J Cryptol (2017) 30:989–1066
DOI: 10.1007/s00145-016-9241-9

Journal of
CRYPTOLOGY

The Hunting of the SNARK*

Nir Bitansky†
MIT, Cambridge, MA, USA
nirbitan@tau.ac.il

Ran Canetti
Boston University, Boston, MA, USA
Tel Aviv University, Tel Aviv, Israel
canetti@tau.ac.il

Alessandro Chiesa
University of California, Berkeley, CA, USA
alexch@berkeley.edu

Shafi Goldwasser
MIT, Cambridge, MA, USA
shafi@csail.mit.edu

Huijia Lin
University of California, Santa Barbara, CA, USA
achel.lin@cs.ucsb.edu

Aviad Rubinstein
University of California, Berkeley, CA, USA
aviad@eecs.berkeley.edu

Eran Tromer
Tel Aviv University, Tel Aviv, Israel
tromer@tau.ac.il

Communicated by Omer Reingold.

Received 5 August 2014 / Revised 18 April 2016
Online publication 3 October 2016

* This research was supported by the Check Point Institute for Information Security, by the Israeli Centers of Research Excellence (I-CORE) Program (Center No. 4/11), by the European Community's Seventh Framework Programme (FP7/2007-2013) Grant 240258, by a European Union Marie Curie grant, by the Israeli Science Foundation, and by the Israeli Ministry of Science and Technology.

This paper is a merge of [11] and [71]. A preliminary version including part of the results appeared in ITCS 2012.

† Research done while at Tel Aviv University.

Abstract. The existence of succinct non-interactive arguments for NP (i.e., non-interactive computationally sound proofs where the verifier's work is essentially independent of the complexity of the NP non-deterministic verifier) has been an intriguing question for the past two decades. Other than CS proofs in the random oracle model (Micali in SIAM J Comput 30(4):1253–1298, 2000), prior to our work the only existing candidate construction is based on an elaborate assumption that is tailored to a specific protocol (Di Crescenzo and Lipmaa in Proceedings of the 4th conference on computability in Europe, 2008). We formulate a general and relatively natural notion of an *extractable collision-resistant hash function (ECRH)* and show that, if ECRHs exist, then a modified version of Di Crescenzo and Lipmaa's protocol is a succinct non-interactive argument for NP. Furthermore, the modified protocol is actually a succinct non-interactive *adaptive argument of knowledge (SNARK)*. We then propose several candidate constructions for ECRHs and relaxations thereof. We demonstrate the applicability of SNARKs to various forms of delegation of computation, to succinct non-interactive zero-knowledge arguments, and to succinct two-party secure computation. Finally, we show that SNARKs essentially imply the existence of ECRHs, thus demonstrating the necessity of the assumption. Going beyond ECRHs, we formulate the notion of *extractable one-way functions* (EOWFs). Assuming the existence of a natural variant of EOWFs, we construct a two-message selective-opening-attack-secure commitment scheme and a three-round zero-knowledge argument of knowledge. Furthermore, if the EOWFs are concurrently extractable, the three-round zero-knowledge protocol is also concurrent zero knowledge. Our constructions circumvent previous black-box impossibility results regarding these protocols by relying on EOWFs as the non-black-box component in the security reductions.

1. Introduction

For the Snark's a peculiar creature, that won't
Be caught in a commonplace way.
Do all that you know, and try all that you don't:
Not a chance must be wasted to-day!
The Hunting of the Snark,
Lewis Carroll

The notion of interactive proof systems [73] is central to both modern cryptography and complexity theory. One extensively studied aspect of interactive proof systems is their expressiveness; this study culminated with the celebrated result that $\mathsf{IP} = \mathsf{PSPACE}$ [122]. Another aspect of such systems, which is the focus of this work, is that proofs for rather complex NP statements can potentially be verified much faster than by direct checking of an NP witness.

