fewer than K preimages, then since the depth of the tree is d, the overall number of exhibited leafs X^*_σ is smaller than K^d, in contrast to our assumption.

Thus, we can find such collisions with the same probability as that of failure. Furthermore, the time to find such a collision is proportional to the extractor's running time which is bounded by

$$\mathrm{poly}(K', \varepsilon^{-1}) \le \mathrm{poly}(\gamma) \ ,$$

since $K' \le K^d$, for $d \le \bar{d}$, $K(\lambda, |z_\lambda|) = K(\lambda, \lambda^{O(1)}) = K^{O(1)}(\lambda, \lambda)$, and $\gamma(\lambda) = K^{\bar{d}}(\lambda, \lambda)/\underline{\varepsilon}$.

The claim now follows from the (K, γ)-collision resistance of H. □

To complete the proof of the proposition, we bound the probability that the adversary answers inconsistently with the extracted list.

Claim 4.2. *For all $\lambda \in \mathbb{N}$,*

$$p := \Pr\left[\begin{array}{l} \mathsf{HT.Ver}(\mathsf{hk}, L, \mathsf{dig}, \sigma, A, \Pi) = 1 \\ A \notin S \end{array} \;\middle|\; \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda) \\ (\mathsf{dig}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ (A, \Pi) \leftarrow \mathcal{A}_2(\mathsf{st}) \\ \hline S \leftarrow \mathsf{Ext}^{\mathcal{A}_2(\mathsf{st})}(\sigma, 1^{K'}, 1^{1/\varepsilon}) \end{array} \right] \le \varepsilon/2 \ .$$

Proof. We rely on Fact 2.1. The extractor makes $t = \frac{2K'}{\varepsilon}$ samples (A, Π) to construct the set S, and adds to S the values A with an accepting proof Π. By Fact 2.1 and using the previous claim:

$$p \le \frac{\mathbb{E}[S]}{t} \le \frac{K' + \Pr\left[S > K'\right] \cdot t}{t} \le \varepsilon/2 + \Pr\left[\text{Ext outputs } \texttt{fail}\right] \le \varepsilon/2 + \mu \le \varepsilon \ ,$$

where the last inequality holds for large enough λ. □

□

This concludes the proof of the proposition.

4.2 Ingredient II: Collision-Free Code

The second building block used in the construction of hash with local opening is a new coding scheme that has a certain collision-freeness property.

The code has the following local decoding flavor. To decode a location i of the input word, we read a small set of locations D of the codeword together with a small random set of *test locations* T. The test locations are independent of i and are only used to synchronize the decoding of different locations. In addition to decoding the value of the i-th location of the input word, we also verify consistency between the values in the locations given by D and those given by T. Collision freeness says the following: for every rectangle $\mathbf{S} = S_1 \times \cdots \times S_n$ of small enough cardinality, with high probability over the choice of T, for any location i and set D used to decode the location i, there are no partial codewords C and C' that collide in the following sense:

- Both C and C' are contained in the rectangle $\mathbf{S}$.
- In both C and C', the values in locations T and D satisfy the consistency test.
- C and C' agree on the locations T, but not on D.

In other words, with high probability an assignment to the test locations, completely fixes how any location is decoded, provided that the symbols read are always taken from the rectangle $\mathbf{S}$.

We now proceed to define and construct the code.

Syntax: A collision-free code is parameterized by

- An output alphabet Σ.
- An input length $L \in \mathbb{N}$ and output length $N \in \mathbb{N}$.
- A index set size function $\Phi(\cdot)$.
- A rectangle size function $\Delta(\cdot)$.

The code is associated with polynomial-time algorithms

$$\mathsf{CFC} = (\mathsf{CFC.Code}, \mathsf{CFC.Chal}, \mathsf{CFC.TestInd}, \mathsf{CFC.DecInd}, \mathsf{CFC.Dec}, \mathsf{CFC.Test})$$

with the following syntax:

- $C \leftarrow \mathsf{CFC.Code}(X)$: is a deterministic algorithm that takes a word $X \in \{0,1\}^L$ and outputs a codeword $C \in \Sigma^N$.
- $R \leftarrow \mathsf{CFC.Chal}(1^\tau)$: is a randomize algorithm that takes a challenge length parameter 1^τ and samples a challenge R.
- $T \leftarrow \mathsf{CFC.TestInd}(R)$: is a deterministic algorithm that takes the challenge R and outputs a set of indices $T \subseteq [N]$ of size $\Phi(\tau)$.
- $D \leftarrow \mathsf{CFC.DecInd}(R, i)$: is a deterministic algorithm that takes the challenge R and an index $i \in [L]$ and outputs a set of indices $D \subseteq [N]$ of size $\Phi(\tau)$.
- $b \leftarrow \mathsf{CFC.Test}(a)$: is a deterministic algorithm that takes an assignment $a : U \to \Sigma$ where $U \subseteq [N]$, and outputs a bit.
- $b \leftarrow \mathsf{CFC.Dec}(a, i)$: is a deterministic algorithm that takes an assignment $a : D \to \Sigma$ where $D \subseteq [N]$ and an index $i \in [L]$ and outputs a bit.

Definition 4.3 (Collision-Free Code)**.** *A collision-free code* $\mathsf{CFC} = (\mathsf{CFC.Code}, \mathsf{CFC.Chal}, \mathsf{CFC.TestInd}, \mathsf{CFC.DecInd}, \mathsf{CFC.Dec}, \mathsf{CFC.Test})$ *satisfies:*

Correctness of Decoding: *For every* $\tau \in \mathbb{N}$*, every word* $X \in \{0,1\}^L$ *and every index* $i \in [L]$*,*

$$\Pr\left[\mathsf{CFC.Dec}\left(\mathsf{CFC.Code}(X)|_D, i\right) = X_i \;\middle|\; \begin{array}{c} R \leftarrow \mathsf{CFC.Chal}(1^\tau) \\ D \leftarrow \mathsf{CFC.DecInd}(R, i) \end{array}\right] = 1 \ .$$

Correctness of Testing: *For every word $X \in \{0,1\}^L$ and every set of indices $U \subseteq [N]$*

$$\mathsf{CFC.Test}\left(\mathsf{CFC.Code}(X)|_U\right) = 1 \ .$$

Collision Freeness: *For every $\tau \in \mathbb{N}$ and every set $S \subset \Sigma$ of size at most $\Delta(\tau)$,*

$$\Pr\left[\begin{array}{l|c} \exists i \in [L] \ \exists a, a' : U \to S : & \\ \quad D = \mathsf{CFC.DecInd}(R, i) & \\ \quad D \cup T \subseteq U & R \leftarrow \mathsf{CFC.Chal}(1^\tau) \\ \quad a|_T = a'|_T & T = \mathsf{CFC.TestInd}(R) \\ \quad \mathsf{CFC.Dec}(a, i) \neq \mathsf{CFC.Dec}(a', i) & \\ \quad \mathsf{CFC.Test}(a) = 1 \wedge \mathsf{CFC.Test}(a') = 1 & \end{array}\right] \leq \frac{1}{2} \ .$$

4.2.1 Construction

We prove the following theorem:

Theorem 4.5. *For any two integers $m \in \mathbb{N}, h \geq 2$ there exists a collision-free code*

$$\mathsf{CFC}_{m,h} = (\mathsf{CFC.Code}_{m,h}, \mathsf{CFC.Chal}_{m,h}, \mathsf{CFC.TestInd}_{m,h}, \mathsf{CFC.DecInd}_{m,h}, \mathsf{CFC.Dec}_{m,h}, \mathsf{CFC.Test}_{m,h}) \ .$$

With the following parameters

$$|\Sigma| \leq h^{3h} \quad , \quad L = h^m \quad , \quad N \leq m \cdot h^{3(m-1)} \quad , \quad \Phi(\tau) = m \cdot \tau^m \quad , \quad \Delta(\tau) = h^{\tau/2} \ .$$

The Code $C \leftarrow \mathsf{CFC.Code}(X)$:
For any two integers $m \in \mathbb{N}, h \geq 2$, we construct a code $\mathsf{CFC}_{m,h}$ as follows. Let $\mathbb{F}$ be a field such that $2h^2 \leq |\mathbb{F}| \leq h^3$. We set

$$\Sigma = \mathbb{F}^h \quad , \quad L = h^m \quad , \quad N = m \cdot |\mathbb{F}|^{m-1} \quad , \quad \Phi(\tau) = m \cdot \tau^m \quad , \quad \Delta(\tau) = h^{\tau/2} \ .$$

The algorithm $\mathsf{CFC.Code}_{m,h}$ takes as input a word $X \in \mathbb{F}^L$, and computes a codeword as follows:

- **Low-Degree Extension:** Let $H \subseteq \mathbb{F}$ be a subset of size h. Recalling that $L = h^m$, we identify the set of indices $[L]$ with the cube H^m. Accordingly, the input word is associated with a function $X : H^m \to \{0,1\}$.

  Let $P_X : \mathbb{F}^m \to \mathbb{F}$ be the low-degree extension of X to $\mathbb{F}^m$. That is, P_X is the m-variate polynomial of individual degree $h-1$ such that $P_X|_{H^m} = X$.

- **Restriction to Axis-Parallel Lines:** For every $v \in \mathbb{F}^m$ and $j \in [m]$ let $\gamma_v^j : \mathbb{F} \to \mathbb{F}^m$ be the line passing through v parallel to the standard basis vector e_j:

  $$\gamma_v^j(\beta) = (v_1, \dots, v_{j-1}, \beta, v_{j+1}, \dots, v_m) \ .$$

  Let $\Gamma = \left\{\gamma_v^j \mid v \in \mathbb{F}^m, j \in [m]\right\}$ be the set of all axis parallel lines. Note that each line in Γ has $\mathbb{F}$ different representations. The total size of Γ is $m \cdot |\mathbb{F}|^{m-1}$. We identify the set of indices $[N]$ with the set Γ. The code outputs the restrictions of P_X to lines in Γ.

  $$\mathsf{CFC.Code}_{m,h}(X) = (P_X(\gamma) \mid \gamma \in \Gamma) \ .$$

  Note the every restriction $P_X(\gamma)$ is a univariate polynomial of degree $h-1$ and therefore, can be described by an element of $\Sigma = \mathbb{F}^h$, for example, by listing the evaluations of the polynomial on the set H. We identify the set Σ with the set of all univariate polynomial of degree $h-1$.

We now describe the other associated algorithms:

- **$R \leftarrow \mathsf{CFC.Chal}_{m,h}(1^\tau)$:**
  takes a parameter 1^τ, samples τ independently uniform elements $\beta_1, \dots, \beta_\tau \leftarrow \mathbb{F}$ and outputs the challenge $R = \{\beta_1, \dots, \beta_\tau\}$.

- $T \leftarrow \mathsf{CFC.TestInd}_{m,h}(R)$**:**
  takes the challenge $R \subseteq \mathbb{F}$ and outputs the following set of $\Phi(\tau)$ indices in $[N] \cong \Gamma$:
$$T = \left\{ \gamma_v^j \;\middle|\; v \in R^m, j \in [m] \right\} \quad .$$
- $D \leftarrow \mathsf{CFC.DecInd}_{m,h}(R, u)$**:**
  takes the challenge $R \subseteq \mathbb{F}$ and an index $u \in H^m \cong [L]$. For $v \in R^m$, $j \in [m]$ let $\gamma_{v,u}^j$ denote the line
$$\gamma_{v,u}^j = \gamma_{v_1,\dots,v_j,u_{j+1},\dots,u_m}^j \quad .$$
  The algorithm outputs the set of $\Phi(\tau)$ indices
$$D = \left\{ \gamma_{v,u}^j \;\middle|\; v \in R^m, j \in [m] \right\} \quad .$$
- $b \leftarrow \mathsf{CFC.Dec}_{m,h}(a, u)$**:**
  The algorithm takes an assignment $a : D \to \Sigma$ and an index $u \in H^m \cong [L]$. If $\gamma_u^1 \in D$ and $a(\gamma_u^1)(u_1) \in \{0, 1\}$ the algorithm outputs $a(\gamma_u^1)(u_1)$. Otherwise, the algorithm fails.
- $b \leftarrow \mathsf{CFC.Test}_{m,h}(a)$**:**
  The algorithm takes an assignment $a : U \to \Sigma$ for $U \subseteq \Gamma$. If there exists $\gamma, \gamma' \in U$ and $\beta, \beta' \in \mathbb{F}$ such that
$$\gamma(\beta) = \gamma'(\beta') \quad \wedge \quad a(\gamma)(\beta) \neq a(\gamma')(\beta') \ ,$$
  the algorithm outputs 0, otherwise it outputs 1.

Remark 4.7 (Public Coins). The challenge algorithm $\mathsf{CFC.Chal}$ is public-coin — it simply outputs random field elements. The output of both $\mathsf{CFC.TestInd}(R)$, $\mathsf{CFC.DecInd}_{m,h}(i, R)$ is deterministically fixed given the coins R.

It is straightforward to verify that the construction satisfies the correctness properties. Next we prove collision-freeness.

Collision Freeness. To prove this, we show that whenever collision freeness is violated with respect to a small set $S \subset \Sigma$, there must exist two distinct polynomials in S that agree on all challenge points in $R \subset \mathbb{F}$:

Claim 4.3. *Fix an index $u \in H^m$, a set $S \subset \Sigma$, and challenge $R \subset \mathbb{F}$. Let*
$$T \leftarrow \mathsf{CFC.TestInd}(R) \quad , \quad D \leftarrow \mathsf{CFC.DecInd}(R, u) \ ,$$
and let $a, a' : U \to S$ be a pair of assignments such that $T \cup D \subseteq U$ and
$$a|_T = a'|_T \tag{1}$$
$$\mathsf{CFC.Dec}(a, u) \neq \mathsf{CFC.Dec}(a', u) \tag{2}$$
$$\mathsf{CFC.Test}(a) = 1 \wedge \mathsf{CFC.Test}(a') = 1 \tag{3}$$
Then there exists $\gamma \in U$ such that the univariate polynomials $a(\gamma), a'(\gamma) \in S$ are distinct and agree on R.

Before proving the claim, we show that it indeed implies collision freeness. Recall that $\mathsf{CFC.Chal}_{m,h}(1^\tau)$ samples a challenge set R that contains τ independently random elements in $\mathbb{F}$, and that Σ contains univariate polynomials of degree $h - 1$. Thus any two distinct polynomials in $S \subseteq \Sigma$ agree on R with probability at most $\left(\frac{h-1}{|\mathbb{F}|}\right)^\tau \leq (h/2)^{-\tau}$. Assuming the set S is of size at most $\Delta(\tau)$, we can take a union bound over all such pairs to deduce collision freeness:
$$\Pr\left[\begin{array}{l} \exists i \in [L] \ \exists a, a' : U \to S : \\ \quad D = \mathsf{CFC.DecInd}(R, i) \\ \quad D \cup T \subseteq U \\ \quad a|_T = a'|_T \\ \quad \mathsf{CFC.Dec}(a, i) \neq \mathsf{CFC.Dec}(a', i) \\ \quad \mathsf{CFC.Test}(a) = 1 \wedge \mathsf{CFC.Test}(a') = 1 \end{array} \;\middle|\; \begin{array}{r} R \leftarrow \mathsf{CFC.Chal}(1^\tau) \\ T = \mathsf{CFC.TestInd}(R) \end{array}\right] \leq \Delta^2(\tau) \cdot \left(\frac{h}{2}\right)^{-\tau} \leq 2^{-\tau} \leq \frac{1}{2} \quad .$$

It is left to prove Claim 4.3.

Proof of Claim 4.3. Recall that

$$T = \{\gamma_v^j\}_{v \in R^m, j \in [m]} \quad , \quad D = \left\{\gamma_{v,u}^j = \gamma_{v_1,\dots,v_j,u_{j+1},\dots,u_m}^j\right\}_{v \in R^m, j \in [m]} \ .$$

For every $v \in R^m$, we have by definition that $\gamma_{v,u}^m = \gamma_v^m$. By the fact that a and a' agree on T (Equation (1)), we deduce that $a(\gamma_{v,u}^m) = a'(\gamma_{v,u}^m)$. Since decoding a, a' at u gives different results (Equation (2)), and since $\gamma_u^1 = \gamma_{v,u}^1$ by definition, we have that $a(\gamma_{v,u}^1) \neq a'(\gamma_{v,u}^1)$. Therefore, there exists $j \in [m-1]$ and $w \in R^m$ such that

$$a(\gamma_{w,u}^j) \neq a'(\gamma_{w,u}^j) \quad \wedge \quad \forall v \in R^m : a(\gamma_{v,u}^{j+1}) = a'(\gamma_{v,u}^{j+1}) \ . \tag{4}$$

For every $\beta \in R$ let $w_\beta = \gamma_v^j(\beta) \in R^m$. Then by definition,

$$\gamma_{w,u}^j(\beta) = \gamma_{w_\beta,u}^{j+1}(u_{j+1}) \ .$$

Therefore, since the assignments a, a' are valid (Equation (3)) and combining with Equation (4), we have

$$a(\gamma_{w,u}^j)(\beta) = a(\gamma_{w_\beta,u}^{j+1})(u_{j+1}) = a'(\gamma_{w_\beta,u}^{j+1})(u_{j+1}) = a'(\gamma_{w,u}^j)(\beta) \ .$$

That is, $a(\gamma_{w,u}^j)$ and $a'(\gamma_{w,u}^j)$ are distinct and agree on every $\beta \in R$. □

4.3 Construction

We now move on to construct a multi-collision-resistant hash with local opening:

$$\mathsf{HLO} = (\mathsf{HLO.Gen}, \mathsf{HLO.Hash}, \mathsf{HLO.Chal}, \mathsf{HLO.Auth}, \mathsf{HLO.Ver}) \ .$$

We use the following two building blocks:

- A hash tree (as defined and constructed in Section 4.1):

$$\mathsf{HT} = (\mathsf{HT.Gen}, \mathsf{HT.Hash}, \mathsf{HT.Auth}, \mathsf{HT.Ver}) \ .$$

- A collision-free code (as defined and constructed in Section 4.2):

$$\mathsf{CFC}_{m,h} = (\mathsf{CFC.Code}_{m,h}, \mathsf{CFC.Chal}_{m,h}, \mathsf{CFC.TestInd}_{m,h}, \mathsf{CFC.DecInd}_{m,h}, \mathsf{CFC.Dec}_{m,h}, \mathsf{CFC.Test}_{m,h}) \ .$$

  (The parameters m, h will be set below.)

We now describe the required algorithms:

- $\mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda)$**:**
  takes the security parameter 1^λ and outputs a key for a hash tree $\mathsf{hk} \leftarrow \mathsf{HT.Gen}(1^\lambda)$. (We assume throughout that hk includes the security parameter 1^λ in the clear.)
- $\mathsf{dig} \leftarrow \mathsf{HLO.Hash}(\mathsf{hk}, X)$**:**
  takes the key hk and an input $X \in \{0,1\}^L$. It proceeds as follows
  - Let $h = \lambda^{.9}$, $m = \log_h L$. Recall that the code $\mathsf{CFC}_{m,h}$ maps words in $\{0,1\}^{h^m} = \{0,1\}^L$ to codewords in Σ^N where:
  $$|\Sigma| \leq h^{3h} = 2^{o(\lambda)} \quad , \quad N \leq m \cdot h^{3(m-1)} = o(L^3) \ .$$
  - Parse the codeword $C = \mathsf{CFC.Code}_{m,h}(X) \in \Sigma^N$ as a sequence of N blocks in $\{0,1\}^\lambda$,[15] hash the codeword and output the digest $\mathsf{dig} = \mathsf{HT.Hash}(\mathsf{hk}, C)$.

[15] Note that $|\Sigma| \leq 2^\lambda$ and can thus be efficiently represented by λ-bit strings.

- $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$**:**
  takes the security parameter 1^λ and the opening size 1^ρ. For every $j \in [\lambda \cdot \rho]$ and every $\tau \in [\bar{\tau}]$ it samples $R_{j,\tau} \leftarrow \mathsf{CFC.Chal}_{m,h}(1^\tau)$. It outputs the challenge

  $$\mathsf{ch} = \{R_{j,\tau}\}_{j\in[\lambda\cdot\rho],\tau\in[\bar{\tau}]} \quad .$$

  Here $\bar{\tau} = \bar{\tau}(\lambda)$ is a parameter that bounds the challenge length. We show how to exactly set this parameter later on.

- $\Pi \leftarrow \mathsf{HLO.Auth}(\mathsf{hk}, X, I, \mathsf{ch})$**:**
  takes the key hk, input $X \in \{0,1\}^L$, an index set $I \subseteq [L]$ and a challenge $\mathsf{ch} = \{R_{j,\tau}\}$. It proceeds as follows

  – Obtains the codeword
  $$C = \mathsf{CFC.Code}_{m,h}(X) \in \{0,1\}^{\lambda\times N} \quad .$$

  – For every $j \in [\lambda \cdot \rho]$, every $\tau \in [\bar{\tau}]$ and every $i \in I$ it obtain the index sets
  $$T_{j,\tau} = \mathsf{CFC.TestInd}_{m,h}(R_{j,\tau}) \quad , \quad D_{i,j,\tau} = \mathsf{CFC.DecInd}_{m,h}(R_{j,\tau}, i) \quad .$$

  – Let U be the set of all indices obtained
  $$U = \bigcup_{\substack{i\in I, j\in[\lambda\cdot\rho],\\ \tau\in[\bar{\tau}]}} T_{j,\tau} \cup D_{i,j,\tau} \quad .$$

  – Output a proof Π that contains the set U, the restriction $C|_U$, and proofs of consistency with the digest dig.
  $$\Pi = \left(U, C|_U, \{\Pi_u = \mathsf{HT.Auth}(\mathsf{hk}, C, u)\}_{u\in U}\right) \quad .$$

- $b \leftarrow \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi)$**:**
  takes the key hk, an input length $L \in \mathbb{N}$, the digest $\mathsf{dig} \in \{0,1\}^\lambda$, the index set I, and an assignment $A : I \to \{0,1\}$, as well as a challenge $\mathsf{ch} = \{R_{j,\tau}\}$ and a corresponding proof $\Pi = \left(U, C|_U, \{\Pi_u\}_{u\in U}\right)$. It accepts if the following conditions are satisfied:

  1. *Hash tree consistency:* for every $u \in U$: $\mathsf{HT.Ver}(\mathsf{hk}, N, \mathsf{dig}, u, C|_u, \Pi_u) = 1$.
  2. *Code consistency:* for every $R_{j,\tau} \in \mathsf{ch}$ and every $i \in I$:
     (a) Let $T = \mathsf{CFC.TestInd}_{m,h}(R_{j,\tau})$ and $D = \mathsf{CFC.DecInd}_{m,h}(R_{j,\tau}, i)$
     (b) $T \cup D \subseteq U$.
     (c) $\mathsf{CFC.Test}_{m,h}(C|_{T\cup D}) = 1$.
     (d) $\mathsf{CFC.Dec}_{m,h}(C|_D, i) = A(i)$.

In the next sections, we prove the following theorem:

Theorem 4.6 (From Hash Tree to Hash with Local Opening)**.** *Let:*

- $\bar{L} = \bar{L}(\lambda)$ *be an input-length bound.*
- HT *be K-collision resistant with input-length bound $\bar{L}$ and accuracy bound $\underline{\varepsilon}(\lambda) = 1/\bar{L}$.*
- $K'(\lambda, \zeta, L) \geq \lambda^{(\ell+k)^{5\ell}}$ *be a collision bound where $\ell = \log_\lambda L$ and $k = \log_\lambda K(\lambda, \zeta, L)$.*
- $\bar{\tau}(\lambda) = 7(\bar{\ell} + \bar{k})$ *be a bound on the challenge length, where $\bar{\ell} = \log_\lambda \bar{L}$ and $\bar{k} = \log_\lambda K(\lambda, \lambda^{\omega(1)}, \bar{L})$.*
- *Assume $(\bar{\ell} + \bar{k})^{\bar{\ell}} \leq \lambda$.*

Then HLO *with challenge length bound $\bar{\tau}$ is K'-collision resistance with length bound $\bar{L}$.*

Remark 4.8 (Parameters). In the definition of $\bar{k}$, the function $\omega(1)$ is an arbitrarily small super-constant, so that throughout $\bar{k} = \log_\lambda K(\lambda, \lambda^{\omega(1)}, \bar{L})$ is a bound on $k = \log_\lambda K(\lambda, \zeta, L)$ for any polynomial advice-size $\zeta(\lambda) = \lambda^{O(1)}$ and length $L(\lambda) \leq \bar{L}(\lambda)$.

Combining Theorem 4.6 and Theorem 4.4 (regarding the existence of hash trees based on multi-collision resistant hash functions), we obtain:

Corollary 4.1 (From Weak Multi-collision-resistant Hash to Hash with Local Opening). *Let:*

- $\bar{L} = \bar{L}(\lambda)$ *be a bound on the input length.*
- $K(\lambda, \zeta)$ *be a collision bound, polynomial in* ζ.
- H *be a weakly* (K, γ)*-collision-resistant hash with compression rate* $\alpha(\lambda)$ *where* $\gamma(\lambda) = \lambda^{\bar{k} \cdot \bar{\ell}_\alpha}$ *for* $\bar{\ell}_\alpha = \log_\alpha \bar{L}$ *and* $\bar{k} = \log_\lambda K(\lambda, \lambda)$.
- $K''(\lambda, \zeta, L) \geq \lambda^{(\ell_\alpha k)^{10\ell}}$ *be a collision bound where* $\ell = \log_\lambda L$, $\ell_\alpha = \log_\alpha L$ *and* $k = \log_\lambda K(\lambda, \zeta)$.
- *Assume* $\left(\bar{k}\bar{\ell}_\alpha\right)^{2\bar{\ell}} \leq \lambda$ *where* $\bar{\ell} = \log_\lambda \bar{L}$ *and* $\bar{k}, \bar{\ell}$ *are defined as before.*

Then there exists a multi-collision-resistant hash with local opening that is K''*-collision resistant with length bound* $\bar{L}$.

Proof. By Theorem 4.4, given a multi-collision resistant H as above, there exists a hash tree HT of arity α that is K'-collision resistant for collision bound $K'(\lambda, \zeta, L) = \lambda^{k \cdot \ell_\alpha}$, input-length bound $\bar{L}$, and accuracy bound $1/\bar{L}$.

Letting $k' = \log_\lambda K'(\lambda, \zeta, L) = k \cdot \ell_\alpha$ and $\ell' = \ell$, and noting that $(\ell_\alpha k)^{2\ell} \geq (\ell' + k')^{\ell'}$, it follows from Theorem 4.6 that there exists a K''-collision resistant hash with local opening for $K''(\lambda, \zeta, L) \geq \lambda^{(\ell_\alpha k)^{10\ell}} \geq \lambda^{(\ell' + k')^{5\ell'}}$. $\square$

4.4 Proof of Theorem 4.6

In this section, we prove Theorem 4.6. We start by noting that the construction has the required correctness and succinctness, and then move on to the main part of the proof which is demonstrating and analysing an extractor.

The correctness of the construction follows readily from that of the hash tree and the collision-free code. We now note that the construction has the required succinctness.

Succinctness. We need to show that the proof size is $|\Pi| = \mathrm{poly}(\lambda, \rho, |I|)$. Recall that proof is of the form

$$\Pi = \left(U, C|_U, \{\Pi_u = \mathsf{HT.Auth}(\mathsf{hk}, C, u)\}_{u \in U}\right) \quad ,$$

where

$$U = \bigcup_{\substack{i \in I, j \in [\lambda \cdot \rho], \\ \tau \in [\bar{\tau}]}} U_{i,j,\tau} \qquad \text{and} \qquad U_{i,j,\tau} = T_{j,\tau} \cup D_{i,j,\tau} \quad .$$

Fix any i, j, τ. Then

$$|U_{i,j,\tau}, C|_{U_{i,j,\tau}}| \leq O(\Phi(\tau) \cdot \log|\Sigma|) \leq O(m \cdot \bar{\tau}^m \cdot \lambda) \leq \frac{10\bar{\ell}}{9} \cdot (7\bar{\ell} + 7\bar{k})^{\frac{10\bar{\ell}}{9}} \cdot O(\lambda) \leq (\bar{\ell} + \bar{k})^{3\bar{\ell}} \cdot O(\lambda)$$

and

$$|\{\mathsf{HT.Auth}(\mathsf{hk}, C, u)\}_{u \in U_{i,j,\tau}}| = |C|_{U_{i,j,\tau}}| \cdot \mathrm{poly}(\lambda) \leq (\bar{\ell} + \bar{k})^{3\bar{\ell}} \cdot \mathrm{poly}(\lambda) \quad .$$

So overall the size of the proof is

$$|I| \cdot \lambda \cdot \rho \cdot \bar{\tau} \cdot |U_{i,j,\tau}, C|_{U_{i,j,\tau}}, |\{\mathsf{HT.Auth}(\mathsf{hk}, C, u)\}_{u \in U_{i,j,\tau}}| \leq \mathrm{poly}(\lambda, |I|, \rho) \cdot (\bar{\ell} + \bar{k})^{4\bar{\ell}} \leq \mathrm{poly}(\lambda, |I|, \rho) \cdot \lambda^4 \quad .$$

4.4.1 The Extractor

We now demonstrate an appropriate extractor as per Definition 4.1.

The extractor $\mathsf{Ext}^{\mathcal{A}_2(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^{K'}, 1^{1/\varepsilon'})$ first derives the following parameters:

- N - the output length of the code.
- $\ell = \log_\lambda L$.
- K - the collision resistance parameter of the underlying hash tree HT.
- $k = \log_\lambda K$.

Step 1: Extracting a Rectangle. In the first step, the extractor obtains a (small) rectangle $\mathbf{S}^{\mathsf{HT}} = S_1^{\mathsf{HT}} \times \cdots \times S_N^{\mathsf{HT}}$ such that the answers of $\mathcal{A}_2$ are contained in $\mathbf{S}^{\mathsf{HT}}$ with high probability. (This part relies on the hash tree.)

Concretely, for every $i \in [N]$, the extractor first constructs an adversary $\mathcal{A}_{2,i}$ that produces openings for the i-th block of the hash tree. $\mathcal{A}_{2,i}$, given oracle access to $\mathcal{A}_2(\cdot;\mathsf{st})$, samples a challenge $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$, queries $\mathcal{A}_2(\cdot;\mathsf{st})$ with ch, and obtains an opening (I, A, Π). If the proof Π is not of the correct form $(U, C|_U, \{\Pi_u\})$, or if $i \notin U$, $\mathcal{A}_{2,i}$ fails and outputs $\perp$. Otherwise, $\mathcal{A}_{2,i}$ outputs $(C|_i, \Pi_i)$.

Let $\mathsf{HT.Ext}$ be the extractor for the hash tree HT guaranteed by Definition 4.2. Ext invokes:

$$S_i^{\mathsf{HT}} \leftarrow \mathsf{HT.Ext}^{\mathcal{A}_{2,i}}(i, 1^K, 1^\varepsilon) \ ,$$

with the adversary $\mathcal{A}_{2,i}$, collision bound K, and accuracy parameter $\varepsilon = \varepsilon'/2N$. It obtains a set $S_i^{\mathsf{HT}} \subseteq \Sigma$ of size at most K. The extracted rectangle is the product set

$$\mathbf{S}^{\mathsf{HT}} = S_1^{\mathsf{HT}} \times \cdots \times S_N^{\mathsf{HT}} \subseteq \Sigma^N \ .$$

In what follows, we say that an assignment $a : U \to \Sigma$ is consistent with $\mathbf{S}^{\mathsf{HT}}$, if for all $i \in U$, $a(i) \in S_i$.

Step 2: Extracting the List of Words. In the second step, given the rectangle $\mathbf{S}^{\mathsf{HT}}$, the extractor obtains a corresponding list of (global) words satisfying Definition 4.1. (This part relies on the collision-free code.)

For this purpose, we define a procedure $\mathsf{CFC.Ext}(R, \mathbf{S}^{\mathsf{HT}})$ that given a challenge R sampled by $\mathsf{CFC.Chal}_{m,h}$ and the rectangle $\mathbf{S}^{\mathsf{HT}}$, outputs a set of words $S_R \subseteq \{0,1\}^L$. Intuitively, this procedure considers assignments $a : U \to \Sigma$ that are consistent with the rectangle $\mathbf{S}^{\mathsf{HT}}$, and attempts to extend them to (partial) words, by decoding. These words are then added to S_R.

$\mathsf{CFC.Ext}(R, \mathbf{S}^{\mathsf{HT}})$ proceeds as follows:

- Let $S_R = \emptyset$ and let $T = \mathsf{CFC.TestInd}_{m,h}(R)$.
- For every assignment $a : T \to \Sigma$ that is consistent with $\mathbf{S}^{\mathsf{HT}}$:
  - For every $i \in [L]$, let $D = \mathsf{CFC.DecInd}_{m,h}(R, i)$.
    Mark the index i as consistent with the bit $b \in \{0,1\}$ if
    * There exists an assignment $a' : (T \cup D) \to \Sigma$ that is consistent with $\mathbf{S}^{\mathsf{HT}}$,
    * $a'|_T = a$,
    * $\mathsf{CFC.Test}_{m,h}(a') = 1$,
    * $\mathsf{CFC.Dec}_{m,h}(a'|_D, i) = b$.
  - For every $i \in [L]$ that has not been marked as consistent with either 0 or 1, mark i as consistent with 0 (here 0 is an arbitrary choice).
  - If for every $i \in [L]$, there exists a bit $b_i \in \{0,1\}$ such that i is marked as consistent with b_i, but not with $1 - b_i$, then add the word $b_1 \dots b_L$ to S_R.
  - Otherwise (there exists $i \in [L]$ that is marked as consistent with both 0 and 1), mark that the challenge R has a collision.

- If R has a collision, output $\perp$. Otherwise, output the set S_R.

The extractor Ext now proceeds as follows:

- Let $\tau = 7(\ell + k)$.
- For every $j \in [\lambda \cdot \rho]$, sample a challenge R_j and extract a corresponding set of words:

$$R_j \leftarrow \mathsf{CFC.Chal}(1^\tau) \quad , \quad S_{R_j} = \mathsf{CFC.Ext}(R_j, \mathbf{S}^{\mathsf{HT}}) \ .$$

- Let J be the set of indices j such that the challenge R_j did not have a collision:

$$J = \left\{ j \in [\lambda \cdot \rho] : S_{R_j} \neq \perp \right\} \ .$$

- Output the union set $S = \bigcup_{j \in J} S_{R_j}$.

4.4.2 Analysis

In this section we prove that Ext satisfies Definition 4.1.

Bounding the Output Size and Running Time of the Extractor. We first show that Ext outputs a set S of size at most $K' := \lambda^{(\ell+k)^{5\ell}}$. We assume that $\ell \geq 2$. Consider the execution of any $\mathsf{CFC.Ext}(R_j, \mathbf{S}^{\mathsf{HT}})$. By the properties of the code $\mathsf{CFC}_{m,h}$, we have that for $T = \mathsf{CFC.TestInd}_{m,h}(R)$:

$$|T| \leq \Phi(\tau) = m \cdot \tau^m \leq \frac{10\ell}{9} \cdot (7\ell + 7k)^{\frac{10\ell}{9}} \leq (\ell + k)^{3\ell} \ .$$

Since the sets $S_1^{\mathsf{HT}}, \dots, S_N^{\mathsf{HT}}$ are each of size at most K, the number of assignments $a : T \to \Sigma$ that are consistent with $\mathbf{S}^{\mathsf{HT}}$ is at most

$$K^{|T|} \leq \left(\lambda^k\right)^{(\ell+k)^{3\ell}} \leq \lambda^{(\ell+k)^{4\ell}} \ . \tag{5}$$

This is also a bound on the output set S_{R_j}, since for every such assignment, CFC.Ext adds at most one word to S_{R_j}. Ext outputs the union of at most $\lambda \cdot \rho \leq \lambda \cdot L = \lambda^{\ell+1}$ such sets, and therefore

$$|S| \leq \lambda^{\ell+1} \cdot \lambda^{(\ell+k)^{4\ell}} \leq \lambda^{(\ell+k)^{5\ell}} = K' \ .$$

Finally, by inspection, one can see that the running time of the extractor is polynomial in the size of $|S|$ and its other inputs. Having already established that $|S| \leq K'$, it follows that the extractor runs in polynomial time in all of its inputs.

The Extractor is Successful. Fix a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, a polynomial-size advice sequence $\{z_\lambda\}_{\lambda \in \mathbb{N}}$, a noticeable function $\varepsilon' = \varepsilon'(\lambda)$, a length $L \leq \bar{L}^{O(1)}$, a large enough security parameter λ, and an opening size $\rho \leq L$.

The Experiment $\mathcal{E}$. We consider the following randomized experiment $\mathcal{E}$:

$$\begin{array}{l} \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ (\mathsf{dig}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ (I, A, \Pi) \leftarrow \mathcal{A}_2(\mathsf{ch}; \mathsf{st}) \\ \hline S \leftarrow \mathsf{Ext}^{\mathcal{A}_2(\cdot; \mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^{K'}, 1^{1/\varepsilon'}) \end{array} \ .$$

We need to show that the adversary answers consistently with the extracted set:

$$\Pr_{\mathcal{E}} \left[\begin{array}{l} |I| \leq \rho \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ A \notin \{X|_I : X \in S\} \end{array} \right] \leq \varepsilon' \ . \tag{6}$$

In what follows:

- $\Pi = (U, C|_U, \{\Pi_u\})$ is the proof output by the adversary.
- $\mathbf{S}^{\mathsf{HT}} = S_1^{\mathsf{HT}} \times \cdots \times S_N^{\mathsf{HT}}$ is the extracted rectangle in the first step of the extraction.
- $S = \bigcup_{j \in J} S_j$ is the output of the extractor Ext.

The Adversary Respects the Rectangle $\mathbf{S}^{\mathsf{HT}}$. For every $i \in [N]$, invoking the K-collision resistance of the hash tree HT for the adversary the adversary $(\mathcal{A}_1, \mathcal{A}_{2,i}{}^{\mathcal{A}_2})$, we have that

$$\Pr_{\mathcal{E}}\left[\begin{array}{l} i \in U \\ \mathsf{HT.Ver}(\mathsf{hk}, N, \mathsf{dig}, i, C|_i, \Pi_i) = 1 \\ C|_i \notin S_i^{\mathsf{HT}} \end{array}\right] \leq \varepsilon = \frac{\varepsilon'}{2N} \quad .$$

Note that $N = o(L^3) \leq \bar{L}^{O(1)}$ and $\varepsilon \geq \bar{L}^{-O(1)} = \underline{\varepsilon}^{\Omega(1)}$ as required to guarantee the above extraction.

Since the verifier $\mathsf{HLO.Ver}$ checks hash tree consistency (item 1 in the definition of $\mathsf{HLO.Ver}$), and taking a union bound, we have that

$$\Pr_{\mathcal{E}}\left[\begin{array}{l} \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ C|_U \text{ is not consistent with } \mathbf{S}^{\mathsf{HT}} \end{array}\right] \leq \frac{\varepsilon'}{2} \quad .$$

Therefore, to establish Equation (6), and deduce successful extraction, it suffices to show:

$$\Pr_{\mathcal{E}}\left[\begin{array}{l} |I| \leq \rho \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ C|_U \text{ is consistent with } \mathbf{S}^{\mathsf{HT}} \\ A \notin \{X|_I : X \in S\} \end{array}\right] \leq \frac{\varepsilon'}{2} \quad . \tag{7}$$

Fixing the Set I. The number of sets $I \subseteq [L]$ such that $|I| \leq \rho$ is at most L^ρ. Therefore, to prove Equation (7) it suffices to show that for every fixed set $I \subseteq [L]$

$$\Pr_{\mathcal{E}}\left[\begin{array}{l} \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ C|_U \text{ is consistent with } \mathbf{S}^{\mathsf{HT}} \\ A \notin \{X|_I : X \in S\} \end{array}\right] \leq \frac{\varepsilon'}{2} \cdot L^{-\rho} \quad . \tag{8}$$

For the rest of the proof, fix the set I.

Consistency of Assignment A with Extraction Relative to Challenge ch. We now show that had we performed extraction relative to the (code) challenges $\{R\}$ given by the challenge ch, then the adversary's assignment A would be consistent with the extracted set. We will then prove that except with small probability if the assignment is consistent with this (hypothetical) extracted set, it would also be consistent with the real extracted set (computed independently of the specific challenge ch.

In what follows:

- Let $\mathsf{ch} = \{R_{j,\tau}\}_{j \in [\lambda \cdot \rho], \tau \in [\bar{\tau}]}$ be the challenge in the experiment $\mathcal{E}$.
- Let $A : I \to \{0,1\}$ be the assignment and let $\Pi = (U, C|_U, \{\Pi_u\})$ be the proof, both produced by the adversary in $\mathcal{E}$.
- Fix $\tau = 7(\ell + k)$ (as fixed by the extractor Ext), and note that $\tau = 7(\ell + k) \leq 7(\bar{\ell} + \bar{k}) = \bar{\tau}$.
- For $j \in [\lambda \cdot \rho]$, let $S_j = \mathsf{CFC.Ext}(R_{j,\tau}, \mathbf{S}^{\mathsf{HT}})$ be the set of extracted words for the challenge $R_{j,\tau}$.
- Let $J = \{j \in [\lambda \cdot \rho] : S_j \neq \bot\}$ be the set of indices j for which there is no collision.
- For every set S_j denote by $S_j|_I = \{X|_I : X \in S_j\}$ the same set of words when restricted to the locations I.

(Note that the extractor Ext in the experiment $\mathcal{E}$ does not necessarily invoke $\mathsf{CFC.Ext}(R_{j,\tau}, \mathbf{S}^{\mathsf{HT}})$. This is only for the sake of the analysis.)

Claim 4.4.

$$\Pr_{\mathcal{E}}\left[\begin{array}{l}\mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, A, \mathsf{ch}, \Pi) = 1 \\ C|_U \textit{ is consistent with } \mathbf{S}^{\mathsf{HT}} \\ \exists j \in J : A \notin S_j|_I\end{array}\right] = 0 \enspace .$$

Proof. Assume toward contradiction that the above condition is violated for some $j \in J$. Let $T = \mathsf{CFC.TestInd}_{m,h}(R_{j,\tau})$. Since the verifier HLO.Ver checks code consistency (item 2 in the definition of HLO.Ver) we have that $T \subseteq U$. Let $a = C|_T : T \to \Sigma$ and note that a is consistent with S^{HT}, since $C|_U$ is.

In the execution of $\mathsf{CFC.Ext}(R_{j,\tau}, \mathbf{S}^{\mathsf{HT}})$, consider the iteration corresponding to the assignment a. Recall that $j \in J$, and thus $R_{j,\tau}$ has no collisions imply that for every $i \in [L]$, there exists a bit $b_i \in \{0,1\}$ such that i is marked consistent with b_i but not with $1 - b_i$. The word $b_1 \dots b_L$ is then added to the output set S_j. Since $A \notin S_j|_I$ (by the assumption toward contradiction), there exists $i \in I$ such that $A(i) \neq b_i$. Let $D = \mathsf{CFC.DecInd}_{m,h}(R, i)$. Again by the fact that HLO.Ver check code consistency (item 2 in its definition) we have that $D \subseteq U$ and

$$\mathsf{CFC.Test}_{m,h}(C|_{T \cup D}) = 1 \quad , \quad \mathsf{CFC.Dec}_{m,h}(C|_D, i) = A(i) \enspace ,$$

contradicting the fact that i was not marked as consistent with $A(i)$. □

We now show that, with high probability, the same consistency holds with respect to the actual extracted list.

In what follows:

- Let $R'_1, \dots, R'_{\lambda \cdot \rho}$ be the challenges sampled by the extractor in the experiment $\mathcal{E}$.
- Let $S'_1, \dots, S'_{\lambda \cdot \rho}$ be the corresponding extracted sets of words.
- Let J' is the set of indices j such that the challenge R'_j did not have a collision.

Then by Claim 4.4, to prove Equation (8) and conclude the proof, it suffices to show that

$$\Pr_{\mathcal{E}}\left[\begin{array}{l}\forall j \in J : A \in S_j|_I \\ \forall j \in J' : A \notin S'_j|_I\end{array}\right] \leq \frac{\varepsilon'}{2} \cdot L^{-\rho} \enspace . \tag{9}$$

The Sets J and J' Are Large. Toward establishing the latter, we first bound from below the size of the sets J and J':

Claim 4.5.

$$\Pr_{\mathcal{E}}\left[\ |J| \geq \tfrac{\lambda \cdot \rho}{4} \quad \bigwedge \quad |J'| \geq \tfrac{\lambda \cdot \rho}{4}\ \right] \geq 1 - 2^{-\Omega(\lambda \cdot \rho)} \enspace .$$

Proof. Any assignment $a : U \to \Sigma$ that is consistent with $\mathbf{S}^{\mathsf{HT}}$ is supported on the set $\bigcup_{i \in [N]} S_i^{\mathsf{HT}}$ which is of size at most

$$N \cdot K \leq L^3 \cdot K = \lambda^{3\ell + k} \leq \left(\lambda^{.9}\right)^{7(\ell+k)/2} = h^{\tau/2} = \Delta(\tau) \enspace .$$

Therefore, for any $j \in [\lambda \cdot \rho]$, by the collision freeness property of the code $\mathsf{CFC}_{m,h}$

$$\Pr_{\mathcal{E}}[j \notin J] = \Pr_{\mathcal{E}}\left[\begin{array}{l}\exists i \in [L] \ \exists a, a' : U \to \Sigma : \\ \quad a, a' \text{ are consistent with } S^{\mathsf{HT}} \\ \quad T = \mathsf{CFC.TestInd}(R_{j,\tau}) \\ \quad D = \mathsf{CFC.DecInd}(R_{j,\tau}, i) \\ \quad D \cup T \subseteq U \\ \quad a|_T = a'|_T \\ \quad \mathsf{CFC.Dec}(a, i) \neq \mathsf{CFC.Dec}(a', i) \\ \quad \mathsf{CFC.Test}(a) = 1 \wedge \mathsf{CFC.Test}(a') = 1\end{array}\right] \leq \frac{1}{2} \enspace .$$

Similarly, $\Pr_{\mathcal{E}}[j \notin J'] \leq 1/2$. Since the random variables $R_{1,\tau} \ \dots, R_{\lambda \cdot \rho, \tau}$ and $R'_1 \ \dots, R'_{\lambda \cdot \rho}$ are all independent the claim follows by a Chernoff bound. □

To prove Equation (9), we rely on the following basic fact.

Fact 4.1. *Let $X_1 \dots, X_n$ and $X'_1 \dots, X'_{n'}$ be independent and identically distributed random variables, where each one represents a subset from a universe U, and let $m = \min(n, n')$. Then*

$$\Pr\left[\exists x \forall j \in [m] : x \in X_j \wedge x \notin X'_j\right] \leq |U| \cdot 2^{-2m} \quad .$$

Proof. Fix any $x \in U$, and let $p_x := \Pr[x \in X_1]$. Then

$$\Pr\left[\forall j \in [m] : x \in X_j \wedge x \notin X'_j\right] = (p_x(1-p_x))^m \leq 2^{-2m} \quad .$$

The statement follows by a union bound over all $x \in U$. □

Now, consider the random variables

$$\{S_j|_I : j \in J\} \quad , \quad \{S'_j|_I : j \in J'\} \quad .$$

For any $n, n' \in [\lambda \cdot \rho]$, conditioned on $|J| = n, |J'| = n'$, the above random variables are independent and idenitically distributed as follows

$$\left\{ S|_I \;\middle|\; \begin{array}{c} R \leftarrow \mathsf{CFC.Chal}(1^\tau) \\ S = \mathsf{CFC.Ext}(R, S^{\mathsf{HT}}) \\ S \neq \bot \end{array} \right\} \quad .$$

Furthermore, these sets consist of elements from the universe $U = 2^I$, which is of size at most 2^ρ. Combining the above with Claim 4.5, we deduce:

$$\begin{aligned}
&\Pr_{\mathcal{E}}\left[\begin{array}{l} \forall j \in J : A \in S_j|_I \\ \forall j \in J' : A \notin S'_j|_I \end{array}\right] \leq \\
&\Pr_{\mathcal{E}}\left[\begin{array}{l} \forall j \in J : A \in S_j|_I \\ \forall j \in J' : A \notin S'_j|_I \end{array} \;\middle|\; \min\left(|J|, |J'|\right) \geq \frac{\lambda \cdot \rho}{4}\right] + \Pr_{\mathcal{E}}\left[\min\left(|J|, |J'|\right) < \frac{\lambda \cdot \rho}{4}\right] \leq \\
&2^\rho \cdot 2^{-\lambda \cdot \rho/4} + 2^{-\Omega(\lambda \cdot \rho)} \leq \\
&2^{-\Omega(\lambda \cdot \rho)} \leq \lambda^{-\omega(1)} \cdot 2^{-\ell\rho} \leq \frac{\varepsilon'}{2} \cdot L^{-\rho} \quad ,
\end{aligned}$$

where the last inequality holds since $\ell \leq \bar{\ell} = o(\lambda)$ (recall that $\bar{\ell}^{\bar{\ell}} \leq \lambda$).

This complete the analysis of the extraction procedure.

5 3-Message Succinct Arguments for NP

In this section, we show how to use multi-collision-resistant hash functions with local opening, to construct succinct argument systems for non-deterministic computations, and in particular for **NP**. In Section 5.1, we recall the definition of such argument systems. In Section 5.2, we recall probabilistically-checkable proofs, which are used in the construction. The construction itself is described and analyzed in Section 5.3.

5.1 Succinct Arguments for Non-Deterministic Computations

We recall the definition of succinct arguments for non-deterministic computations [Kil92, BG08].

The Universal Relation. The universal relation $\mathcal{R}_{\mathcal{U}}$ consists of pairs (u, w) where $u = (M, x, t)$, and M is a description of a Turing machine such that $M(x, w)$ accepts within t steps.

Definition 5.1 (Succinct Arguments)**.** *A succinct argument system for the universal relation $\mathcal{R}_{\mathcal{U}}$ is given by the pair of interactive PPT algorithms (P, V) satisfying the following requirements:*

Efficient Verifier and Relatively-Efficient Prover: *There exists a universal polynomial $p(\cdot)$ such that for every $u = (M, x, t) \in \{0,1\}^\lambda \cap \mathcal{L}(\mathcal{R}_{\mathcal{U}})$, where $t \leq 2^\lambda$, and every $w \in \mathcal{R}_{\mathcal{U}}(u)$:*

- *The prover* $\mathsf{P}(u, w)$ *runs in time* $p(\lambda, t)$.
- *The verifier* $\mathsf{V}(u)$ *runs in time* $p(\lambda)$.

Completeness: *For every* $u = (M, x, t) \in \{0,1\}^\lambda \cap \mathcal{L}(\mathcal{R}_\mathcal{U})$*, where* $t \leq 2^\lambda$*, and every* $w \in \mathcal{R}_\mathcal{U}(u)$*:*

$$\Pr\left[\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}\rangle(x) = 1\right] = 1 \ .$$

Proof of Knowledge for Computation-Time Bound $\bar{t}(\lambda)$ ***with*** $\tau(\lambda)$***-Time Extractor:*** *There exists an extractor* E *such that for any noticeable function* $\varepsilon(\lambda) = \lambda^{-O(1)}$*, any PPT prover* P^* *with polynomial-size advice* $\{z_\lambda\}_\lambda$*, any* $t(\lambda) \leq \bar{t}^{O(1)}$*, any large enough* $\lambda \in \mathbb{N}$*, and any* (M, x) *such that* $u = (M, x, t) \in \{0,1\}^\lambda$*,*

$$\text{if} \qquad \Pr\left[\langle \mathsf{P}^* \leftrightarrows \mathsf{V}\rangle(u)\right] \geq \varepsilon, \qquad \text{then} \qquad \Pr\left[\mathcal{R}_\mathcal{U}(u) \ni w \leftarrow \mathsf{E}^{\mathsf{P}^*}(u; 1^{1/\varepsilon}, |z_\lambda|)\right] \geq 1 - 2^{-\Omega(\lambda)} \ .$$

The extractor runs in time $\tau(\lambda, t, \varepsilon^{-1}, |z_\lambda|)$.

We now state the main theorem proved in this section regarding the existence of succinct arguments (according to the above definition) based on weak multi-collision-resistant hash functions (Definition 3.2). The theorem has three parts. The first is a polynomial version that guarantees security for computations of arbitrary polynomial time, based on polynomial assumptions. The second is a super-polynomial version that guarantees security even for slightly super-polynomial computations, relying on slightly super-polynomial assumptions. The first two are parts are for the case that the hash function is polynomially compressing, the third part addresses linear compression.

Theorem 5.1 (Succinct Arguments for Non-Deterministic Computations)**.**

- *Assuming a polynomially-compressing weakly* K*-collision-resistant hash for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$*, there exist succinct arguments for any computation-time bound* $\bar{t}(\lambda) = \lambda^{O(1)}$*, with polynomial extraction time* $\tau(\lambda) = \lambda^{O(1)}$.
- *For any (arbitrary small)* $\tau(\lambda) = \omega(1)$*, there exists* $t(\lambda) = \lambda^{\omega(1)}$ *such that assuming a polynomially-compressing weakly* (K, γ)*-collision-resistant hash for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$ *and* $\gamma(\lambda) = \lambda^\tau$*, there exist succinct arguments for computation-time bound* $\bar{t}$*, with extraction time* $\tau(\lambda) = \gamma^{O(1)}$.
- *For any (arbitrary small)* $\tau(\lambda) = \omega(\log \lambda)$*, there exists* $t(\lambda) = \lambda^{\omega(1)}$ *such that assuming a linearly-compressing weakly* (K, γ)*-collision-resistant hash for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$ *and* $\gamma(\lambda) = \lambda^\tau$*, there exist succinct arguments for computation-time bound* $\bar{t}$*, with extraction time* $\tau(\lambda) = \gamma^{O(1)}$.

If the underlying hash functions are keyless, the argument has 3 *messages (and* 4 *otherwise).*

In the next sections, we describe the construction behind the theorem, and its building blocks. Eventually, the theorem is derived as a corollary of a more general Theorem 5.2 proven in Section 5.3.

5.2 Probabilistically-Checkable Proofs

A (verifier-efficient) *probabilistically checkable proof* (PCP) for the universal relation $\mathcal{R}_\mathcal{U}$ consists of polynomial-time algorithms $(\mathsf{PCP.P}, \mathsf{PCP.V})$ with the following syntax:

- $\pi \leftarrow \mathsf{PCP.P}(u, w)$: takes a pair $(u, w) \in \mathcal{R}_\mathcal{U}$ and outputs a proof string π.
- $b \leftarrow \mathsf{PCP.V}^\pi(u)$: takes the instance u and makes queries into the string π. It output a bit b.

Definition 5.2 (PCP)**.** *A PCP for the universal relation* $\mathcal{R}_\mathcal{U}$ $\mathsf{PCP} = (\mathsf{PCP.P}, \mathsf{PCP.V})$ *satisfies:*

Efficient Verifier and Relatively-Efficient Prover: *There exists a universal polynomial* $p(\cdot)$ *such that for every* $(u, w) = ((M, x, t), w) \in \mathcal{R}_\mathcal{U}$*:*

- *The prover* $\mathsf{PCP.P}(u, w)$ *runs in time* $p(|u|, t)$.

- *The verifier* $\mathsf{PCP.V}^{(\cdot)}(u)$ *runs in time* $p(|u|)$.

Completeness: *For every* $(u, w) \in \mathcal{R}_{\mathcal{U}}$*, and* $\pi \leftarrow \mathsf{PCP.P}(u, w)$*:*

$$\Pr\left[\mathsf{PCP.V}^{\pi}(u) = 1\right] = 1 \enspace .$$

Witness Decoding: *There exists a decoder* $\mathsf{PCP.D}$ *such that for any* π^* *and any* $u = (M, x, t)$ *if*

$$\Pr\left[\mathsf{PCP.V}^{\pi^*}(u)\right] \geq 2^{-|u|} \enspace ,$$

then $\mathsf{PCP.D}(u, \pi)$ *outputs a witness* $w \in \mathcal{R}_{\mathcal{U}}(u)$ *in time* $\mathrm{poly}(|u|, t)$.

PCPs as above exist in the literature (see for instance [BG08] and references within).

5.3 Construction

We now describe the protocol based on multi-collision-resistant hashing with local opening. The protocol follows the standard recipe of combining PCPs with hashing with local openings [Kil92, Mic00, BG08]. The essential difference is that instead of the absolute binding guarantee that is typically provided by completely collision-resistant hash tress, we only have the weak binding guarantee of multi-collision resistance as defined and constructed in Section 4.

We turn to describe the construction. In what follows:

- $\mathsf{PCP} = (\mathsf{PCP.P}, \mathsf{PCP.V})$ is a PCP system (as defined in Section 5.2).
- $\mathsf{HLO} = (\mathsf{HLO.Gen}, \mathsf{HLO.Hash}, \mathsf{HLO.Chal}, \mathsf{HLO.Auth}, \mathsf{HLO.Ver})$ is a hash with local opening (as defined in Section 4).

Notation. We will use the following notation:

- When we want to be explicit about the PCP verifier's randomness r, we write $\mathsf{PCP.V}^{\pi}(u; r)$.
- We denote by $\rho = \rho(\lambda)$ the number of queries that $\mathsf{PCP.V}$ makes, for $u \in \{0,1\}^{\lambda}$.
- For a string π, and verifier randomness r, we denote by $I^{\pi}_{u,r}$ the set of queries that the verifier makes:

$$I^{\pi}_{u,r} := \left\{ i_1, \dots, i_{\rho} \in [|\pi|] \;\middle|\; i_j \text{ is the } j\text{-th query made by } \mathsf{PCP.V}^{\pi^*}(u; r) \right\} \enspace .$$

- We denote by $L = L(\lambda, t)$ the length of a PCP proof generated by $\mathsf{PCP.P}$ for any instance $u = (M, x, t) \in \{0,1\}^{\lambda} \cap \mathcal{L}(\mathcal{R}_{\mathcal{U}})$.

We will now show that the above protocol is a succinct argument assuming multi-collision resistance of the underlying hash with local opening. In the following, let $\bar{t}(\lambda)$ be any time bound function and let $\bar{L}(\lambda) = L(\lambda, \bar{t}(\lambda))$ be the bound on the length of corresponding PCP proofs.

Theorem 5.2. *Assume* HLO *is* K*-collision resistant for input-length bound* $\bar{L}$*. Then Protocol 1 is a succinct argument for* $\bar{t}$*-bounded computations. If* HLO *is keyless, then the protocol has* 3*-messages.*

Furthermore, the witness extractor runs in time:

$$\tau(\lambda, t, \varepsilon^{-1}, \zeta) = \mathrm{poly}(\lambda, t, \varepsilon^{-1}, K) \enspace ,$$

for $K = K(\lambda, \zeta, t^{O(1)})$.

Protocol 1

Common Input: an instance $u = (M, x, t) \in \mathcal{L}(\mathcal{R}_{\mathcal{U}}) \cap \{0,1\}^\lambda$, for security parameter λ.

Auxiliary Input of P**:** a witness $w \in \mathcal{R}_{\mathcal{U}}(x)$.

1. V samples a key $\mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda)$ and sends hk to P.
   In Keyless Setting: this message is not sent. $\mathsf{hk} \equiv 1^\lambda$ throughout the protocol.

2. P computes a proof $\pi \leftarrow \mathsf{PCP.P}(u, w)$ and a digest $\mathsf{dig} = \mathsf{HLO.Hash}(\mathsf{hk}, \pi)$.
   It sends dig to V.

3. V samples random coins r for $\mathsf{PCP.V}^{(\cdot)}(u)$ and a challenge $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$.
   It sends (r, ch) to P.

4. P computes the set of verifier queries $I := I^\pi_{u,r}$ and a proof $\Pi = \mathsf{HLO.Auth}(\mathsf{hk}, \pi, I, \mathsf{ch})$.
   It sends $(I, \pi|_I, \Pi)$ to V.

5. V verifies consistency of the opened proof bits with the digest dig by running $\mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \pi|_I, \mathsf{ch}, \Pi)$. It also verifies the PCP, and that $I = I^\pi_{u,r}$, by running $\mathsf{PCP.V}^{\pi|_I}(u; r)$. It accepts if and only if all checks pass.

Figure 1: A Succinct Argument for Non-Deterministic Computations.

Proof. We first note that completeness as well as verifier and prover efficiency requirements follow directly from that of the underlying PCP and the hash with local opening.

From hereon, we focus on establishing the proof of knowledge property. We first describe the extractor. Fix any PPT prover P^* with polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$, any sequence of inputs $u = (M, x, t) \in \{0,1\}^\lambda$, and any noticeable convincing probability $\varepsilon(\lambda) = \lambda^{-O(1)}$. As before, we let $L = L(\lambda, t)$ denote the length of PCP proofs and let $\rho = \rho(\lambda)$ be the number of queries that $\mathsf{PCP.V}$ makes for instances u. Also, we let $K = K(\lambda, |z_\lambda|, L)$, where K is the collision bound of HLO.

The Extractor $\mathsf{E}^{\mathsf{P}^*(\cdot; z_\lambda)}(u; 1^{1/\varepsilon}, |z_\lambda|)$**:**

1. Sample a key $\mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda)$, feed hk to P^*, who provides a digest dig.
   In Keyless Setting: $\mathsf{hk} \equiv 1^\lambda$ is fed to P^*.

2. Let $\mathsf{P}^*_2(\cdot; \mathsf{st})$ capture how P^* authenticates in the second message. That is, $\mathsf{P}^*_2(\cdot; \mathsf{st})$ given a challenge ch, samples PCP randomness r for $\mathsf{PCP.V}^{(\cdot)}(u)$ on its own, and runs $\mathsf{P}^*(r, \mathsf{ch}; \mathsf{st})$, where st is the state of P^* right after it produced the digest dig in the previous step.

   Obtain a set S of candidate PCPs from the HLO extractor:

$$S \leftarrow \mathsf{Ext}^{\mathsf{P}^*_2(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon'}) \ .$$

3. For every $\pi^* \in S$ decode a candidate witness $w^* = \mathsf{PCP.D}(u, \pi^*)$. If $w^* \in \mathcal{R}_{\mathcal{U}}(u)$, output w^*.

4. The extractor repeats the above three steps for at most λ/ε times, and aborts if no witness is found.

Running Time. As required, the extractor runs in time

$$\tau(\lambda, t, \varepsilon^{-1}, |z_\lambda|) = \lambda \cdot \varepsilon^{-1} \cdot \mathrm{poly}(\lambda, L, \varepsilon^{-1}, K(\lambda, |z_\lambda|, L)) = \mathrm{poly}(\lambda, t, \varepsilon^{-1}, K(\lambda, |z_\lambda|, t^{O(1)})) \ .$$

Successful Witness Extraction. We now analyze the extractor. We first recall that the fact that P^* convinces the verifier V with probability ε, which implies the following:

$$\Pr\left[\begin{array}{l|l} I = I_{u,r}^{\pi^*} & \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \pi^*|_I, \mathsf{ch}, \Pi) = 1 & (\mathsf{dig}, \mathsf{st}) \leftarrow \mathsf{P}^*(\mathsf{hk}; z_\lambda) \\ \mathsf{PCP.V}^{\pi^*|_I}(u; r) = 1 & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ & r \leftarrow \textit{randomness for } \mathsf{PCP.V}^{(\cdot)}(u) \\ & (I, \pi^*|_I, \Pi) \leftarrow \mathsf{P}^*(r, \mathsf{ch}; \mathsf{st}) \end{array}\right] \geq \varepsilon \ .$$

Next, by the HLO extraction guarantee, we know that except with small probability, the prover always opens consistently with the extracted set of strings:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ & (\mathsf{dig}, \mathsf{st}) \leftarrow \mathsf{P}^*(\mathsf{hk}; z_\lambda) \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \pi^*|_I, \mathsf{ch}, \Pi) = 1 & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ \pi^*|_I \notin \{\pi|_I : \pi \in S\} & r \leftarrow \textit{randomness for } \mathsf{PCP.V}^{(\cdot)}(u) \\ & (I, \pi^*|_I, \Pi) \leftarrow \mathsf{P}^*(r, \mathsf{ch}; \mathsf{st}) \\ \cline{2-2} & S \leftarrow \mathsf{Ext}^{\mathsf{P}_2^*(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \end{array}\right] \leq \varepsilon/2 \ .$$

It follows that with noticeable probability the verifier V accepts an opening that is consistent with the extracted set of strings:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{HLO.Gen}(1^\lambda) \\ I = I_{u,r}^{\pi^*} & (\mathsf{dig}, \mathsf{st}) \leftarrow \mathsf{P}^*(\mathsf{hk}; z_\lambda) \\ \mathsf{PCP.V}^{\pi^*|_I}(u; r) = 1 & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ \pi^*|_I \in \{\pi|_I : \pi \in S\} & r \leftarrow \textit{randomness for } \mathsf{PCP.V}^{(\cdot)}(u) \\ & (I, \pi^*|_I, \Pi) \leftarrow \mathsf{P}^*(r, \mathsf{ch}; \mathsf{st}) \\ \cline{2-2} & S \leftarrow \mathsf{Ext}^{\mathsf{P}_2^*(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \end{array}\right] \geq \varepsilon/2 \ .$$

We shall call $(\mathsf{hk}, (\mathsf{dig}, \mathsf{st}), \mathsf{ch}, S)$ good if conditioned on these values the above occurs with good probability:

$$\Pr\left[\begin{array}{l|l} I = I_{u,r}^{\pi^*} & \\ \mathsf{PCP.V}^{\pi^*|_I}(u; r) = 1 & r \leftarrow \textit{randomness for } \mathsf{PCP.V}^{(\cdot)}(u) \\ & (I, \pi^*|_I, \Pi) \leftarrow \mathsf{P}^*(r, \mathsf{ch}; \mathsf{st}) \\ \pi^*|_I \in \{\pi|_I : \pi \in S\} & \end{array}\right] \geq \varepsilon/4 \ .$$

First, by averaging, we know that the probability that $(\mathsf{hk}, (\mathsf{dig}, \mathsf{st}), \mathsf{ch}, S)$ are good is at least $\varepsilon/4$. Furthermore, for any such good tuple, by the fact that the extracted set is always small $|S| \leq K$, there exists $\pi \in S$ such that

$$\Pr\left[\begin{array}{l|l} I = I_{u,r}^{\pi^*} & \\ \mathsf{PCP.V}^{\pi^*|_I}(u; r) = 1 & r \leftarrow \textit{randomness for } \mathsf{PCP.V}^{(\cdot)}(u) \\ & (I, \pi^*|_I, \Pi) \leftarrow \mathsf{P}^*(r, \mathsf{ch}; \mathsf{st}) \\ \pi^*|_I = \pi|_I & \end{array}\right] \geq \varepsilon/4K \ .$$

In particular, for such π,

$$\Pr_r\left[\mathsf{PCP.V}^{\pi}(u; r) = 1\right] \geq \varepsilon/4K > 2^{-\lambda} \ ,$$

in which case, $\mathsf{PCP.D}(u, \pi)$ outputs a valid witness w.

It follows that whenever the extractor samples a good tuple it finds a witness. Since good tuples occur with probability $\varepsilon/4$, after λ/ε attempts, the extractor will find a witness except with probability $2^{-\lambda/4}$.

This completes the proof of the theorem. □

6 3-Message Zero Knowledge via Weak Memory Delegation

In this section, we construct a 3-message zero-knowledge argument for **NP**, which can be based on weak multi-collision resistance and fully-homomorphic encryption, both with slight super-polynomial hardness.

The main tool behind these constructions is *weak memory delegation*. We start by defining this notion and explaining why it is sufficient for the goal of 3-message zero knowledge. Then (through most of this section) we concentrate on constructing weak memory delegation schemes based on multi-collision resistant hashing with local opening.

6.1 Weak Memory Delegation

In a two-message memory delegation scheme [CKLR11], an untrusted server provides the client a short commitment or *digest* dig of a large memory D. The client can then delegate any arbitrary deterministic computation M to be executed over the memory. The server responds with the computation's output y, as well as a short proof of correctness that can be verified by the client in time that is independent of that of the delegated computation and the size of the memory.

In the common definition of memory delegation, the soundness requirement says that having provided the digest dig, the prover should not be able to prove that a given computation M results in more than a single outcome y. Naturally, this requires that the short digest fixes (in a computational sense) at most a single underlying memory, which is usually achieved based on collision-resistant hashing. With the aim of replacing collision-resistance with multi-collision resistance (and to potentially avoid an extra setup step), we consider a weaker soundness requirement. The requirement roughly says that the attacker should not be able to prove consistency with *too many* outcomes y. There are several conceivable ways to capture this requirement. For simplicity, we give a somewhat restricted form of this intuition that only says that that the attacker should not be able to prove the correctness of an outcome $y \leftarrow Y$, sampled at random from a sufficiently entropic distribution Y, except with small probability.

We proceed to present the formal syntax and definition.

Syntax: A two-message *memory delegation scheme* consists of algorithms:

$$(\mathsf{MD.Gen}, \mathsf{MD.Mem}, \mathsf{MD.Query}, \mathsf{MD.Prove}, \mathsf{MD.Ver}) \ ,$$

with the following syntax:

- $\mathsf{pp} \leftarrow \mathsf{MD.Gen}(1^\lambda)$: a randomized polynomial-time algorithm that given the security parameter 1^λ, outputs public parameters pp. In the keyless setting, this algorithm is deterministic and outputs fixed parameters $\mathsf{pp} \equiv 1^\lambda$.
- $\mathsf{dig} \leftarrow \mathsf{MD.Mem}(\mathsf{pp}, D)$: a deterministic polynomial-time algorithm that given pp and a memory D, outputs a digest dig of the memory.
- $(\mathsf{q}, \mathsf{vst}) \leftarrow \mathsf{MD.Query}(1^\lambda)$: a randomized polynomial-time algorithm that given the security parameter 1^λ, outputs a query q and a secret state vst.
- $\pi \leftarrow \mathsf{MD.Prove}(\mathsf{pp}, D, (M,t,y), \mathsf{q})$: a deterministic algorithm that takes the public parameters pp, a memory string D, a (deterministic) Turing machine M, an output string y, and time bound t, such that $(M,t,y) \in \{0,1\}^\lambda$ and $|D| \le t \le 2^\lambda$. It outputs a proof π that $M(D)$ outputs y within t steps.
- $b \leftarrow \mathsf{MD.Ver}(\mathsf{pp}, \mathsf{dig}, (M,t,y), \mathsf{vst}, \pi)$: a deterministic polynomial time oracle algorithm that takes the public parameters pp, a digest dig, a (deterministic) Turing machine M, a time bound t, and an output string y, such that $(M,t,y) \in \{0,1\}^\lambda$, together with a pair (vst, π), and outputs an acceptance bit b.

Definition 6.1 (Entropic Distribution Ensemble)**.** *We say that an efficiently samplable distribution ensemble* $\{Y_\lambda\}_{\lambda\in\mathbb{N}}$ *is entropic if*

$$H_\infty(Y_\lambda) := -\log \max_{y \in \mathrm{supp}(Y_\lambda)} \Pr[Y_\lambda = y] = \Omega(\lambda) \ .$$

Definition 6.2 (Weak Memory Delegation)**.** *A two-message delegation scheme*

$$\mathsf{MD} = (\mathsf{MD.Gen}, \mathsf{MD.Mem}, \mathsf{MD.Query}, \mathsf{MD.Prove}, \mathsf{MD.Ver})$$

satisfies:

Efficient Verifier and Relatively-Efficient Prover: *There exists a universal polynomial* $p(\cdot)$ *such that for every* $(M,t,y) \in \{0,1\}^\lambda$*, and every* D *such that* $M(D)$ *outputs* y *within* t *steps, and* $|D| \le t \le 2^\lambda$*:*

- *The prover* $\mathsf{MD.Prove}(\mathsf{pp}, D, (M, t, y), \mathsf{q})$ *runs in time* $p(\lambda, t)$.
- *The verifier* $\mathsf{MD.Ver}(\mathsf{pp}, \mathsf{dig}, (M, t, y), \mathsf{vst}, \pi)$ *runs in time* $p(\lambda)$.

Correctness: *For every security parameter* $\lambda \in \mathbb{N}$*, every* $(M, t, y) \in \{0,1\}^{\lambda}$*, and every* D *such that* $M(D)$ *outputs* y *within* t *steps, and* $|D| \leq t \leq 2^{\lambda}$*:*

$$\Pr\left[\mathsf{MD.Ver}(\mathsf{pp}, \mathsf{dig}, (M, t, y), \mathsf{vst}, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{MD.Gen}(1^{\lambda}) \\ \mathsf{dig} \leftarrow \mathsf{MD.Mem}(\mathsf{pp}, D) \\ (\mathsf{q}, \mathsf{vst}) \leftarrow \mathsf{MD.Query}(1^{\lambda}) \\ \pi \leftarrow \mathsf{MD.Prove}(\mathsf{pp}, D, (M, t, y), \mathsf{q}) \end{array} \right] = 1 \ .$$

Weak Soundness for Computation-Time Bound $\bar{t}(\lambda)$***:*** *For every pair of PPT adversaries* $(\mathcal{A}_1, \mathcal{A}_2)$ *and polynomial-size advice* $\{z_{\lambda}\}_{\lambda \in \mathbb{N}}$*, there exists a negligible function* μ*, such that for every ensemble of samplable entropic distributions* $\{Y_{\lambda}\}_{\lambda \in \mathbb{N}}$*, every* $t(\lambda) \leq \bar{t}^{O(1)}$*, every* $\lambda \in \mathbb{N}$*, letting* $K = K(\lambda, |z_{\lambda}|, t)$*,*

$$\Pr\left[\mathsf{MD.Ver}(\mathsf{pp}, \mathsf{dig}, (M, t, y), \mathsf{vst}, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{MD.Gen}(1^{\lambda}) \\ (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pp}; z_{\lambda}) \\ (\mathsf{q}, \mathsf{vst}) \leftarrow \mathsf{MD.Query}(1^{\lambda}) \\ y \leftarrow Y_{\lambda} \\ \pi \leftarrow \mathcal{A}_2(\mathsf{q}, y; \mathsf{st}) \end{array} \right] \leq \mu(\lambda) \ .$$

We now state the main theorem proved in this section regarding the existence of two-message memory delegation (according to the above definition) based on weak multi-collision-resistant hash functions (Definition 3.2). The theorem has three parts. The first is a polynomial version that guarantees security for computations of arbitrary polynomial time, based on polynomial multi-collision resistance and quasi-polynomially-secure fully-homomorphic encryption. The second is a super-polynomial version that guarantees security even for slightly super-polynomial computations, relying on slightly super-polynomial multi-collision resistance. The first two parts address polynomial compression, the third part addresses linear compression.

Theorem 6.1 (Two-Message Delegation)**.**

- *Assuming a polynomially-compressing weakly* K*-collision-resistant hash for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$*, and* $\mathrm{quasipoly}(\lambda)$*-secure fully-homomorphic encryption, there exists a two-message memory-delegation scheme with weak soundness for any computation-time bound* $\bar{t}(\lambda) = \lambda^{O(1)}$.
- *For any (arbitrary small)* $\tau(\lambda) = \omega(1)$*, there exists* $\bar{t}(\lambda) = \lambda^{\omega(1)}$ *such that assuming a polynomially-compressing weakly* (K, γ)*-collision-resistant hash, for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$ *and* $\gamma(\lambda) = \lambda^{\tau}$*, and* $\mathrm{quasipoly}(\lambda)$*-secure fully-homomorphic encryption, there exists a two-message memory-delegation scheme with weak soundness for computation-time bound* $\bar{t}$*.*
- *For any (arbitrary small)* $\tau(\lambda) = \omega(\log \lambda)$*, there exists* $\bar{t}(\lambda) = \lambda^{\omega(1)}$ *such that assuming a weakly* (K, γ)*-collision-resistant hash, for* $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$ *and* $\gamma(\lambda) = \lambda^{\tau}$*, and* $\mathrm{quasipoly}(\lambda)$*-secure fully-homomorphic encryption, there exists a two-message memory-delegation scheme with weak soundness for computation-time bound* $\bar{t}$*.*

If the underlying hash functions are keyless, the scheme does not require trusted public parameters.

In the next sections, we describe the construction behind the theorem, and its building blocks. Eventually, the theorem is derived as a corollary of a more general Theorem 6.4 proven in Section 6.3.

From Weak Memory Delegation to 3-Message Zero Knowledge. As explained in the intro, in [BBK+16] a 3-message zero knowledge protocol is constructed in the *global hash model*, where we assume the existence of a collision-resistant hash function as a public parameter generated in a setup phase. We show how to replace the collision-resistant hash with an keyless weakly multi-collision-resistant hash, to deduce:

Theorem 6.2 (Corollary of Theorem 6.1 and [BBK+16]). *For any function $\gamma(\lambda) = \lambda^{-\omega(\log \lambda)}$, assume a linearly-compressing weakly (K,γ)-collision-resistant hash for $K(\lambda,\zeta) = \mathrm{poly}(\lambda,\zeta)$, $\mathrm{quasipoly}(\lambda)$-secure fully-homomorphic encryption, circuit-private 1-hop homomorphic encryption, and non-interactive perfectly-binding commitments. Then there exists a 3-message zero-knowledge argument for any language in **NP**.*

Remark 6.1. We refer the reader to Appendix B.2 for the definition of a circuit-private 1-hop homomorphic encryption, but mention that it can be constructed based on many standard cryptographic assumptions, and in particular, based on the learning with errors assumption.

We refer the reader to Appendix B.1 for the construction of our 3-round zero-knowledge protocol.

Proof Sketch. In [BBK+16], the collision-resistant hash function is used for two purposes:

- As the public parameters for a two-message memory delegation scheme.
- In the construction of a 3-message witness-indistinguishable proof of knowledge with first-message-dependent instances (see definition in Appendix B.1). We note that this use is somewhat less essential and can be avoided assuming, in addition, ZAPs [DN07].

We now argue that weak multi-collision-resistant is sufficient for both.

For the first one, we recall that the soundness (or proof-of-knowledge) guarantee of the [BBK+16] protocol is shown by a reduction to the soundness of memory delegation. Concretely, the reduction [BBK+16, Section 3.1] transforms any convincing prover into an attacker for the memory delegation scheme. We observe that the constructed attacker is, in fact, strong enough to also break multi-collision resistance. Specifically, the constructed attacker samples a digest dig and a computation M such that with noticeable probability ε (in [BBK+16], $\eta^{\Omega(1)}$) over a random output y, it successfully produces a convincing proof consistently with (M, y), which is enough to break weak soundness of memory delegation.

The second use of collision-resistance is in [BBK+16, Section 2.5], where the 3-message witness-indistinguishability protocol of Lapidot and Shamir [LS90a] is transformed into one where the first message has a fixed length, independently of the statement to be proven. There, collision-resistance is used to hash the first (commitment) message in the LS protocol, where the preimage commitment is revealed at the end. To prove that this system is an argument of knowledge, one relies on the fact that when rewinding an ε-convincing prover, to obtain different openings, the prover always opens the same commitment (hash preimage), or it could break collision-resistance. Here we can replace the collision-resistant hash with a K-collision-resistant hash, we still have the guarantee that one of at most K commitments is opened with high probability $\varepsilon^{\Omega(1)}/K$ with different challenges. □

6.2 Oracle Memory Delegation

Toward the construction of (weak) memory delegation we define a weaker notion called *oracle memory delegation*, which will be used as a building block in our construction. Roughly speaking, this is a memory delegation scheme in a hybrid model, where rather than providing a short digest to a memory. The server provides the memory in full *as an oracle*. The client, however, only accesses a small part of this oracle (similar to the interactive PCP model of [KR08]).

Syntax: A two-message oracle memory delegation scheme consists of algorithms:

$$(\mathsf{OMD.Mem}, \mathsf{OMD.Query}, \mathsf{OMD.Prove}, \mathsf{OMD.Ver}) \ ,$$

with the following syntax:

- $\widehat{D} \leftarrow \mathsf{OMD.Mem}(1^\lambda, D)$: a deterministic polynomial-time algorithm that given the security parameter 1^λ, and a memory D, outputs an encoding $\widehat{D}$ of the memory.
- $(\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda)$: a randomized polynomial-time algorithm that given the security parameter 1^λ outputs a query q, a set $I \subseteq \mathbb{N}$ of oracle queries, and a secret state vst.
- $\pi \leftarrow \mathsf{OMD.Prove}(D, (M, t, y), \mathsf{q})$: a deterministic algorithm that takes a memory string D, a (deterministic) Turing machine M, an output string y, and time bound t, such that $(M, t, y) \in \{0,1\}^\lambda$ and $|D| \le t \le 2^\lambda$. It outputs a proof π that $M(D)$ outputs y within t steps.

- $b \leftarrow \mathsf{MD.Ver}^{(\cdot)}((M,t,y),\mathsf{vst},\pi)$: a deterministic polynomial time oracle-aided algorithm that takes a (deterministic) Turing machine M, a time bound t, and an output string y, such that $(M,t,y) \in \{0,1\}^\lambda$, together with a pair (vst,π), and outputs an acceptance bit b.

Definition 6.3. *A two-message oracle-memory delegation scheme* $\mathsf{OMD} = (\mathsf{OMD.Mem}, \mathsf{OMD.Query}, \mathsf{OMD.Prove}, \mathsf{OMD.Ver})$ *satisfies:*

Efficient Verifier and Relatively-Efficient Prover: *There exists a universal polynomial $p(\cdot)$ such that for every $(M,t,y) \in \{0,1\}^\lambda$, and every D such that $M(D)$ outputs y within t steps, and $|D| \le t \le 2^\lambda$:*

- *The prover* $\mathsf{OMD.Prove}(D,(M,t,y),\mathsf{q})$ *runs in time $p(\lambda,t)$.*
- *The verifier* $\mathsf{MD.Ver}^{(\cdot)}((M,t,y),\mathsf{vst},\pi)$ *runs in time $p(\lambda)$.*

Correctness: *For every security parameter $\lambda \in \mathbb{N}$, every $(M,t,y) \in \{0,1\}^\lambda$, and every D such that $M(D)$ outputs y within t steps, and $|D| \le t \le 2^\lambda$:*

$$\Pr\left[\mathsf{OMD.Ver}^{\widehat{D}|_I}((M,t,y),\mathsf{vst},\pi) = 1 \;\middle|\; \begin{array}{l} \widehat{D} \leftarrow \mathsf{OMD.Mem}(1^\lambda, D) \\ (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ \pi \leftarrow \mathsf{OMD.Prove}(D,(M,t,y),\mathsf{q}) \end{array} \right] = 1 \ .$$

*γ**-Soundness for Computation-Time Bound** $\bar{t}(\lambda)$**:** for every pair of probabilistic $\gamma^{O(1)}$-time adversaries $(\mathcal{A}_1,\mathcal{A}_2)$ and polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a negligible function μ, such that for every $t(\lambda) \le \bar{t}^{O(1)}$, every $\lambda \in \mathbb{N}$, letting $K = K(\lambda,|z_\lambda|,t)$,*

$$\Pr\left[\begin{array}{l} y \ne y' \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M,t,y),\mathsf{vst},\pi) = 1 \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M,t,y'),\mathsf{vst},\pi') = 1 \end{array} \;\middle|\; \begin{array}{l} (\widehat{D}^*, M, y, y', \mathsf{st}) \leftarrow \mathcal{A}_1(1^\lambda; z_\lambda) \\ (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ (\pi,\pi') \leftarrow \mathcal{A}_2(\mathsf{q};\mathsf{st}) \end{array} \right] \le \mu(\gamma(\lambda)) \ .$$

Theorem 6.3 (Follows from [KRR14])**.** *For any functions $\bar{t}(\lambda), \gamma \le 2^\lambda$, assuming the existence of levelled fully homomorphic encryption (*FHE*) that is $\gamma \cdot \mathrm{quasipoly}(\bar{t})$-secure, there exists a 2-message oracle-memory delegation scheme that is γ-sound for time bound $\bar{t}$.*

Proof Sketch. Kalai, Raz, and Rothblum [KRR14] show that a levelled FHE that is $\gamma \cdot \mathrm{quasipoly}(\bar{t})$-secure implies a γ-sound 2-message delegation scheme (for all deterministic computations) for $\bar{t}$-time computations. Furthermore, they prove that the verifier does not need to read the input, but rather only needs to read a random point in the low-degree extension of the input. This point can be generated non-adaptively ahead of time. Also, the complexity of the verifier does not depend on the input length, and is a fixed in the security parameter (provided that the input is shorter than 2^λ.)

This suggests the existence of an oracle-memory delegation scheme where the memory D is treated as the input, and its encoding $\widehat{D}$ is simply its low-degree extension. The only gap is that in [KRR14], it is assumed that $\widehat{D}$ is an honest low-degree extension, whereas in our case, the adversary may output an arbitrary encoding. This gap is naturally bridged by adding a low-degree test. We note that the queries for a low degree test can be generated non-adaptively ahead of time, and the complexity of the test only depends on the security parameter. □

6.3 Construction

We now construct a weak memory delegation scheme. We essentially show how to compile any oracle-memory-delegation scheme into a weak (non-oracle) memory delegation scheme based on multi-collision-resistant hashing with local opening.

We turn to describe the construction. In what follows:

- $\mathsf{OMD} = (\mathsf{OMD.Mem}, \mathsf{OMD.Query}, \mathsf{OMD.Prove}, \mathsf{OMD.Ver})$ is an oracle-memory delegation scheme (as defined in Section 6.2).

- $\mathsf{HLO} = (\mathsf{HLO.Gen}, \mathsf{HLO.Hash}, \mathsf{HLO.Chal}, \mathsf{HLO.Auth}, \mathsf{HLO.Ver})$ is a hash with local opening (as defined in Section 4).
- $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ is a secret-key (leveled) fully homomorphic encryption scheme [Gen09].

Notation. We will use the following notation:

- We denote by $\rho = \rho(\lambda)$ be the number of oracle queries generated by $\mathsf{OMD.Query}(1^\lambda)$.
- We denote by $L = L(\lambda, t)$ the maximal length of an encoded oracle $\widehat{D}$ generated by $\mathsf{OMD.Mem}(1^\lambda, D)$ for any memory D of size at most t.

We now describe the scheme's algorithms:

$$\mathsf{MD} = (\mathsf{MD.Gen}, \mathsf{MD.Mem}, \mathsf{MD.Query}, \mathsf{MD.Prove}, \mathsf{MD.Ver}) \ .$$

- $\mathsf{MD.Gen}(1^\lambda)$:
  - Sample a key $\mathsf{HLO.Gen}(1^\lambda)$.
  - Output $\mathsf{pp} = \mathsf{hk}$. In the keyless setting, $\mathsf{pp} \equiv 1^\lambda$.
- $\mathsf{MD.Mem}(\mathsf{hk}, D)$:
  - Encode the memory $\widehat{D} = \mathsf{OMD.Mem}(D)$.
  - Output the digest $\mathsf{dig} = \mathsf{HLO.Hash}(\mathsf{hk}, \widehat{D})$.
- $\mathsf{MD.Query}(1^\lambda)$:
  - Sample $(\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda)$ and
  - Sample a secret key $\mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and encrypt the oracle queries $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(I)$.
  - Sample a challenge $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$, where recall that ρ is the size of query set I.
  - Output the query $\mathsf{q}' = (\mathsf{q}, \mathsf{ct}, \mathsf{ch})$ and state $\mathsf{vst}' = (\mathsf{vst}, \mathsf{ch}, \mathsf{sk})$.
- $\mathsf{MD.Prove}(\mathsf{hk}, D, (M, t, y), (\mathsf{q}, \mathsf{ct}, \mathsf{ch}))$:
  - Compute a proof $\pi \leftarrow \mathsf{OMD.Prove}(D, (M, t, y), \mathsf{ct})$,
  - Let $A(I)$ be the function that computes $\Pi = \mathsf{HLO.Auth}(\mathsf{hk}, \widehat{D}, I, \mathsf{ch})$, and outputs $(\widehat{D}|_I, \Pi)$.
  - Homomorphically compute $\widehat{\mathsf{ct}} = \mathsf{FHE.Eval}(\mathsf{ct}, A(\cdot))$.
  - Output as the proof $\pi' = (\pi, \widehat{\mathsf{ct}})$.
- $\mathsf{MD.Ver}(\mathsf{hk}, \mathsf{dig}, (M, t, y), (\mathsf{vst}, \mathsf{ch}, \mathsf{sk}), (\pi, \widehat{\mathsf{ct}}))$:
  - Decrypt $(\widehat{D}|_I, \Pi) = \mathsf{FHE.Dec}_{\mathsf{sk}}(\widehat{\mathsf{ct}})$.
  - Verify consistency with digest $\mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \widehat{D}|_I, \mathsf{ch}, \Pi)$.
  - Run the oracle verifier $\mathsf{OMD.Ver}^{\widehat{D}|_I}((M, t, y), \mathsf{vst}, \pi)$.
  - Accept if and only if both checks pass.

We will now show that the above scheme is a weak memory delegation scheme assuming multi-collision resistance of the underlying hash with local opening. In the following, let $\bar{t}(\lambda)$ be any time bound function and let $\bar{L}_{\bar{t}}(\lambda) = L(\lambda, \bar{t}(\lambda))$ be the bound encoded memories $\widehat{D}$ for $|D| \leq \bar{t}(\lambda)$. Also, in what follows, $K = K(\lambda, \zeta, L)$ is such that $K(\lambda, \lambda, \bar{L}_{\bar{t}}) \leq 2^{o(\lambda)}$ and $\gamma(\lambda) = K(\lambda, \lambda, \bar{L}_{\bar{t}}) \leq 2^{o(\lambda)}$.

Theorem 6.4. *Assume that* HLO *is* K*-collision resistant for input-length bound* $\bar{L}_{\bar{t}}$*,* OMD *is* γ*-sound for computation-time-bound* $\bar{t}$*, and* FHE *is* γ*-secure. Then the scheme* MD *is a memory delegation scheme with* K*-weak soundness for computation-time bound* $\bar{t}$*. If* HLO *is keyless, then scheme does not require trusted public parameters.*

Proof. We first note that correctness as well as verifier and prover efficiency requirements follow directly from that of the underlying oracle-memory delegation and the hash with local opening.

From hereon, we focus on proving weak soundness. Fix any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ any polynomial-size advice $\{z_\lambda\}_{\lambda\in\mathbb{N}}$, any entropic distribution ensemble $\{Y_\lambda\}_{\lambda\in\mathbb{N}}$ and any noticeable function $\varepsilon(\lambda) = \lambda^{-O(1)}$. As before, we let $L = L(\lambda, t)$ denote the length of any encoded memory $\widehat{D}$ for $|D| \le t$, and let $\rho = \rho(\lambda)$ be the number of oracle queries that $\mathsf{OMD.Query}(1^\lambda)$ generates. Also, we let $K = K(\lambda, |z_\lambda|, L)$, where K is the collision bound of HLO.

Assume toward contradiction that for infinitely many $\lambda \in \mathbb{N}$, and $t(\lambda) \le \bar{t}^{O(1)}$, the adversary $\mathcal{A}$ violates K-weak soundness:

$$\Pr\left[\mathsf{MD.Ver}(\mathsf{hk}, \mathsf{dig}, (M,t,y), \mathsf{vst}', \pi') = 1 \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ (\mathsf{q}', \mathsf{vst}') \leftarrow \mathsf{MD.Query}(1^\lambda) \\ y \leftarrow Y_\lambda \\ \pi' \leftarrow \mathcal{A}_2(\mathsf{q}, y; \mathsf{st}) \end{array} \right] \ge \varepsilon \ .$$

Spelling this out according to the construction,[16]

$$\Pr\left[\begin{array}{l} \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \widehat{D}^*|_I, \mathsf{ch}, \Pi) = 1 \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M,t,y), \mathsf{vst}, \pi) = 1 \\ \textit{where } (\widehat{D}^*|_I, \Pi) := \mathsf{FHE.Dec_{sk}}(\widehat{\mathsf{ct}}) \end{array} \;\middle|\; \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(I) \\ \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ y \leftarrow Y_\lambda \\ (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st}) \end{array} \right] \ge \varepsilon \ . \tag{10}$$

Our next step is to show that we can extract from $\mathcal{A}$ a set of encoded memories $S = \left\{\widehat{D}\right\}$ of size at most K, which $\mathcal{A}$ is almost always consistent with. We define $\mathcal{A}_2^{\mathsf{HLO}}(\cdot; \mathsf{st})$ which captures how $\mathcal{A}_2$ authenticates its answers.

Adversary $\mathcal{A}_2^{\mathsf{HLO}}(\cdot; \mathsf{st})$: given the state st produced by $\mathcal{A}_1$ after it created a digest dig, does the following:

- It samples on its own:
  - $(\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda)$,
  - $\mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda)$,
  - $\mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(I)$,
  - $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$,
  - $y \leftarrow Y_\lambda$.
- Obtains $(\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st})$.
- Decrypts $(\widehat{D}^*|_I, \Pi) = \mathsf{FHE.Dec_{sk}}(\widehat{\mathsf{ct}})$.
- Outputs $(I, \widehat{D}^*|_I, \Pi)$.

[16]The part of the experiment colored in gray will not change throughout proof.

Next, by the HLO extraction guarantee, there exists an extractor Ext, such that by our definition of $\mathcal{A}_2^{\mathsf{HLO}}$, except with small probability, the actual adversary $\mathcal{A}_2$ opens consistently with the extracted set of strings:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ & (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ & (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ & \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{HLO.Ver}(\mathsf{hk}, L, \mathsf{dig}, I, \widehat{D}^*|_I, \mathsf{ch}, \Pi) = 1 & \mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(I) \\ \widehat{D}^*|_I \notin \left\{\widehat{D}|_I : \widehat{D} \in S\right\} & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ & y \leftarrow Y_\lambda \\ & (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st}) \\ & (\widehat{D}^*|_I, \Pi) = \mathsf{FHE.Dec_{sk}}(\widehat{\mathsf{ct}}) \\ \hline & S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \end{array}\right] \leq \frac{\varepsilon}{2} \ . \qquad (11)$$

Combining Equations 10 and 11, it follows that with noticeable probability the oracle-memory delegation verifier accepts consistently with the extracted set of strings:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ & (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ & (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ \exists \widehat{D}^*|_I \in \left\{\widehat{D}|_I : \widehat{D} \in S\right\}, & \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M, t, y), \mathsf{vst}, \pi) = 1 & \mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(I) \\ & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ & y \leftarrow Y_\lambda \\ & (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st}) \\ \hline & S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \end{array}\right] \geq \frac{\varepsilon}{2} \ .$$

Next, we observe that by the semantic security of the FHE scheme, the same holds when $\mathcal{A}_2$ obtains an encryption that is independent of the oracle queries I, except with negligible probability $\lambda^{-\omega(1)} \ll \varepsilon/4$:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ & (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ & (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ \exists \widehat{D}^*|_I \in \left\{\widehat{D}|_I : \widehat{D} \in S\right\}, & \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M, t, y), \mathsf{vst}, \pi) = 1 & \mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(0^\rho) \\ & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ & y \leftarrow Y_\lambda \\ & (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st}) \\ \hline & S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \end{array}\right] \geq \frac{\varepsilon}{4} \ .$$

Since $|S| \leq K$, it follows that the above event occurs for a random choice of $\widehat{D}^* \leftarrow S$ with good probability:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ & (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ & (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ & \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M, t, y), \mathsf{vst}, \pi) = 1 & \mathsf{ct} \leftarrow \mathsf{FHE.Enc_{sk}}(0^\rho) \\ & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ & y \leftarrow Y_\lambda \\ & (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q}, \mathsf{ct}, \mathsf{ch}), y; \mathsf{st}) \\ \hline & S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \\ & \widehat{D}^* \leftarrow S \end{array}\right] \geq \frac{\varepsilon}{4K} \ . \qquad (12)$$

By a standard averaging argument, we know that with high probability the above also occurs simultaneously for two independent choices of $y, y' \leftarrow Y_\lambda$:

$$\Pr\left[\begin{array}{l|l} & \mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda) \\ & (\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda) \\ & (\mathsf{q}, I, \mathsf{vst}) \leftarrow \mathsf{OMD.Query}(1^\lambda) \\ & \mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda) \\ & \mathsf{ct} \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(0^\rho) \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M,t,y),\mathsf{vst},\pi) = 1 & \mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho) \\ \mathsf{OMD.Ver}^{\widehat{D}^*|_I}((M,t,y'),\mathsf{vst},\pi') = 1 & y, y' \leftarrow Y_\lambda \\ & (\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q},\mathsf{ct},\mathsf{ch}), y; \mathsf{st}) \\ & (\pi', \widehat{\mathsf{ct}}') \leftarrow \mathcal{A}_2((\mathsf{q},\mathsf{ct},\mathsf{ch}), y'; \mathsf{st}) \\ \hline & S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon}) \\ & \widehat{D}^* \leftarrow S \end{array}\right] \geq \Omega\left(\frac{\varepsilon^3}{K^3}\right) .$$

This immediately implies a breaker $\mathcal{A}^{\mathsf{OMD}} = (\mathcal{A}_1^{\mathsf{OMD}}, \mathcal{A}_2^{\mathsf{OMD}})$ for the oracle-memory delegation:

- $\mathcal{A}_1^{\mathsf{OMD}}(1^\lambda; z_\lambda)$ samples:
  - $\mathsf{hk} \leftarrow \mathsf{MD.Gen}(1^\lambda)$,
  - $(\mathsf{dig}, M, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{hk}; z_\lambda)$,
  - $S \leftarrow \mathsf{Ext}^{\mathcal{A}_2^{\mathsf{HLO}}(\cdot;\mathsf{st})}(1^\lambda, 1^\rho, 1^L, 1^K, 1^{2/\varepsilon})$.
  - $\widehat{D}^* \leftarrow S$.
  - $y, y' \leftarrow Y$.
- Output $(\widehat{D}^*, M, y, y', (\mathsf{st}, y, y'))$.
- $\mathcal{A}_2^{\mathsf{OMD}}(\mathsf{q}; (\mathsf{st}, y, y'))$ samples:
  - $\mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda)$,
  - $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(0^\rho)$,
  - $\mathsf{ch} \leftarrow \mathsf{HLO.Chal}(1^\lambda, 1^\rho)$,
  - $(\pi, \widehat{\mathsf{ct}}) \leftarrow \mathcal{A}_2((\mathsf{q},\mathsf{ct},\mathsf{ch}), y; \mathsf{st})$.
    $(\pi', \widehat{\mathsf{ct}}') \leftarrow \mathcal{A}_2((\mathsf{q},\mathsf{ct},\mathsf{ch}), y'; \mathsf{st})$.
- It outputs (π, π').

It follows directly from Equation 12 that $\mathcal{A}^{\mathsf{OMD}}$ breaks the underlying oracle-memory delegation scheme with probability at least $\Omega(\frac{\varepsilon^3}{K^3}) - 2^{-\Omega(\lambda)}$, where the $2^{-\Omega(\lambda)} = 2^{-H_\infty(Y_\lambda)}$ accounts for the probability that y, y' collide.

This completes the proof of the theorem. □

7 1-Message Statistically-Hiding Commitments with Weak Binding

In this section, we define and construct weakly-binding statistically-hiding commitments. The essential difference between such commitments and standard statistically-hiding commitments is in the binding guarantee. Whereas in standard commitments an efficient adversary should not be able to open a commitment to two distinct plaintexts, now we only require that it cannot open a commitment to more than K plaintexts. Analogously to the case of collision-resistant hash functions, under this definition, it is possible to consider such commitments that are completely non-interactive (without any key). Here the number of possible openings may scale with the adversary's non-uniform advice.

In Section 7.1, we define the notion and show how to obtain it from multi-collision-resistant hash functions. In Section 8, we show how to use such commitments to construct ε-zero-knowledge proofs from such commitments. Relying on keyless commitments, the resulting protocols improve the known round-complexity for these tasks.

7.1 Definition

We define weakly-binding statistically-hiding commitments and then describe a construction.

Syntax: A commitment scheme is associated with an input length function $L(\lambda)$ and polynomial-time algorithms $\mathsf{SHC} = (\mathsf{SHC.Gen}, \mathsf{SHC.Com})$ with the following syntax:

- $\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda)$: a probabilistic algorithm that takes the security parameter 1^λ and outputs a key $\mathsf{hk} \in \{0,1\}^\lambda$. In the keyless setting, this algorithm is deterministic and outputs a fixed key $\mathsf{hk} \equiv 1^\lambda$.
- $c \leftarrow \mathsf{SHC.Com}(X; \mathsf{hk})$: a probabilistic algorithm that takes the key hk and an input $X \in \{0,1\}^{L(\lambda)}$ and outputs a commitment $c \in \{0,1\}^\lambda$. When we want to be explicit about the randomness r used by the algorithm, we may write $\mathsf{SHC.Com}(X; \mathsf{hk}, r)$.

Definition 7.1 (Weakly-Binding Statistically-Hiding Commitments)**.** *A weakly-binding statistically-hiding commitment* $\mathsf{SHC} = (\mathsf{SHC.Gen}, \mathsf{SHC.Com})$ *satisfies:*

Statistical Hiding: *For any two plaintexts $X, X' \in \{0,1\}^{L(\lambda)}$, and any key $\mathsf{hk} \in \{0,1\}^\lambda$, the corresponding commitments are statistically close:*

$$\mathsf{SHC.Com}(X; \mathsf{hk}) \approx_s \mathsf{SHC.Com}(X'; \mathsf{hk}) \ .$$

Weak K-Binding: *For any PPT $\mathcal{A}$ and any sequence of polynomial-size advice $\{z_\lambda\}_{\lambda \in \mathbb{N}}$, there is a negligible function μ, such that for any $\lambda \in \mathbb{N}$, letting $K = K(\lambda, |z_\lambda|)$,*

$$\Pr\left[\begin{array}{c} c_1 = \cdots = c_K \\ \forall i \neq j : X_i \neq X_j \end{array} \middle| \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda) \\ ((X_1, r_1) \ldots, (X_K, r_K)) \leftarrow \mathcal{A}(\mathsf{hk}; z_\lambda) \\ \forall i : c_i = \mathsf{SHC.Com}(X_i; \mathsf{hk}, r) \end{array}\right] \leq \mu(\lambda) \ .$$

Remark 7.1 (Plaintext Size). While in standard commitments we typically restrict attention to bit plaintexts, weakly-binding commitments only become meaningful for plaintexts from a super-polynomial message space $\{0,1\}^{L(\lambda)}$ for $L(\lambda) = \omega(\log \lambda)$.

7.2 Construction

We next describe a simple construction based on K-collision resistance. At high-level, the construction is similar to the construction of statistically-hiding commitments from collision-resistant hash functions [DPP93, HM96], only that collision-resistant hash functions are replaced with K-collision resistance. We describe it here for completeness.

Let L be the input length, and let $\mathsf{H} = (\mathsf{H.Gen}, \mathsf{H.Hash})$ be a K-collision-resistant hash function mapping $L(\lambda) + 2\lambda$ bits to λ bits. Let $\mathcal{H} = \{\mathcal{H}_\lambda\}$ be a family of pairwise independent hash functions mapping $L(\lambda) + 2\lambda$ bits to $L(\lambda)$ bits.

- $\mathsf{SHC.Gen}(1^\lambda)$: applies $\mathsf{H.Gen}(1^\lambda)$ and outputs the produced key hk. (In the keyless setting $\mathsf{hk} \equiv 1^\lambda$).
- $\mathsf{SHC.Com}(X; \mathsf{hk})$: samples a pairwise independent $h \leftarrow \mathcal{H}_\lambda$ and randomness $r \leftarrow \{0,1\}^{L(\lambda)+2\lambda}$. It computes $Y_r = \mathsf{H.Hash}(\mathsf{hk}, r)$, $z_r = h(r)$ and outputs: $c = (h, Y_r, z_r \oplus X)$.

Proposition 7.1. *The scheme is statistically-hiding and K-binding.*

The proof is a natural extension of the proof of statistically-hiding commitments from collisions-resistance. Below we give a short sketch. (The proof for statistical hiding will in fact be much simpler than in [DPP93, HM96], by relying on the notion of average min entropy and corresponding generalization of the leftover hash lemma [DORS08].)

Proof Sketch. To prove K-binding, note that for any $c = (h, Y, z)$ and randomness r there could be at most a single X such that $\mathsf{SHC.Com}(X; \mathsf{hk}, r) = c$. Thus opening c to K plaintexts X, implies exhibiting K preimages of Y under $\mathsf{H.Hash}(\mathsf{hk}, \cdot)$.

To prove statistical hiding, note that the average min-entropy [DORS08] of $r|Y_r$ is at least

$$H_\infty(r|Y_r) := \mathbb{E}_{Y \leftarrow Y_r} H_\infty(r|Y_r = Y) \geq L(\lambda) + 2\lambda - |Y_r| = L(\lambda) + \lambda \ .$$

By the generalized leftover hash lemma [DORS08],

$$(h, Y_r, z_r \oplus X) \approx_\varepsilon (h, Y_r, u) \ ,$$

for a uniformly random and independent $u \leftarrow \{0,1\}^{L(\lambda)}$ and $\varepsilon = O(\sqrt{2^{-H_\infty(r|Y_r)} \cdot 2^{L(\lambda)}}) \leq O(2^{-\lambda/2})$. □

Remark 7.2 (Shrinkage). In the above commitment scheme, the size of the commitment is proportional to the size of the plaintext. The commitment can be made shrinking by another application of the hash function over the commitment. This weakens the binding property quadratically.

8 4-Message Zero-Knowledge Proofs

In this section, we show how to use weakly-binding statistically-hiding commitments to get constant round zero-knowledge proofs with improved round complexity or assumptions. Previously, the most round-efficient zero-knowledge proof was that of Goldreich and Kahan [GK96a], which had five messages and could be based on binding (rather than weakly binding) statistically-hiding commitments.[17] Relying on keyless (weakly-binding) commitments, the constructed proof system has four messages, which crosses a previous black-box lower bound of Katz [Kat12] that shows that zero-knowledge proofs cannot be achieved in four messages if the simulator access the verifier as a black box. Indeed, our simulator will only be semi black-box — it will use the verifier as a black-box, but will depend on the size of the non-uniform verifier circuit. In the keyed setting, we obtain a five-message protocol as in [GK96a], but assuming only keyed multi-collision-resistant hashing instead of standard collision-resistant hashing.

In the literature, there are several common definitions of zero knowledge against non-uniform verifiers. The classical definition asks that any non-uniform verifier has a non-uniform simulator. A stronger definition is that of *universal simulation*, which asks that there's a single uniform simulator that can simulate any non-uniform verifier, given its code as auxiliary input.

The definition we achieve is stronger than the classical one, but slightly weaker than the universal one. We show a universal simulator that is non-uniform — it uses a (universal) non-uniform advice of arbitrarily super-constant length.

Definition 8.1 (Non-Uniform Simulation)**.** *A protocol $\langle \mathsf{P} \leftrightarrows \mathsf{V} \rangle$ for an **NP** relation $\mathcal{R}(x,w)$ is a zero-knowledge proof with an ℓ-non-uniform simulator if it satisfies completeness and soundness as in Definition 2.1 and the following augmented zero knowledge requirement.*

Zero Knowledge with ℓ-Non-Uniform Simulator: *There exists a PPT simulator S and advice $z = \left\{ z_\lambda \in \{0,1\}^{\ell(\lambda)} \right\}_\lambda$ such that for every polynomial-size circuit family $\mathsf{V}^* = \{\mathsf{V}^*_\lambda\}_\lambda$:*

$$\{\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}^*_\lambda(x) \rangle\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \approx_c \{\mathsf{S}(x; \mathsf{V}^*_\lambda, z_\lambda)\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \ .$$

Alternatively, we can achieve the standard universal definition if we assume that the underlying commitment is slightly super-polynomially secure (e.g. any efficient adversary breaks it with probability at most $2^{-\log^2 \lambda}$, rather than some negligible function that can depend on the adversary).[18] We can also achieve a universal simulator if we settle for the notion of ε-zero-knowledge [DNRS03], a relaxation of zero-knowledge where the simulator gets as input a noticeable accuracy parameter ε. See discussion at the end of Section 8.2.

[17]We note that zero-knowledge arguments (that are only computationally sound) are known in four rounds.

[18]Such super-polynomially secure weakly-binding commitments can be constructed from super-polynomially secure multi-collision-resistant hashing as in the previous section.

8.1 Construction

To present the protocol, we will use the abstraction of *Sigma Protocols*, which are a public-coin three-message proof system that only give a very weak zero-knowledge guaranteed.

Definition 8.2. *A sigma protocol for an **NP** relation $\mathcal{R}$ is a three-message protocol (α, β, γ) between a prover $\Sigma.\mathsf{P}$ and a public-coin verifier $\Sigma.\mathsf{V}$, with the following properties:*

Completeness: *for any $(x, w) \in \mathcal{R}$, $\langle \Sigma.\mathsf{P}(w) \leftrightarrows \Sigma.\mathsf{V} \rangle(x) = 1$.*

Soundness: *for any $x \in \{0,1\}^\lambda \setminus \mathcal{L}$ and unbounded prover $\Sigma.\mathsf{P}^*$,*

$$\Pr\left[\langle \Sigma.\mathsf{P}^* \leftrightarrows \Sigma.\mathsf{V} \rangle(x) = 1\right] \leq \lambda^{-\omega(1)} \ .$$

Special Zero-Knowledge: *There exists a PPT simulator $\Sigma.\mathsf{S}$ such that*

$$\{(\alpha, \gamma) \leftarrow \Sigma.\mathsf{P}(x, w; \beta)\}_{\substack{(x,w)\in\mathcal{R} \\ x,\beta\in\{0,1\}^\lambda}} \approx_c \{(\alpha, \gamma) \leftarrow \Sigma.\mathsf{S}(x, \beta)\}_{\substack{(x,w)\in\mathcal{R} \\ x,\beta\in\{0,1\}^\lambda}} \ ,$$

where $\Sigma.\mathsf{P}(x, w; \beta)$ is the prover's message distribution when the verifier's challenge is fixed to β.

The next claim follows from the zero-knowledge requirement, and will be used throughout the analysis.

Claim 8.1 (First-Message Indistinguishability). *In every Σ protocol:*

$$\{\alpha \leftarrow \Sigma.\mathsf{P}_1(x, w)\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \approx_c \left\{\alpha \leftarrow \Sigma.\mathsf{S}_1(x, 0^\lambda)\right\}_{\substack{(x,w)\in\mathcal{R} \\ |x|=\lambda}} \ ,$$

where $\Sigma.\mathsf{P}_1(x, w)$ and $\Sigma.\mathsf{S}_1(x, \beta)$ are the distributions on the first message of the prover and simulator.

Proof Sketch. Note that the first prover message is computed independently of the verifier's message; namely, $\Sigma.\mathsf{P}_1(x, w) \equiv \Sigma.\mathsf{P}_1(x, w; 0^\lambda)$. The claim now follows by the zero-knowledge guarantee. □

Sigma protocols are known to follow from classical zero-knowledge proof systems such as the (parallel repetition) of the 3-coloring protocol [GMW91], based on non-interactive statistically-binding commitments. A slight relaxation where all the parties and the simulator obtain a global common random string can be obtained from one-way functions, by using Naor's commitments [Nao91]. For simplicity of exposition, we shall rely on the above formulation, without a common random string.

From Sigma Protocols to Zero-Knowledge Proofs. The proof system of Goldreich and Kahan [GK96a] can be seen as a general transformation from Sigma protocols to full-fledged zero knowledge relying on perfectly (or statistically) hiding commitments. We naturally augment their transformation by using weakly-binding statistically-hiding commitments. The resulting protocol is essentially the same, with the exception that weakly-binding commitments can be keyless, in which case we get a 4-message proof rather than a 5-message one. The main difference is in the analysis of zero knowledge. Our revised analysis will show that indeed weak binding is sufficient.

We describe the protocol in Figure 2 and then proceed to its analysis.

8.2 Analysis

Proposition 8.1. *The protocol is a zero-knowledge proof with an ℓ-non-uninform simulator, where $\ell(\cdot)$ is any super constant function.*

We now proceed to prove the proposition.

We shall rely on the following lemma that, in the keyless setting, roughly says that for any K-binding commitment, every adversary has a K-size set of values, so that it can legally open commitments to values outside this set with negligible probability. We state the lemma for the more general keyed setting, where this set may depend on the key.

Protocol 2

Common Input: an instance $x \in \mathcal{L}(\mathcal{R}) \cap \{0,1\}^{\lambda}$, for security parameter λ.

Auxiliary Input of P**:** a witness $w \in \mathcal{R}(x)$.

1. P samples a key $\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^{\lambda})$ and sends hk to V.
   In Keyless Setting: this message is not sent. $\mathsf{hk} \equiv 1^{\lambda}$ throughout the protocol.
2. V samples a random $\beta \leftarrow \{0,1\}^{\lambda}$, computes a commitment $c \leftarrow \mathsf{SHC.Com}(\beta; \mathsf{hk})$. It sends c to P and stores the commitment randomness r it used.
3. P starts emulating $\Sigma.\mathsf{P}(x, w)$, obtains a message α and sends α to V.
4. V opens the commitment by sending (β, r) to P.
5. P completes emulating $\Sigma.\mathsf{P}(x, w)$ with verifier message β, obtains message γ and sends it to V.
6. V runs $\Sigma.\mathsf{V}(x, \alpha, \beta, \gamma)$ and accepts if and only if $\Sigma.\mathsf{V}$ accepts.

Figure 2: A ZK Proof for **NP** relation $\mathcal{R}$.

Lemma 8.1 (The Heavy Set)**.** *Let* $\mathsf{SHC} = (\mathsf{SHC.Gen}, \mathsf{SHC.Com})$ *be a* K*-binding statistically-hiding commitment. For any PPT* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and polynomial* q *there exists a negligible function* μ *such that for any polynomial-size advice* $\{z_\lambda\}_{\lambda \in \mathbb{N}}$ *of size at most* $q(\lambda)$*, and letting* $K = K(\lambda, q(\lambda))$*:*

$$\Pr_{\substack{\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^{\lambda}) \\ c \leftarrow \mathcal{A}_1(z_\lambda, \mathsf{hk})}} \left[\exists \mathbf{X} \text{ of size } K : \Pr_{(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk})} [c = \mathsf{SHC.Com}(X; \mathsf{hk}, r) \wedge X \notin \mathbf{X}] \leq \mu(\lambda) \right] \geq 1 - \mu(\lambda) \ .$$

Remark 8.1. Note that in the above lemma the negligible function μ only depends on the size $q(\lambda)$ of the advice and not on the advice itself.

Proof. For any polynomial $q = q(\lambda)$, auxiliary input $z \in \{0,1\}^q$, fixed key hk, and commitment c. Let $\mathbf{X} = \mathbf{X}(z, \mathsf{hk})$ be the K elements in $\{0,1\}^{\ell}$ that $\mathcal{A}_2(z, \mathsf{hk})$ opens consistently with c with maximal probability. Say that (hk, c) is ε-good if

$$\Pr_{(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk})} [c = \mathsf{SHC.Com}(X; \mathsf{hk}, r) \wedge X \notin \mathbf{X}] \geq \varepsilon \ .$$

For any polynomial q, let

$$\varepsilon^*(\lambda) := \sup \left\{ \varepsilon \;\middle|\; \max_{z_\lambda \in \{0,1\}^q} \Pr_{\substack{\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^{\lambda}) \\ c \leftarrow \mathcal{A}_1(z_\lambda, \mathsf{hk})}} [(\mathsf{hk}, c) \text{ is } \varepsilon\text{-good}] \geq \varepsilon \right\} \ .$$

Then it is sufficient to prove that ε^* is negligible, in which case the lemma holds for any function $\mu > \varepsilon^*$. Assume toward contradiction that this is not the case, meaning that there is a noticeable function $\varepsilon = \varepsilon(\lambda)$ and $z = \{z_\lambda\}_\lambda$ such that for infinitely many $\lambda \in \mathbb{N}$:

$$\Pr_{\substack{\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^{\lambda}) \\ c \leftarrow \mathcal{A}_1(z_\lambda, \mathsf{hk})}} [(\mathsf{hk}, c) \text{ is } \varepsilon\text{-good}] \geq \varepsilon \ .$$

We'll construct a PPT adversary $\mathcal{B}$ with advice $z = \{z_\lambda\}_\lambda$ that breaks K-binding. $\mathcal{B}(z_\lambda, \mathsf{hk})$ obtains $c \leftarrow \mathcal{A}_1(z_\lambda, \mathsf{hk})$. It then takes $2K\,(4K/\varepsilon)^2$ samples $(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk})$ and outputs all (X, r) such that $\mathsf{SHC.Com}(X, r) = c$ (without repetitions). Fix any ε-good (hk, c). We show that $\mathcal{B}$ outputs K distinct (X, r) that map to c with probability at least $1/4$. This would overall imply that $\mathcal{B}$ breaks K-binding with probability at least $\varepsilon/4$.

We consider two cases. First, consider the case that

$$\min_{X\in\mathbf{X}} \Pr\left[(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk}) \wedge c = \mathsf{SHC.Com}(X; \mathsf{hk}, r)\right] \geq (\varepsilon/4K)^2 \quad .$$

Here the expected number of samples required to collect all elements in $\mathbf{X}$ is at most $K\,(4K/\varepsilon)^2$. Thus, by Markov's inequality, after $2K\,(4K/\varepsilon)^2$ samples, $\mathcal{B}$ has collected K distinct opening of c, with probability at least $1/2$.

Now, consider the case that

$$\min_{X\in\mathbf{X}} \Pr\left[(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk}) \wedge c = \mathsf{SHC.Com}(X; \mathsf{hk}, r)\right] < (\varepsilon/4K)^2 \quad .$$

Since $\mathbf{X}$ includes the elements opened with maximal density, the above implies that for any $X \notin \mathbf{X}$,

$$\Pr\left[(X, r) \leftarrow \mathcal{A}_2(z_\lambda, \mathsf{hk}) \wedge c = \mathsf{SHC.Com}(X; \mathsf{hk}, r)\right] < (\varepsilon/4K)^2 \quad . \tag{13}$$

Since (hk, c) is ε-good, we know that the expected number of samples to collect K elements $X \notin \mathbf{X}$, with repetitions, is at most K/ε. By Markov's inequality, after $2K/\varepsilon$ samples, K such elements are collected with probability at least $1/2$. By Equation 13, the probability that these elements are not distinct is at most

$$\left(\frac{2K}{\varepsilon}\right)^2 \cdot \left(\frac{\varepsilon}{4K}\right)^2 \leq 1/4 \quad .$$

Thus, in this case, $\mathcal{B}$ outputs K distinct openings of c with probability at least $1/4$. □

Proof of Proposition 8.1. As in [GK96a], the fact that the protocol is (statistically) sound follows directly from the statistical soundness of the Sigma protocol and the statistical hiding of the commitments.

Henceforth, we concentrate on proving zero knowledge. We first describe the zero-knowledge simulator and then analyze its validity and running time. Our simulator will be a black-box simulator in the code of the verifier, with the exception that it depends on the size of the verifier's circuit. First, we describe a uniform simulator that is given an accuracy parameter ε as auxiliary input, and prove that there exists a negligible function μ such that as long as $\varepsilon \geq \mu$, our simulator runs in expected polynomial time and has simulation accuracy ε. (This already yields the notion of ε-zero-knowledge.) We will then show how to obtain a non-parametrized simulator with a negligible simulation error at the expense of allowing the simulator to have small non-uniform advice (independent of the statement and verifier) or assuming weakly-binding commitments with slightly super-polynomial security.

In what follows fix a (w.l.o.g deterministic) verifier $\mathsf{V}^* = \{\mathsf{V}^*_\lambda\}_\lambda$. In what follows, $K' = \mathrm{poly}(|\mathsf{V}^*_\lambda|, \lambda)$ is a fixed polynomial that depends only on the size of the verifier V^*_λ and the security parameter λ, which is specified later in the analysis.

The Simulator $\mathsf{S}^{\mathsf{V}^*_\lambda}(x, \varepsilon)$:

1. Sample a key $\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda)$ and feed hk to V^*.
   In Keyless Setting: this step is skipped. $\mathsf{hk} \equiv 1^\lambda$ throughout.
2. Obtain a commitment c from V^*.
3. Sample a dummy first message $\alpha \leftarrow \Sigma.\mathsf{S}_1(x, 0^\lambda)$, and feed it to V.
4. If V^* successfully opens its commitment c to a challenge message β^*, store β^* and proceed to the next step. Otherwise, abort and output the transcript so far.
5. Set counters $N = 0, n = 0$. While $n < \lambda$ and $N < \overline{N} := \min\left\{\lambda^2 K' \varepsilon^{-1}, 2^{\sqrt{\lambda}}\right\}$ repeat the following:
   (a) Rewind V^* and repeat Step 2.

(b) Sample $\alpha \leftarrow \Sigma.\mathsf{S}_1(x, 0^\lambda)$, and feed α to V^*.

(c) If V^* responds with β^*, increase $n \leftarrow n + 1$.

(d) Increase $N \leftarrow N + 1$.

6. Set $\tilde{p} = n/N$.
7. Repeat the following at most $\lambda/\tilde{p}$ times:

   (a) Rewind V^* and repeat Step 2.

   (b) Sample simulated $(\alpha, \gamma) \leftarrow \Sigma.\mathsf{S}(x, \beta^*)$, and feed α to V^*.

   (c) If V^* responds with β^*, complete the simulation with γ.
8. If the latter step failed, abort and output `timeout`.

The Running Time. Consider the PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that takes as input a hash key hk and auxiliary input $z = (\mathsf{V}^*, x)$, consisting of a verifier circuit V^*, and an instance x and does the following:

- $\mathcal{A}_1(z, \mathsf{hk})$: feeds V^* with the first prover message hk and obtains a commitment c. It outputs c.
- $\mathcal{A}_2(z, \mathsf{hk})$: samples $\alpha \leftarrow \Sigma.\mathsf{S}_1(x, 0^\lambda)$, feeds it to V^* as the second prover message and obtains from V^* an opening (β, r) of c. It outputs (β, r).

Let $q = |z|$. Let μ be the negligible function given by Lemma 8.1 with respect to $\mathcal{A}$ and q. We prove:

Claim 8.2. *For* $\varepsilon(\lambda) \geq \Omega(\mu(\lambda))$, $\mathsf{S}^{\mathsf{V}^*_\lambda}(x, \varepsilon)$ *runs in time* $\mathrm{poly}(\lambda)$.

Proof. First, note that the simulator is consistent with the definition of the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with auxiliary input $z_\lambda = (\mathsf{V}^*_\lambda, x)$. The simulator's first step of sampling hk and obtaining c is captured by $\mathcal{A}_1$. Steps 3 and 5b are captured by $\mathcal{A}_2$. Accordingly, we can rely on Lemma 8.1, for $K = K(q(\lambda), \lambda)$:

$$\Pr_{\substack{\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda) \\ c \leftarrow \mathcal{A}_1(z_\lambda; \mathsf{hk})}} \left[\exists \mathbf{X} \text{ of size } K : \Pr_{(X, r) \leftarrow \mathcal{A}_2(z_\lambda; \mathsf{hk})} [c = \mathsf{SHC.Com}(X; \mathsf{hk}, r) \wedge X \notin \mathbf{X}] \leq \mu(\lambda) \right] \geq 1 - \mu(\lambda) \ .$$

First, we claim that we can assume that (hk, c) sampled by the simulator are such that a set $\mathbf{X}$ as above exists. Indeed, such a set does not exist with probability at most μ. Since the running-time of the simulator is at most $K'\varepsilon^{-1} \cdot \mathrm{poly}(\lambda)$, and $\varepsilon \geq \Omega(\mu)$, the contribution to the expected running time of this event is at most $\mu\varepsilon^{-1}K' \cdot \mathrm{poly}(\lambda) \leq \mathrm{poly}(\lambda)$.

Hereafter, let $\mathbf{X}$ be a set as above. Let β^* be the value that V^* opens in Step 3. To analyze the simulator's running time we consider two cases:

- $\beta^* \notin \mathbf{X}$: the probability that such an element is opened in Step 3 is at most μ, and thus (similarly to the above) the contribution of such elements to the expected running time is at most $\mu\varepsilon^{-1}K' \cdot \mathrm{poly}(\lambda) \leq \mathrm{poly}(\lambda)$.
- $\beta^* \in \mathbf{X}$: every such element is opened in Step 3 with some probability $p(\beta^*)$ and the corresponding expected number of trials in Step 5b is at most $1/p(\beta^*)$. Furthermore, by a standard tail bound, except with probability $2^{-\Omega(\lambda)}$, $\tilde{p} \geq p(\beta^*)/2$, in which case the number of trials in Step 7b is at most $2\lambda/p(\beta^*)$. In the worst-case, $\tilde{p} > 2^{-\sqrt{\lambda}}$. Overall, each such element β^* contributes at most $\mathrm{poly}(\lambda)$ to the expectation. Since $|\mathbf{X}| \leq K$, the overall contribution of such elements is at most $K \cdot \mathrm{poly}(\lambda) \leq \mathrm{poly}(\lambda)$.

This completes the run-time analysis. □

The Accuracy. We will show that

$$\langle \mathsf{P}(w) \leftrightarrows \mathsf{V}^*_\lambda \rangle \approx_{\varepsilon + \lambda^{-\omega(1)}} \mathsf{S}^{\mathsf{V}^*_\lambda}(x, \varepsilon) \ .$$

For this, we consider a hybrid simulator $\mathsf{S}'(x, w, \varepsilon)$ that is given also the witness as the input and always samples messages using $\Sigma.\mathsf{P}(x, w)$ instead of $\Sigma.\mathsf{S}$; that is:

- In Steps 3 and 5b, instead of sampling $\alpha \leftarrow \Sigma.\mathsf{S}_1(x, 0^\lambda)$, sample $\alpha \leftarrow \Sigma.\mathsf{P}_1(x, w)$.
- In Step 7b, instead of sampling $(\alpha, \gamma) \leftarrow \Sigma.\mathsf{S}(x, \beta^*)$, sample $(\alpha, \gamma) \leftarrow \Sigma.\mathsf{P}(x, w; \beta^*)$.

Claim 8.3. *For $\varepsilon \geq \Omega(\mu)$, the view generated by the hybrid simulator is indistinguishable from that generated by the original simulator:*

$$\mathsf{S}^{\mathsf{V}^*_\lambda}(x, \varepsilon) \approx_c \mathsf{S}'(x, w, \varepsilon) \ .$$

Proof Sketch. The claim follows by the special zero-knowledge and first-message indistinguishability of the sigma protocol and the fact that S runs in expected polynomial time.

Specifically, assume toward contradiction that there exists a poly-size distinguisher D that distinguishes the two with probability δ. First, note that since S runs in expected polynomial time $T = \mathrm{poly}(\lambda)$, it terminates after $3T\delta^{-1}$ steps except with probability $\delta/3$. We argue that S' terminates after $3T\delta^{-1}$ steps except with probability $\delta/3 + \lambda^{-\omega(1)}$. Otherwise, we can distinguish oracles $\Sigma.\mathsf{S}_1(x, 0^\lambda), \Sigma.\mathsf{S}(x, \cdot)$ from oracles $\Sigma.\mathsf{P}_1(x, w), \Sigma.\mathsf{P}(x, w; \cdot)$ in polynomial time $O(3T\delta^{-1})$, contradicting special zero knowledge and first-message indistinguishability.

It follows that D distinguishes with probability

$$\delta - \delta/3 - \delta/3 - \lambda^{-\omega(1)} \geq \delta/3 - \lambda^{-\omega(1)}$$

augmented versions of S and S' that are truncated after $3T\delta^{-1}$ steps. Once again, this leads to a polynomial-time distinguisher for the above two pairs of oracles. ☐

Next, consider the PPT adversary $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ that takes as input a hash key hk and auxiliary input $z = (\mathsf{V}^*, x, w)$, consisting of a verifier circuit V^*, an instance x, and a witness w, and does the following:

- $\mathcal{A}_1(z, \mathsf{hk})$: feeds V^* with the first prover message hk and obtains a commitment c. It outputs c.
- $\mathcal{A}_2(z, \mathsf{hk})$: Samples $\alpha \leftarrow \Sigma.\mathsf{P}_1(x, w)$ and feeds it to V^* as the second prover message. It obtains from V^* an opening (β, r) of c, and outputs (β, r).

Let $q' = |z|$. Let μ' be the negligible function given by Lemma 8.1 with respect to $\mathcal{A}'$ and q'. We prove:

Claim 8.4. *For $\varepsilon(\lambda) \geq \Omega(\mu'(\lambda))$,*

$$\Pr\left[\textit{simulator } \mathsf{S}' \textit{ outputs } \mathtt{timeout}\right] \leq \varepsilon + \lambda^{-\omega(1)}.$$

Proof. First, note that the simulator S' is consistent with the definition of the adversary $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ with auxiliary input $z_\lambda = (\mathsf{V}^*_\lambda, x, w)$. The simulator's first step of sampling hk and obtaining c is captured by $\mathcal{A}_1$. Steps 3,5b, 7b are captured by $\mathcal{A}_2$. Accordingly, we can rely on Lemma 8.1, for $K' = K'(q'(\lambda), \lambda)$:

$$\Pr_{\substack{\mathsf{hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda) \\ c \leftarrow \mathcal{A}'_1(z_\lambda; \mathsf{hk})}} \left[\exists \mathbf{X}' \text{ of size } K' : \Pr_{(X,r) \leftarrow \mathcal{A}'_2(z_\lambda; \mathsf{hk})} [c = \mathsf{SHC.Com}(X; \mathsf{hk}, r) \wedge X \notin \mathbf{X}'] \leq \mu'(\lambda)\right] \geq 1 - \mu'(\lambda) \ .$$

We claim that we can assume that (hk, c) sampled by the simulator are such that a set $\mathbf{X}'$ as above exists. Indeed, such a set does not exist with probability at most μ', and thus this has at most negligible affect on the probability of the $\mathtt{timeout}$ event.

Hereafter, let $\mathbf{X}'$ be a set as above. Let β^* be the value that V^* opens in Step 3. To analyze the simulation accuracy we consider two cases:

- $\beta^* \notin \mathbf{X}'$: the probability that such an element is opened in Step Step 3 is at most μ', and thus this event has only a negligible effect on the probability of a $\mathtt{timeout}$ event.
- $\beta^* \in \mathbf{X}'$: here we consider two subcases according to the probability $p(\beta^*)$ that β^* is opened:
  - $p(\beta^*) \geq \lambda^2 \overline{N}^{-1}$: in this case the estimation loop computing $\tilde{p}$, by waiting for λ samples of β^* ends successfully, except with probability $2^{-\Omega(\lambda)}$. In this case, by a standard tail bound, $\tilde{p} \leq 2p(\beta^*)$, except with probability $2^{-\Omega(\lambda)}$. In this case, the loop in Step 7b iterates at least $\lambda/2p(\beta^*)$ times and ends without outputting $\mathtt{timeout}$ with probability $1 - 2^{-\Omega(\lambda)}$. Overall, $\mathtt{timeout}$ is output with probability at most $2^{-\Omega(\lambda)}$.

– $p(\beta^*) < \lambda^2 \overline{N}^{-1}$: since $|\mathbf{X}'| \leq K'$, the overall density of such elements β^* is at most $K'\lambda^2\overline{N}^{-1} \leq K'\lambda^2 \max\left\{\varepsilon/\lambda^2 K', 2^{-\sqrt{\lambda}}\right\}$. Thus, this event affects the probability of `timeout` by at most $\varepsilon + \lambda^{-\omega(1)}$.

This completes the proof of Claim 8.4. □

To complete the accuracy analysis, consider an augmented version $\mathsf{S}''(x,w)$ of S' that repeatedly performs Step 7b until success (rather than aborting after $\lambda/\tilde{p}$ failed attempts (this simulator does not depend on ε). By the last claim, the statistical distance between the views generated by these two simulators is at most $\varepsilon + \lambda^{-\omega(1)}$:

$$\mathsf{S}'(x,w,\varepsilon) \approx_{\varepsilon+\lambda^{-\omega(1)}} \mathsf{S}''(x,w) \enspace .$$

It is left to note that the view generated by S'' is identically distributed to the view generated in an interaction:

$$\mathsf{S}''(x,w) \equiv \langle \mathsf{P}(w) \leftrightarrows \mathsf{V}^*_\lambda \rangle(x) \enspace .$$

Indeed, S'' first samples β^* from the same distribution as in the real protocol, and then samples (α,γ) from this distribution conditioned on β^*, by rejection sampling.

This concludes the accuracy analysis. □

Deducing ε-Zero-Knowledge and Non-Parameterized Zero-Knowledge. The above analysis shows that for any function $\varepsilon(\lambda) \geq \mu(\lambda) + \mu'(\lambda)$, we get a simulator of expected polynomial running time and accuracy ε. In particular, this holds for any noticeable $\varepsilon = \lambda^{-O(1)}$ and implies the notion of ε-zero-knowledge.

To obtain, standard (non-parameterized) zero knowledge, the simulator should be able to compute a negligible upper bound $\lambda^{-\omega(1)} \geq u(\lambda) \geq \mu(\lambda) + \mu'(\lambda)$, and can then set $\varepsilon = u$. The functions μ and μ' only depend on (a polynomial bound on) the size of the verifier, but may generally not be efficiently computable, and accordingly such an upper bound u may not be efficiently computable.

Such an upper bound can be efficiently computed with a small non-uniform advice. That is, letting $\alpha(\lambda)$ be any super-constant function, we can keep a table M_λ that maps any constant c to a value $M_\lambda(c) \in [\alpha(\lambda)]$ such that if the size of the verifier is bounded by λ^c then $u_c(\lambda) = \lambda^{-M_\lambda(c)}$ is an upper bound on $\mu_c(\lambda) + \mu'_c(\lambda)$, where μ_c, μ'_c are the corresponding negligible functions, and $u_c(\lambda)$ is a negligible function. For instance, choose $M_\lambda(c) = \min\{\alpha(\lambda), -\log(\mu_c(\lambda) + \mu'_c(\lambda))\}$. Thus, arbitrarily super-constant length advice is sufficient.

Alternatively, we could avoid the advice altogether, if the underlying commitment is such that we can efficiently compute the negligible probability that an adversary of a give size fails (e.g., if we're guaranteed that any efficient adversary breaks K-binding with probability at most $2^{-\log^2\lambda}$).

Acknowledgements

We thank Zvika Brakerski, Ran Raz, and Vinod Vaikuntanathan for their collaboration in the early stage of this project. We also thank Benny Applebaum, Irit Dinur, John Steinberger, Avi Wigderson, and Ryan Williams for valuable discussions. We are grateful to Oded Goldreich for valuable discussions and suggestions regarding the presentation of our results.

References

[AM13] Benny Applebaum and Yoni Moses. Locally computable UOWHF with linear shrinkage. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 486–502, 2013.

[App16] Benny Applebaum. Cryptographic hardness of random local functions - survey. *Computational Complexity*, 25(3):667–722, 2016.

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 106–115, 2001.

[BBK$^+$16] Nir Bitansky, Zvika Brakerski, Yael Tauman Kalai, Omer Paneth, and Vinod Vaikuntanathan. 3-message zero knowledge against human ignorance. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, pages 57–83, 2016.

[BCC$^+$14] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. The hunting of the SNARK. *IACR Cryptology ePrint Archive*, 2014:580, 2014.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for snarks and proof-carrying data. In *STOC*, pages 111–120, 2013.

[BCPR14] Nir Bitansky, Ran Canetti, Omer Paneth, and Alon Rosen. On the existence of extractable one-way functions. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 505–514, 2014.

[BDRV17] Itay Berman, Akshay Degwekar, Ron D. Rothblum, and Prashant Nalini Vasudevan. Multi collision resistant hash functions and their applications. Cryptology ePrint Archive, Report 2017/489, 2017. http://eprint.iacr.org/2017/489.

[BEG$^+$94] Manuel Blum, William S. Evans, Peter Gemmell, Sampath Kannan, and Moni Naor. Checking the correctness of memories. *Algorithmica*, 12(2/3):225–244, 1994.

[BG08] Boaz Barak and Oded Goldreich. Universal arguments and their applications. *SIAM J. Comput.*, 38(5):1661–1694, 2008.

[BGGL01] Boaz Barak, Oded Goldreich, Shafi Goldwasser, and Yehuda Lindell. Resettably-sound zero-knowledge and its applications. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 116–125, 2001.

[BJY97] Mihir Bellare, Markus Jakobsson, and Moti Yung. Round-optimal zero-knowledge arguments based on any one-way function. In *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, pages 280–305, 1997.

[BLV03] Boaz Barak, Yehuda Lindell, and Salil P. Vadhan. Lower bounds for non-black-box zero knowledge. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 384–393, 2003.

[BP04] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In *Proceedings of the 24th Annual International Cryptology Conference*, pages 273–289, 2004.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 320–335, 2002.

[CD09] Ran Canetti and Ronny Ramzi Dakdouk. Towards a theory of extractable functions. In *TCC*, pages 595–613, 2009.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[CKLR11] Kai-Min Chung, Yael Tauman Kalai, Feng-Hao Liu, and Ran Raz. Memory delegation. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 151–168, 2011.

[Dam89] Ivan Damgård. A design principle for hash functions. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, pages 416–427, 1989.

[DCL08] Giovanni Di Crescenzo and Helger Lipmaa. Succinct NP proofs from an extractability assumption. In *Proceedings of the 4th Conference on Computability in Europe*, pages 175–185, 2008.

[DN07] Cynthia Dwork and Moni Naor. Zaps and their applications. *SIAM J. Comput.*, 36(6):1513–1543, 2007.

[DNRS03] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. *J. ACM*, 50(6):852–921, 2003.

[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.

[DPP93] Ivan Damgård, Torben P. Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. In *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, pages 250–265, 1993.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[DS11] Yevgeniy Dodis and John P. Steinberger. Domain extension for macs beyond the birthday barrier. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, pages 323–342, 2011.

[FGJ18] Nils Fleischhacker, Vipul Goyal, and Abhishek Jain. On the existence of three round zero-knowledge proofs. *IACR Cryptology ePrint Archive*, 2018:167, 2018.

[FS89] Uriel Feige and Adi Shamir. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO*, pages 526–544, 1989.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178, 2009.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. *i*-hop homomorphic encryption and rerandomizable yao circuits. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, pages 155–172, 2010.

[GI01] Venkatesan Guruswami and Piotr Indyk. Expander-based constructions of efficiently decodable codes. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 658–667, 2001.

[GK96a] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *J. Cryptology*, 9(3):167–190, 1996.

[GK96b] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM J. Comput.*, 25(1):169–192, 1996.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009.

[HIOS15] Iftach Haitner, Yuval Ishai, Eran Omri, and Ronen Shaltiel. Parallel hashing via list recoverability. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 173–190, 2015.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 201–215, 1996.

[HRVW09] Iftach Haitner, Omer Reingold, Salil P. Vadhan, and Hoeteck Wee. Inaccessible entropy. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 611–620, 2009.

[HT98] Satoshi Hada and Toshiaki Tanaka. On the existence of 3-round zero-knowledge protocols. In *Proceedings of the 18th Annual International Cryptology Conference*, pages 408–423, 1998.

[HV16] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 397–429, 2016.

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 230–235, 1989.

[Jou04] Antoine Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 306–316, 2004.

[Kat12] Jonathan Katz. Which languages have 4-round zero-knowledge proofs? *J. Cryptology*, 25(1):41–56, 2012.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732, 1992.

[Kil94] Joe Kilian. On the complexity of bounded-interaction and noninteractive zero-knowledge proofs. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 466–477, 1994.

[KNY17a] Ilan Komargodski, Moni Naor, and Eylon Yogev. Collision resistant hashing for paranoids: Dealing with multiple collisions. Cryptology ePrint Archive, Report 2017/486, 2017. http://eprint.iacr.org/2017/486.

[KNY17b] Ilan Komargodski, Moni Naor, and Eylon Yogev. White-box vs. black-box complexity of search problems: Ramsey and graph property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:15, 2017.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive PCP. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 536–547, 2008.

[KRR14] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 485–494, 2014.

[KRR17] Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. From obfuscation to the security of fiat-shamir for proofs. In *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, pages 224–251, 2017.

[Lev87] Leonid A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[LS90a] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *CRYPTO*, pages 353–365, 1990.

[LS90b] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, pages 353–365, 1990.

[Mer89] Ralph C. Merkle. A certified digital signature. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, pages 218–238, 1989.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.

[MP91] Nimrod Megiddo and Christos H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.*, 81(2):317–324, 1991.

[MT07] Ueli M. Maurer and Stefano Tessaro. Domain extension of public random functions: Beyond the birthday barrier. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 187–204, 2007.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, pages 96–109, 2003.

[OV12] Rafail Ostrovsky and Ivan Visconti. Simultaneous resettability from collision resistance. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:164, 2012.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, pages 160–176, 2003.

[Rog06] Phillip Rogaway. Formalizing human ignorance. In *Progressin Cryptology - VIETCRYPT 2006, First International Conferenceon Cryptology in Vietnam, Hanoi, Vietnam, September 25-28, 2006, Revised Selected Papers*, pages 211–228, 2006.

[Unr07] Dominique Unruh. Random oracles and auxiliary input. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 205–223, 2007.

A Multi-collision Resistance in the Auxiliary-Input Random Oracle Model

In this appendix, we show that multi-collision-resistant hashing (with very good parameters) exists in hte model of *random oracles with auxiliary inputs* of Unruh [Unr07]. In this model, the adversary $\mathcal{A}$ consists of two parts $(\mathcal{A}_1, \mathcal{A}_2)$. First $\mathcal{A}_1(\mathcal{R})$, who is completely unbounded, obtains a full description of a (shrinking) random oracle $\mathcal{R}$ and outputs some (short) auxiliary information z about the random oracle. Then at the second stage $\mathcal{A}_2^{\mathcal{R}}(z)$ obtains this auxiliary input, as well as oracle access to $\mathcal{R}$, and attempts to output an K-collision in $\mathcal{R}$. We show that $\mathcal{A}$ cannot output collisions that are significantly larger than the size of the auxiliary input. Specifically, a random oracle is (K, γ)-collision resistant for $K(\lambda, \zeta) = O(\zeta/\lambda)$ and $\gamma(\lambda) = 2^{(1-\Omega(1))\lambda}$.

Proposition A.1. *For $\ell > \lambda$, let $\mathcal{R}$ denote a random function from the set of functions $\{0,1\}^\ell \to \{0,1\}^\lambda$. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2^{(\cdot)})$, where $\mathcal{A}_1$ is an unbounded algorithm that outputs $z \in \{0,1\}^\zeta$, and $\mathcal{A}_2^{(\cdot)}$ is an unbounded algorithm that makes T oracle queries and outputs $(X_1, \dots, X_K) \in \{0,1\}^{\ell \times K}$. Then*

$$\Pr_{\mathcal{R},\mathcal{A}}\left[\begin{array}{c} \forall i \neq j : X_i \neq X_j \\ \mathcal{R}(X_1) = \cdots = \mathcal{R}(X_K) \end{array} \middle| \begin{array}{r} z \leftarrow \mathcal{A}_1(\mathcal{R}) \\ (X_1, \dots, X_K) \leftarrow \mathcal{A}_2^{\mathcal{R}}(z) \end{array}\right] \leq 2^{-K(\lambda - \log T) + \zeta + \lambda} .$$

Proof. We assume w.l.o.g that $\mathcal{A}$ is deterministic and that the oracle queries made by $\mathcal{A}_2$ are always distinct. Let $\mathcal{S}$ be the set of oracles $\mathcal{R}$ for which $\mathcal{A}$ successfully finds a K-collision. We show that

$$|\mathcal{S}| \leq 2^{\lambda \cdot 2^\ell - K(\lambda - \log T) + \zeta + \lambda} ,$$

which suffices as the total number of oracles $\mathcal{R}$ is $2^{\lambda \cdot 2^\ell}$.

Concretely, we show how to uniquely represent any $\mathcal{R} \in \mathcal{S}$ using $\lambda \cdot 2^\ell - K(\lambda - \log T) + \zeta + \lambda$ bits. Fix any such $\mathcal{R} \in \mathcal{S}$, and consider a corresponding execution of $\mathcal{A}$. Let z be the resulting auxiliary input, let $\mathbf{X} = \{X_1, \dots, X_K\}$ be the resulting multi-collision, and let $\mathbf{Q} = \{Q_1, \dots, Q_T\}$ be the set of oracle queries that $\mathcal{A}_2^{\mathcal{R}}(z)$ makes. We represent $\mathcal{R}$ as follows:

- z,
- $L_{\mathbf{Q} \cap \mathbf{X}} = (i \in [T] \mid Q_i \in \mathbf{X})$,
- $\mathcal{R}(X_1)$,
- $L_{\mathbf{Q} \setminus \mathbf{X}} := (\mathcal{R}(Q_i) \mid Q_i \notin \mathbf{X})$,
- $L_{\overline{\mathbf{Q} \cup \mathbf{X}}} := (\mathcal{R}(X) \mid X \notin \mathbf{X} \cup \mathbf{Q})$.

Note that the auxiliary input size is ζ, the set $L_{\mathbf{Q} \cap \mathbf{X}}$ can be represented by at most $|\mathbf{X}| \cdot \log T$, the value $\mathcal{R}(X_1)$ by λ bits, and the last two sets $L_{\mathbf{Q} \setminus \mathbf{X}}$ and $L_{\overline{\mathbf{Q} \cup \mathbf{X}}}$ by $\lambda \cdot (2^\ell - |\mathbf{X}|)$ bits (together). In sum, this representation costs $\lambda \cdot 2^\ell - K(\lambda - \log T) + \zeta + \lambda$ as required.

To see that this representation is unique, note that it allows to reconstruct $\mathcal{R}$ as follows. First emulate $\mathcal{A}_2^{\mathcal{R}}(z)$. When it makes its ith query Q_i, if $i \in L_{\mathbf{Q} \cap \mathbf{X}}$, answer with $\mathcal{R}(X_1)$. Otherwise answer from $L_{\mathbf{Q} \setminus \mathbf{X}}$, and keep track of the current location in the list. Finally, obtain all of $\mathbf{X}$. At this point, we have all the pairs $(X, \mathcal{R}(X))$ such that $X \in \mathbf{Q} \cup \mathbf{X}$, and we can complete $L_{\overline{\mathbf{Q} \cup \mathbf{X}}}$ to a full description of the function $\mathcal{R}$. □

B Construction of 3-Round Zero-Knowledge Argument

In this appendix, and for the sake of completeness, we provide the details of the 3-message zero-knowledge argument from Section 6, which are taken almost verbatim from [BBK+16].

B.1 Witness Indistinguishability with First-Message-Dependent Instances

We define 3-message WI proofs of knowledge where the choice of statement and witness may depend on the first message in the protocol. In particular, the first message is generated independently of the statement and witness. Also, while we do allow the content of the message to depend on the length ℓ of the statement, the message length should be of fixed to λ (this allows to also deal with statements of length $\ell > \lambda$). The former requirement was formulated in several previous works (see, e.g., [HV16]) and the latter requirement was defined in [BCPR14].

Definition B.1 (WIPOK with first-message-dependent instances)**.** *Let $\langle \mathsf{P} \leftrightarrows \mathsf{V} \rangle$ be a 3-message argument for $\mathcal{L}$ with messages $(\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)$; we say that it is a WIPOK with first-message-dependent instances if it satisfies:*

1. **Completeness with first-message-dependent instances:** *For any instance choosing function X, and $\ell, \lambda \in \mathbb{N}$,*

$$\Pr\left[\mathsf{V}(x, \mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3; r') = 1 \;\middle|\; \begin{array}{r} \mathsf{wi}_1 \leftarrow \mathsf{P}(1^\lambda, \ell; r) \\ (x, w) \leftarrow X(\mathsf{wi}_1) \\ x \in \mathcal{L}, w \in \mathcal{R}_{\mathcal{L}}(x) \\ \mathsf{wi}_2 \leftarrow \mathsf{V}(\ell, \mathsf{wi}_1; r') \\ \mathsf{wi}_3 \leftarrow \mathsf{P}(x, w, \mathsf{wi}_1, \mathsf{wi}_2; r) \end{array}\right] = 1 \;\;,$$

where $r, r' \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)}$ are the randomness used by P *and* V.

The honest prover's first message wi_1 *is of length λ, independent of the length ℓ of the statement x.*

2. **Adaptive witness-indistinguishability:** *For any polynomial $\ell(\cdot)$, non-uniform PPT verifier* $\mathsf{V}^* = \{\mathsf{V}^*_\lambda\}_{\lambda \in \mathbb{N}}$ *and all $\lambda \in \mathbb{N}$:*

$$\Pr\left[\mathsf{V}^*_\lambda(x, \mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) = b \;\middle|\; \begin{array}{r} \mathsf{wi}_1 \leftarrow \mathsf{P}(1^\lambda, \ell(\lambda); r) \\ x, w_0, w_1, \mathsf{wi}_2 \leftarrow \mathsf{V}^*_\lambda(\mathsf{wi}_1) \\ \mathsf{wi}_3 \leftarrow \mathsf{P}(x, w_b, \mathsf{wi}_1, \mathsf{wi}_2; r) \end{array}\right] \leq \frac{1}{2} + \mathrm{negl}(\lambda) \;\;,$$

where $b \leftarrow \{0,1\}$, $r \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)}$ is the randomness used by P*, $x \in \mathcal{L} \cap \{0,1\}^{\ell(\lambda)}$ and $w_0, w_1 \in \mathcal{R}_{\mathcal{L}}(x)$.*

3. **Adaptive proof of knowledge:** *there is a uniform PPT extractor $\mathcal{E}$ such that for any polynomial $\ell(\cdot)$, all large enough $\lambda \in \mathbb{N}$, and any deterministic prover* P^**:*

$$\textit{if}\; \Pr\left[\mathsf{V}(\mathsf{tr}; r') = 1 \;\middle|\; \begin{array}{r} \mathsf{wi}_1 \leftarrow \mathsf{P}^* \\ \mathsf{wi}_2 \leftarrow \mathsf{V}(\ell(\lambda), \mathsf{wi}_1; r') \\ x, \mathsf{wi}_3 \leftarrow \mathsf{P}^*(\mathsf{wi}_1, \mathsf{wi}_2) \\ \mathsf{tr} = (x, \mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \end{array}\right] \geq \varepsilon \;\;,$$

$$\textit{then}\; \Pr\left[\begin{array}{l} \mathsf{V}(\mathsf{tr}; r') = 1 \\ w \leftarrow \mathcal{E}^{\mathsf{P}^*}(1^{1/\varepsilon}, \mathsf{tr}) \\ w \notin \mathcal{R}_{\mathcal{L}}(x) \end{array} \;\middle|\; \begin{array}{l} \mathsf{wi}_1 \leftarrow \mathsf{P}^* \\ \mathsf{wi}_2 \leftarrow \mathsf{V}(\ell(\lambda), \mathsf{wi}_1; r') \\ x, \mathsf{wi}_3 \leftarrow \mathsf{P}^*(\mathsf{wi}_1, \mathsf{wi}_2) \\ \mathsf{tr} = (x, \mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \end{array}\right] \leq \mathrm{negl}(\lambda) \;\;,$$

where $x \in \{0,1\}^{\ell(\lambda)}$, and $r' \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)}$ is the randomness used by V.

Instantiation. Protocols with first-message-dependent instances follow directly from the WIPOK protocol constructed in [BCPR14], assuming ZAPs and non-interactive commitments (there, the first message is taken from a fixed distribution that is completely independent of the instance).

Next, we sketch how such a protocol can be constructed without ZAPs, but assuming keyless multi-collision-resistant hash functions.

The Lapidot-Shamir protocol. As observed in [OV12], the Lapidot-Shamir variant of the 3-message (honest verifier) zero-knowledge protocol for Hamiltonicity [LS90b] is such that the first and second messages only depend on the size of the instance $|x| = \ell$, but not on the instance and witness themselves. The protocol, in particular, supports instances up to size ℓ that depend on the prover's first message. However, the size of the first message wi_1 in the protocol is $|\mathsf{wi}_1| > \ell$. We, on the other hand, would like to allow the instance x to be of an arbitrary polynomial size in $|\mathsf{wi}_1|$, and in particular such that $|\mathsf{wi}_1| < \ell$.

We now sketch a simple transformation from any such protocol where, in addition, the verifier's message is independent of the first prover message, into a protocol that satisfies the required first-message dependence of instances. Indeed, the verifier message in the Lapidot-Shamir protocol is simply a uniformly random string, and hence the transformation can be applied here.

The Transformation. Let $\ell(\lambda) > \lambda$ be any polynomial function and let $\mathcal{H}$ be a keyless multi-collision-resistant hash function from $\{0,1\}^{\ell(\lambda)}$ to $\{0,1\}^\lambda$. In the new protocol $(\mathsf{P}_{\mathsf{new}}, \mathsf{V}_{\mathsf{new}})$, the prover computes the first message mes_1 for instances of length $\ell(\lambda)$. Then, rather than sending mes_1 in the clear, the prover $\mathsf{P}_{\mathsf{new}}$ sends $y = \mathcal{H}_\lambda(\mathsf{mes}_1) \in \{0,1\}^\lambda$.

The verifier proceeds as in the previous protocol (P, V) (note that mes_1 is not required for it to compute mes_2). Finally the prover $\mathsf{P}_{\mathsf{new}}$ answers as in the original protocol, and also sends mes_1 in the clear. The verifier $\mathsf{V}_{\mathsf{new}}$ accepts, if it would in the original protocol and mes_1 is a preimage of y under $\mathcal{H}_\lambda$.

We first note that now the size of the instance ℓ can be chosen to be an arbitrary polynomial in the length $\lambda = |\mathsf{wi}_1|$ of the first WI message. In addition, we note that the protocol is still WI, as the view of the verifier $\mathsf{V}_{\mathsf{new}}$ in the new protocol can be perfectly simulated from the view of the verifier V in the old protocol, by hashing the first message on its own.

Finally, we observe that any prover $\mathsf{P}^*_{\mathsf{new}}$ that convinces the verifier in the new protocol of accepting with probability ε, can be transformed into a prover P^* that convinces the verifier of the original protocol, or to a collision-finder that finds many collisions.

B.2 1-Hop Homomorphic Encryption

A *1-hop homomorphic encryption scheme* [GHV10] allows a pair of parties to securely evaluate a function as follows: the first party encrypts an input, the second party homomorphically evaluates a function on the ciphertext, and the first party decrypts the evaluation result. (We do not require any compactness of post-evaluation ciphertexts.)

Definition B.2. *A scheme* $(\mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$*, where* $\mathsf{Enc}, \mathsf{Eval}$ *are probabilistic and* Dec *is deterministic, is a semantically-secure, circuit-private, 1-hop homomorphic encryption scheme if it satisfies the following properties:*

- **Perfect correctness:** *For any* $\lambda \in \mathbb{N}$, $x \in \{0,1\}^n$ *and circuit* C*:*

$$\Pr\left[\begin{array}{c} (\mathsf{ct}, \mathsf{sk}) \leftarrow \mathsf{Enc}(x) \\ \widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ct}, C) \\ \mathsf{Dec}_{\mathsf{sk}}(\widehat{\mathsf{ct}}) = C(x) \end{array}\right] = 1 \ .$$

  where the probability is over the coin tosses of Enc *and* Eval.

- **Semantic security:** *For any non-uniform PPT* $\mathcal{A} = \{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$*, every* $\lambda \in \mathbb{N}$*, and any pair of inputs* $x_0, x_1 \in \{0,1\}^{\mathrm{poly}(\lambda)}$ *of equal length,*

$$\Pr_{\substack{b \leftarrow \{0,1\} \\ (\mathsf{ct}, \cdot) \leftarrow \mathsf{Enc}(x_b)}} [\mathcal{A}_\lambda(\mathsf{ct}) = b] \leq \frac{1}{2} + \mathrm{negl}(\lambda) \ .$$

- **Circuit privacy:** *The randomized evaluation procedure,* Eval*, should not leak information on the input circuit* C*. This should hold even for malformed ciphertexts. Formally, let* $\mathcal{E}(x) = \mathsf{Supp}(\mathsf{Enc}(x))$ *be the set of all legal encryptions of* x*, let* $\mathcal{E}_\lambda = \cup_{x \in \{0,1\}^n} \mathcal{E}(x)$ *be the set legal encryptions for strings of length* n*, and let* $\mathcal{C}_\lambda$ *be the set of all circuits on* λ *input bits. There exists a (possibly unbounded) simulator* $\mathcal{S}_{\mathsf{1hop}}$ *such that:*

$$\{C, \mathsf{Eval}(c, C)\}_{\substack{n \in \mathbb{N}, C \in \mathcal{C}_\lambda \\ x \in \{0,1\}^n, c \in \mathcal{E}(x)}} \approx_c \left\{C, \mathcal{S}_{\mathsf{1hop}}(c, C(x), 1^{|C|})\right\}_{\substack{n \in \mathbb{N}, C \in \mathcal{C}_\lambda \\ x \in \{0,1\}^n, c \in \mathcal{E}(x)}}$$
$$\{C, \mathsf{Eval}(c, C)\}_{\substack{n \in \mathbb{N} \\ C \in \mathcal{C}_\lambda, c \notin \mathcal{E}_\lambda}} \approx_c \left\{C, \mathcal{S}_{\mathsf{1hop}}(c, \perp, 1^{|C|})\right\}_{\substack{n \in \mathbb{N} \\ C \in \mathcal{C}_\lambda, c \notin \mathcal{E}_\lambda}} \ .$$

B.3 A 3-Round Zero-Knowledge Argument

Ingredients and notation:

- A two-message weak memory delegation scheme $(\mathsf{MD.Gen}, \mathsf{MD.Mem}, \mathsf{MD.Query}, \mathsf{MD.Prove}, \mathsf{MD.Ver})$ for γ-bounded computations.
- A semantically secure and circuit-private, 1-hop homomorphic encryption scheme $(\mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ as in Definition B.2.

- A 3-message WIPOK for **NP** with first-message-dependent instances as in Definition B.1. We denote its messages by $(\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)$.
- A non-interactive perfectly-binding commitment scheme Com.
- For some $\mathsf{wi}_1, \mathsf{cmt}$, denote by $\mathcal{M}_{\mathsf{wi}_1,\mathsf{cmt}}$ a Turing machine that given memory $D = \mathsf{V}^*$ parses V^* as a Turing machine, runs V^* on input $(\mathsf{wi}_1, \mathsf{cmt})$, parses the result as $(u, \mathsf{wi}_2, \mathsf{q}, \widehat{\mathsf{ct}}_\tau)$, and outputs u.
- Denote by $\mathcal{V}_{\mathsf{param}}$ a circuit that has the string param hard-coded and operates as follows. Given as input a verification state vst for the delegation scheme:
  - parse $\mathsf{param} = (\mathsf{wi}_1, \mathsf{cmt}, \mathsf{q}, u, \mathsf{dig}, t, \pi)$,
  - return 1 ("accept") if either of the following occurs:
    - the delegation verifier accepts: $\mathsf{MD.Ver}(\mathsf{dig}, \mathcal{M}_{\mathsf{wi}_1,\mathsf{cmt}}, t, u, \mathsf{vst}, \pi) = 1$,
    - the query is inconsistent: $(\mathsf{q}, \mathsf{vst}) \notin \mathsf{MD.Query}(1^\lambda)$.[19]

  In words, $\mathcal{V}_{\mathsf{param}}$, given the verification state vst, first verifies the proof π that "$\mathcal{M}_{\mathsf{wi}_1,\mathsf{cmt}}(D) = (u, \cdots)$" where D is the database corresponding to the digest dig. In addition, it verifies that q is truly consistent with the coins vst. If the query is consistent, but the proof is rejected $\mathcal{V}_{\mathsf{param}}$ also rejects.
- Denote by $\mathbf{1}$ a circuit of the same size as $\mathcal{V}_{\mathsf{param}}$ that always returns 1.

We describe our 3-round zero-knowledge protocol in Figure 3.

C Achieving Local Opening Generically in Fully-Binding Commitments

In this section, we describe a transformation from any commitment scheme to a new one that supports local opening. That is, where it is possible to succulently open any bit of the committed string without opening the entire commitment. The transformation adds at most two messages to the commit phase of the original commitment and increases its communication by a fixed polynomial number of bits independent of the length of the committed string. The round complexity of the commitment's opening phase remains unchanged. The transformation only assumes a keyed family of multi-collision resistant hash functions.

Applied to the 4-message commitments of [KNY17a], we get a statistically hiding 5-message shrinking commitment with local opening for all polynomial size strings from keyed multi-collision resistance. The existence of such commitments was left open by the work of [KNY17a].

C.1 Overview

The basic idea behind the transformation is to use public-coin succinct arguments of knowledge for **NP** based on keyed multi-collision-resistance (See Section 5). Specifically, instead of fully opening the commitment, the sender can reveal a specific bit of the committed string, and provide a succinct proof of knowledge that this bit is consistent with the commitment.

Since the arguments are interactive, a naive implementation of this idea would add interaction to the commitment's opening phase which may be undesirable in some settings. Instead, we show how to move this interaction to the commitment phase. During the commitment phase, the location of the bit that should be opened is not yet specified; therefore, we execute multiple arguments in parallel, one for every possible location. To avoid a blowup in communication, the receiver will use the same random coins in all parallel executions and the prover will compress its communication by hashing its messages in every round with a hash tree based on the multi-collision hash (see Section 4.1).

In the commitment's opening phase, when the location to open is specified, the sender will locally open the execution that argues about the specified location. It is possible to differ the opening of the provers messages to the

[19] We can assume without loss of generality that vst consists of the random coins of $\mathsf{MD.Query}$ and thus can assume that it is easy to test if q is consistent with vst or not.

Protocol 3

Common Input: an instance $x \in \mathcal{L} \cap \{0,1\}^\lambda$, for security parameter λ.

P**:** a witness $w \in \mathcal{R}_\mathcal{L}(x)$.

1. P computes
   - wi_1, the first message of the WIPOK for statements of length $\ell_\Psi(\lambda)$, where ℓ_Ψ is the length of the statement Ψ defined in Step 3 below,
   - $\mathsf{cmt} \leftarrow \mathsf{Com}(0^\lambda, 0^{\log\gamma(\lambda)})$, a commitment to the all zero string,

   and sends $(\mathsf{wi}_1, \mathsf{cmt})$.

2. V computes
   - wi_2, the second message of the WIPOK.
   - $(\mathsf{q}, \mathsf{vst}) \leftarrow \mathsf{MD.Query}(1^\lambda)$, where we assume w.l.o.g that vst consists of the coins of $\mathsf{MD.Query}$.
   - $(\mathsf{ct}_\mathsf{vst}, \mathsf{sk}) \leftarrow \mathsf{Enc}_\mathsf{sk}(\mathsf{vst})$, an encryption of the verification state,
   - $u \leftarrow \{0,1\}^\lambda$, a uniformly random string,

   and sends $(u, \mathsf{wi}_2, \mathsf{q}, \mathsf{ct}_\mathsf{vst})$.

3. P computes
   - $\widehat{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathbf{1}, \mathsf{ct}_\mathsf{vst})$, an evaluation of the constant one function,
   - wi_3, the third WIPOK message for the statement
     $\Psi = \Psi_1(x) \vee \Psi_2(\mathsf{wi}_1, \mathsf{cmt}, \mathsf{q}, u, \mathsf{ct}_\mathsf{vst}, \widehat{\mathsf{ct}})$ of length $\ell_\Psi(\lambda)$ given by:

$$\left\{ \exists w \;\middle|\; (x,w) \in \mathcal{R}_\mathcal{L} \right\} \bigvee$$
$$\left\{ \exists \begin{array}{l} \mathsf{dig}, \pi, r_\mathsf{cmt} \in \{0,1\}^\lambda \\ t \le \gamma(\lambda) \end{array} \;\middle|\; \begin{array}{l} \mathsf{cmt} = \mathsf{Com}(\mathsf{dig}, t; r_\mathsf{cmt}) \\ \mathsf{param} = (\mathsf{wi}_1, \mathsf{cmt}, \mathsf{q}, u, \mathsf{dig}, t, \pi) \\ \widehat{\mathsf{ct}} = \mathsf{Eval}(\mathcal{V}_\mathsf{param}, \mathsf{ct}_\mathsf{vst}) \end{array} \right\} ,$$

     using the witness $w \in \mathcal{R}_\mathcal{L}(x)$ for Ψ_1,

   and sends $(\widehat{\mathsf{ct}}, \mathsf{wi}_3)$.

4. V verifies the WIPOK proof $(\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)$ for the statement Ψ and that $\mathsf{Dec}_\mathsf{sk}(\widehat{\mathsf{ct}}) = 1$.

Figure 3: A 3-message ZK argument of knowledge for **NP**.

opening phase since the arguments are public coin. Even though the hash tree only provides a local binding guarantee, and a malicious sender may be able to open every message in a polynomial number of different ways, we can still leverage the soundness of the succinct arguments taking advantage of the fact that they're constant round.[20]

To ensure that the arguments do not break the hiding property of the commitment, the sender also commits to its messages with a non-interactive weakly-binding statistically-hiding commitment based on keyed multi-collision resistance (See Section 7). As before, since the original arguments are public-coin and constant rounds, completeness and soundness are maintained.

As described, this transformation adds all of the universal argument's messages to the commitment phase, we observe that some of these messages can be sent in parallel to the first commitment phase, so that overall we add at most two messages depending on the ordering of sender receiver messages in the underlying commitment.

C.2 Interactive Commitments

We define interactive commitments and commitments with local opening.

Definition C.1 (Commitment Scheme)**.** *An interactive commitment scheme* COM *is given by a pair of interactive PPT algorithms* $(\mathsf{COM.S}, \mathsf{COM.R})$ *and a pair of (non-interactive) PPT algorithms* $(\mathsf{COM.Open}, \mathsf{COM.Ver})$ *satisfying the following requirements:*

Completeness: *For every* $\lambda \in \mathbb{N}$ *and for every string* $X \in \{0,1\}^*$

$$\Pr\left[\mathsf{COM.Ver}(X, \mathsf{C}, \pi) = 1 \;\middle|\; \begin{array}{l} r_{\mathsf{S}} \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)} \\ \mathsf{C} \leftarrow \langle \mathsf{COM.S}(X; r_{\mathsf{S}}) \leftrightarrows \mathsf{COM.R}\rangle(1^\lambda) \\ \pi \leftarrow \mathsf{COM.Open}(X, \mathsf{C}; r_{\mathsf{S}}) \end{array}\right] = 1 \ .$$

Let $\mathsf{rc} = \mathsf{rc}(\lambda, |X|)$ *be the round complexity of the interaction* $\langle \mathsf{COM.S}(X; r_{\mathsf{S}}) \leftrightarrows \mathsf{COM.R}\rangle(1^\lambda)$ *and let* $\mathsf{cc} = \mathsf{cc}(\lambda, |X|)$ *be its communication complexity.*

Hiding: *For every polynomial* $L = L(\lambda)$ *and every polynomial-size adversary* R^* *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$ *and for every pair of strings* $X_0, X_1 \in \{0,1\}^L$

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} b \leftarrow \{0,1\} \\ b' \leftarrow \langle \mathsf{COM.S}(X_b) \leftrightarrows \mathsf{R}^*\rangle(1^\lambda) \end{array}\right] \leq \frac{1}{2} + \mu(\lambda) \ .$$

If hiding holds against computationally unbounded adversaries R^**, then* COM *is said to be statistically hiding.*

Binding: *For every polynomial-size adversaries* $\mathcal{A}, \mathsf{S}^*$ *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$*,*

$$\Pr\left[\begin{array}{l} X_0 \neq X_1 \\ \mathsf{COM.Ver}(X_0, \mathsf{C}, \pi_0) = 1 \\ \mathsf{COM.Ver}(X_1, \mathsf{C}, \pi_1) = 1 \end{array} \;\middle|\; \begin{array}{l} \mathsf{C} \leftarrow \langle \mathsf{S}^* \leftrightarrows \mathsf{COM.R}\rangle(1^\lambda) \\ (X_0, X_1, \pi_0, \pi_1) \leftarrow \mathcal{A}(\mathsf{C}) \end{array}\right] \leq \mu(\lambda) \ .$$

Definition C.2 (Commitment Scheme with Local Opening)**.** *An interactive commitment scheme*

$$\mathsf{LC} = (\mathsf{LC.S}, \mathsf{LC.R}, \mathsf{LC.Open}, \mathsf{LC.Ver})$$

supports local opening if there exist local opening and verification algorithms $(\mathsf{LC.LOpen}, \mathsf{LC.LVer})$ *satisfying:*

Completeness: *For every* $\lambda, L \in \mathbb{N}$*, every string* $X \in \{0,1\}^L$*, and every index* $i \in [L]$

$$\Pr\left[\mathsf{LC.LVer}(L, X_i, i, \mathsf{C}, \pi) = 1 \;\middle|\; \begin{array}{l} r_{\mathsf{S}} \leftarrow \{0,1\}^{\mathrm{poly}(\lambda)} \\ \mathsf{C} \leftarrow \langle \mathsf{LC.S}(X; r_{\mathsf{S}}) \leftrightarrows \mathsf{LC.R}\rangle(1^\lambda) \\ \pi \leftarrow \mathsf{LC.LOpen}(X, i, \mathsf{C}; r_{\mathsf{S}}) \end{array}\right] = 1 \ .$$

Succinctness: *There exists a fixed polynomial* poly*, such that the length of the opening* π *in the above (honest) experiment is* $|\pi| = \mathrm{poly}(\lambda)$ *for any* $L \leq 2^\lambda$.

[20]This is reminiscent of reductions from public-coin protocols to resettable soundness [GK96b, BGGL01].

Binding: *For every polynomially bounded function $L = L(\lambda)$ and polynomial-size adversaries $\mathcal{A}, \mathsf{S}^*$ there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[\begin{array}{l} \mathsf{LC.LVer}(L, 0, i, \mathsf{C}, \pi_0) = 1 \\ \mathsf{LC.LVer}(L, 1, i, \mathsf{C}, \pi_1) = 1 \end{array} \middle| \begin{array}{l} \mathsf{C} \leftarrow \langle \mathsf{S}^* \leftrightarrows \mathsf{LC.R}\rangle(1^\lambda) \\ (i, \pi_0, \pi_1) \leftarrow \mathcal{A}(\mathsf{C}) \end{array}\right] \leq \mu(\lambda) \enspace .$$

C.3 Transformation

We prove the following

Theorem C.1. *Assuming a polynomially-compressing weakly K-collision-resistant hash for $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$, any commitment scheme can be transformed into a commitment scheme with local opening. Statistical hiding (if existed) is preserved. Round complexity is preserved up to at most two messages. Communication complexity is preserved up to a fixed polynomial in the security parameter.*[21]

We assume the existence of the following ingredients:

- An interactive commitment scheme with round complexity rc and communication complexity cc

$$\mathsf{Com} = (\mathsf{COM.S}, \mathsf{COM.R}, \mathsf{COM.Open}, \mathsf{COM.Ver}) \enspace .$$

- A 2δ-message succinct argument system (P, V) for the universal relation $\mathcal{R}_{\mathcal{U}}$ with proof of knowledge for computation-time bound $\bar{t}(\lambda) = \lambda^{O(1)}$ and polynomial extraction time $\tau(\lambda) = \lambda^{O(1)}$.
- A K-collision-resistant hash tree for $K(\lambda, \zeta) = \mathrm{poly}(\lambda, \zeta)$, input-length bound $\bar{L}(\lambda) = \lambda^{O(1)}$, and accuracy bound $\underline{\varepsilon}(\lambda) = \lambda^{-O(1)}$.

$$\mathsf{HT} = (\mathsf{HT.Gen}, \mathsf{HT.Hash}, \mathsf{HT.Auth}, \mathsf{HT.Ver}) \enspace .$$

- A non-interactive weakly K-binding statistically-hiding commitment $\mathsf{SHC} = (\mathsf{SHC.Gen}, \mathsf{SHC.Com})$.

Note that the hash function assumed in the theorem imply the last three primitives as shown in the previous sections (where succinct arguments with $\delta = 2$).

We construct an interactive commitment scheme with local opening

$$\mathsf{LC} = (\mathsf{LC.S}, \mathsf{LC.R}, \mathsf{LC.Open}, \mathsf{LC.Ver}, \mathsf{LC.LOpen}, \mathsf{LC.LVer}) \enspace .$$

For simplicity, we first describe a system with round complexity $\mathsf{rc} + 2\delta$ and communication complexity $\mathsf{cc} + \mathrm{poly}(\lambda)$. We then explain how to optimize round complexity to $\mathsf{rc} + 1$ or $\mathsf{rc} + 2$ given 4-message succinct arguments.

Commitment Phase. The algorithms $\mathsf{LC.S}$ and $\mathsf{LC.R}$ take as input the security parameter 1^λ, $\mathsf{LC.S}$ takes a string the $X \in \{0,1\}^L$ to be committed.

1. $\mathsf{LC.S}$ and $\mathsf{LC.R}$ emulate the interaction $\langle \mathsf{COM.S}(X) \leftrightarrows \mathsf{COM.R}\rangle(1^\lambda)$ and obtain the commitment C. Let r_{S} denote the randomness used by $\mathsf{LC.S}$ in the emulation of $\mathsf{COM.S}$.
2. $\mathsf{LC.S}$ generates an opening $\pi \leftarrow \mathsf{COM.Open}(X, \mathsf{C}; r_{\mathsf{S}})$ and stores it.
3. $\mathsf{LC.R}$ generates keys for the hash tree and the non-interactive weakly binding commitment

$$\mathsf{HT.hk} \leftarrow \mathsf{HT.Gen}(1^\lambda) \quad , \quad \mathsf{SHC.hk} \leftarrow \mathsf{SHC.Gen}(1^\lambda) \enspace ,$$

and sends them to $\mathsf{LC.S}$.

[21]As in our previous results we can rely instead of linear compression if we require quasi-polynomial security for the hash functions.

4. $\mathsf{LC.S}$ and $\mathsf{LC.R}$ emulate L encrypted parallel executions of the succinct argument (P,V). For any execution $i \in [L]$, $\mathsf{LC.S}$ proves to $\mathsf{LC.R}$ the following NP statement denoted by $\mathsf{ST}_{\mathsf{C},i,X_i}$

$$\mathsf{ST}_{\mathsf{C},i,X_i}: \quad \exists (X', \pi') : \mathsf{COM.Ver}(X', \mathsf{C}, \pi') = 1 \wedge X'_i = X_i \ .$$

Specifically, $\mathsf{LC.S}$ and $\mathsf{LC.R}$ emulate any round $j \in [\delta]$ of the argument as follows:

(a) $\mathsf{LC.R}$ sends public coins r^j_{V}. (The same coins are used for all parallel executions.)

(b) For every $i \in [L]$, $\mathsf{LC.S}$ feeds the message r^i_{V} to the prover P proving the i-th statement and obtains a message a^j_i. For simplicity we assume that $a^j_i \in \{0,1\}^\lambda$.

(c) $\mathsf{LC.S}$ computes the hash
$$\mathsf{dig}^j \leftarrow \mathsf{HT.Hash}\left(\mathsf{HT.hk}, \left(a^j_1, \ldots, a^j_L\right)\right) \ .$$

(d) $\mathsf{LC.S}$ sends $\mathsf{LC.R}$ a commitment to the digest. Let r^j_c denote the commitment's randomness.
$$c^j \leftarrow \mathsf{SHC.Com}(\mathsf{dig}^j; \mathsf{SHC.hk}, r^j_c) \ .$$

5. $\mathsf{LC.S}$ and $\mathsf{LC.R}$ output the commitment C' that consists of the original interactive commitment, the hash and non-interactive commitment keys, and the encrypted transcript of the argument
$$\mathsf{C}' = \left(\mathsf{C}, \mathsf{HT.hk}, \mathsf{SHC.hk}, \left(r^j_{\mathsf{V}}, c^j\right)_{j\in[\delta]}\right) \ .$$

Opening Phase. The (non-local) algorithms $\mathsf{LC.Open}$ and $\mathsf{LC.Ver}$ simply emulate the algorithms $\mathsf{COM.Open}$ and $\mathsf{COM.Ver}$ ignoring the messages of the interactive argument.

The local opening algorithm $\mathsf{LC.LOpen}$ is given the committed string $X \in \{0,1\}^L$, an index $i \in [L]$, the commitment C', and sender randomness r_{S}. For every $j \in [\delta]$, $\mathsf{LC.LOpen}$ reconstructs the values computed by $\mathsf{LC.S}$:
$$\left(a^j_1, \ldots, a^j_L\right) \quad , \quad \mathsf{dig}^j \quad , \quad r^j_c \ .$$

For every $j \in [\delta]$, $\mathsf{LC.LOpen}$ opens the non-interactive commitment and the i-th hash tree block
$$\pi' = \left(\left(\mathsf{dig}^j, r^j_c\right) \quad , \quad a^j_i \quad , \quad \mathsf{HT.Auth}\left(\mathsf{HT.hk}, \left(a^j_1, \ldots, a^j_L\right), i\right)\right)_{j\in[\delta]} \ .$$

The local verification algorithm $\mathsf{LC.LVer}$ is given a bit b, an index $i \in [L]$, the commitment
$$\mathsf{C}' = \left(\mathsf{C}, \mathsf{HT.hk}, \mathsf{SHC.hk}, \left(r^j_{\mathsf{V}}, c^j\right)_{j\in[\delta]}\right) \ ,$$

and the opening
$$\pi' = \left(\left(\mathsf{dig}^j, r^j_c\right) \quad , \quad a^j \quad , \quad \pi^j\right)_{j\in[\delta]} \ .$$

For every $j \in [\delta]$, it verifies that

- $c^j = \mathsf{SHC.Com}(\mathsf{dig}^j; \mathsf{SHC.hk}, r^j_c)$.
- $\mathsf{HT.Ver}(\mathsf{HT.hk}, L, \mathsf{dig}^j, i, a^j, \pi^j) = 1$.

$\mathsf{LC.LVer}$ also verifies that the verifier V accepts the transcript $\left(r^j_{\mathsf{V}}, a^j\right)_{j\in[\delta]}$ as a proof for the statement $\mathsf{ST}_{\mathsf{C},i,b}$.

Optimizing Round Complexity. We explain how the round complexity
The round complexity of the resulting commitment scheme can be further optimized as follows.

- The last message of the argument can be included in the opening and not sent during the commitment phase.

- The keys for the hash tree and non-interactive weakly-binding commitments, as well as the public verifier randomness sent in the first message of the succinct argument can be included with some receiver message (if exists) in the initial commitment.
- If in the original interactive commitment Com, the sender communicates last, then it is possible to add the first prover message of the argument to the last round of the commitment.

When the original commitment and the interactive argument both require 4 messages (as is the case for the arguments from Section 5 and commitments from [KNY17a]), the new commitment will require 5 messages (where in the last message the verifier just sends public coins).

C.4 Analysis

The efficiency properties of the new commitment follow readily from the construction. The (statistical) hiding of the new commitment follows directly from the (statistical) hiding of the original interactive commitment and the fact that all the additional messages $\mathsf{LC.S}$ sends to $\mathsf{LC.R}$ are under a statistically hiding non-interactive commitment. (If the original commitment is only computationally hiding, so is the new commitment.)

We next sketch the proof of the binding property. Assume there exist adversaries $\mathcal{A}, \mathsf{S}^*$ that break binding with noticeable probability with respect to some index $i \in [L]$. Without loss of generality, assume $\mathcal{A}, \mathsf{S}^*$ are deterministic. We construct adversaries $\mathcal{A}', \mathsf{S}'$ that break the binding of the original commitment COM with noticeable probability.

The adversary S' emulates S^* in its interaction with $\mathsf{COM.R}$. It obtains a commitment C and the state of S^* before the succinct argument. The adversary $\mathcal{A}'$ executes $\mathcal{A}$ and the residual adversary S^* to extract two different openings as follows. For every $b \in \{0,1\}$, we construct a prover P^*_b that convinces V to accept the statement $\mathsf{ST}_{\mathsf{C},i,b}$ with noticeable probability. Then we run the knowledge extractor E of the succinct argument to extract a pair of valid openings $(X_0, \pi_0), (X_1, \pi_1)$ of C such that $X_0[i] \neq X_1[i]$. This is in contradiction to the binding of the original commitment and will thus conclude the proof.

The rest of this proof sketch is dedicated to describing and analyzing the prover P^*_b. The (stateful) prover P^*_b is defined as follows. In the j-th round, P^*_b maintains in its state the interaction in the previous $j-1$ rounds $\left(r_{\mathsf{V}}^{j'}, a^{j'}\right)_{j' \in [j-1]}$. Given the verifier coins r_{V}^{j} in the j-th round, P^*_b samples coins $r_{\mathsf{V}}^{j+1}, \dots, r_{\mathsf{V}}^{\delta}$ and feeds the residual adversary S^* with the messages $r_{\mathsf{V}}^{1}, \dots, r_{\mathsf{V}}^{\delta}$. It obtains a commitment C', executes $\mathcal{A}(\mathsf{C}')$ and obtains a pair of openings (π'_0, π'_1). If $\mathsf{LC.LVer}(L, b, i, \mathsf{C}', \pi'_b)$ rejects, P^*_b aborts. Otherwise, let

$$\pi'_b = \left(\left(\mathsf{dig}^j, r_c^j\right) \quad , \quad a_{\mathcal{A}}^j \quad , \quad \pi^j \right)_{j \in [\delta]} \quad .$$

If for some $j' \in [j-1]$, $a^{j'} \neq a_{\mathcal{A}}^{j'}$, P^*_b aborts. Otherwise, P^*_b responds with the message $a_{\mathcal{A}}^j$. By the definition of the procedure $\mathsf{LC.LVer}$ we have that if P^*_b does not abort it produces an accepting proof. Therefore, it is sufficient to show that P^*_b does not abort with noticeable probability. We prove inductively that if P^*_b reaches the j-th round without aborting with noticeable probability, it will also not abort in the j-th round with with some related noticeable probability. Since the argument is constant round we deduce that P^*_b does not abort at all with noticeable probability.

Fix an index $j \in [\delta]$, a commitment C, and coins $r_{\mathsf{V}}^{1}, \dots, r_{\mathsf{V}}^{j-1}$ sent to P^*_b in the first $j-1$ rounds such that P^*_b reaches the j-th round without aborting with some noticeable probability p. Let r_{V}^{j} be the random message sent to P^*_b in the j-th round. Let $c^1, \dots, c^{j-1}$ be the commitments sent by the emulation of the residual adversary S^* given the messages $r_{\mathsf{V}}^{1}, \dots, r_{\mathsf{V}}^{j-1}$. Since S^* is deterministic, these commitments are fixed by the choice of $r_{\mathsf{V}}^{1}, \dots, r_{\mathsf{V}}^{j-1}$.

We compare the execution of P^*_b in responding to the message r_{V}^{j-1} and in responding to the message r_{V}^{j}. When responding to the message r_{V}^{j-1}, P^*_b samples random coins to send S^* in the j-th round (and in subsequent rounds). Since the message r_{V}^{j} is itself random, the view of S^* (and therefore, also of $\mathcal{A}$) is identically distributed in both executions. The only difference between the executions is that when answering the message r_{V}^{j}, P^*_b compares the $(j-1)$-st extracted message $a_{\mathcal{A}}^{j-1}$ to the $(j-1)$-st message a^{j-1} saved in its internal state. Note that P^*_b also tests that:

- $\mathsf{LC.LVer}$ accepts $\mathcal{A}$'s opening.
- $a_{\mathcal{A}}^{j'} = a^{j'}$ for every $j < j-1$.

However, these tests are performed both when responding to the message r_{V}^{j-1} and when responding to the message r_{V}^{j} and they pass with exactly the same probability p.

We argue based on the weak binding properties of the hash tree and the non-interactive commitment, that the extra consistency test between $a_{\mathcal{A}}^{j-1}$ and a^{j-1} must pass with noticeable probability. Recall that the messages $a_{\mathcal{A}}^{j-1}$ and a^{j-1} are identically distributed and with probability at least p they are both an opening of the commitment c^{j-1} (otherwise P_b^* would have aborted before the j-th round with probability larger than p). That is,

$$c^{j-1} = \mathsf{SHC.Com}(\mathsf{dig}_{\mathcal{A}}^{j-1}; \mathsf{SHC.hk}) \quad \wedge \quad \mathsf{HT.Ver}(\mathsf{HT.hk}, L, \mathsf{dig}_{\mathcal{A}}^{j-1}, i, a_{\mathcal{A}}^{j-1}, \pi_{\mathcal{A}}^{j-1}) = 1 \ ,$$
$$c^{j-1} = \mathsf{SHC.Com}(\mathsf{dig}^{j-1}; \mathsf{SHC.hk}) \quad \wedge \quad \mathsf{HT.Ver}(\mathsf{HT.hk}, L, \mathsf{dig}^{j-1}, i, a^{j-1}, \pi^{j-1}) = 1 \ .$$

It follows that there exists a set S of size K^2 such that $a_{\mathcal{A}}^{j-1}, a^{j-1} \in S$ with probability at least $p/2$. Otherwise we can break either the K-collision resistance of the hash tree or the weak K-binding of the non-interactive commitment .

Since $a_{\mathcal{A}}^{j-1}, a^{j-1}$ are identically distributed, it follows that $a_{\mathcal{A}}^{j-1} = a^{j-1}$ with probability at least $p/2K^2$. Since all other tests pass with probability at least p, we have that P_b^* does not abort in the j-th round with probability at least $p^2/2K^2$.