We know that if statistical soundness is required, then any non-trivial savings would cause unlikely complexity-theoretic collapses (see, e.g., [25,67,83,125]). However, if we settle for proof systems with only computational soundness (also known as interactive arguments [10]), then significant savings can be made. Indeed, using collision-resistant hash functions (CRHs), Kilian [89] shows a four-message interactive argument for NP: The prover first uses a Merkle hash tree [103] to bind itself to a polynomial-size PCP (probabilistically checkable proof) [6,7] string for the statement and then answers the PCP verifier's queries while demonstrating consistency with the Merkle tree. This way, membership of an instance y in an NP language L can be verified in time that is bounded by $p(k, |y|, \log t)$, where t is the time to evaluate the NP verification relation for L on input y, p is a fixed polynomial independent of L, and k is a security parameter that determines the soundness error. Following tradition, we call such argument systems *succinct*.

Can we have succinct argument systems that are *non-interactive*? Having posed and motivated this question, Micali [104] provides a one-message succinct non-interactive argument for NP, in the random oracle model, by applying the Fiat–Shamir paradigm [62] to Kilian's protocol. In the standard model, such "totally non-interactive" succinct arguments (against non-uniform provers) do not exist except for "quasi-trivial" languages (i.e., languages in $\mathsf{BPtime}(n^{\mathrm{polylog} n})$), because the impossibility results for statistical soundness can be directly extended to this case. Nonetheless, it may still be possible to obtain a slightly weaker notion of non-interactivity:

Definition 1.1. A $\underline{succinct non-interactive argument}$ $(SNARG)$ is a succinct argument where the verifier (or a trusted entity) generates ahead of time a succinct *verifier-generated reference string* (VGRS) and sends it to the prover. The prover can then use the VGRS to generate a succinct non-interactive proof π for a statement y of his choice. (The VGRS is thus independent of the statements to be proven later, and the definition requires "adaptive soundness," since y is chosen by the prover, potentially based on the VGRS.)

In this paper, we consider the following questions:

Can SNARGs for NP exist in the standard model?
And if so, under what assumptions can we prove their existence?

Attempted Solutions To answer the above question, Aiello et al. [1] propose to avoid Kilian's hash-then-open paradigm, and instead use a computational polylogarithmic PIR (private information retrieval) scheme [43,90] to access the PCP oracle as a long database. The verifier's first message consists of the queries of the underlying PCP verifier, encrypted using the PIR encryption algorithm. The prover applies the PIR sender algorithm to the PCP oracle, and the verifier then runs the underlying PCP verifier on the values obtained from the PIR protocol. However, Dwork et al. [56] point out that this "PCP+PIR approach" is inherently problematic, because a cheating prover could "zigzag" and answer different queries according to different databases.[1] (Natural extensions that try to force consistency by using multiple PIR instances run into trouble due to potential PIR malleability.)

Di Crescenzo and Lipmaa [49] propose to address this problem by further requiring the prover to bind itself (in the clear) to a specific database using a Merkle tree (MT) as in Kilian's protocol. Intuitively, the prover should now be forced to answer according to a single PCP string. In a sense, this "PCP+MT+PIR approach" squashes Kilian's four-message protocol down to two messages "under the PIR." However, while initially appealing, it is not a priori clear how this intuition can be turned into a proof of security under some well-defined properties of the Merkle tree hash. Indeed, to prove soundness of their protocol Di Crescenzo and Lipmaa use an assumption that is nonstandard in

[1] The problem becomes evident when implementing the PIR using fully homomorphic encryption; indeed, since any efficient adversarial strategy can be executed "under the encryption," such a solution would be as insecure as sending the PCP queries in the clear.

two main ways: First, it is a "knowledge assumption," in the sense that any adversary that generates a value of a certain form is assumed to "know" a corresponding preimage (see more discussion on such assumptions below). Furthermore, their assumption is very specific and intimately tied to the actual hash, PIR, and PCP schemes in use, as well as the language under consideration. Two other non-interactive arguments for NP, based on more concise knowledge assumptions, are due to Mie [105] and Groth [81]. However, neither of these protocols is succinct: In both protocols, the verifier's runtime is polynomially related to the time needed to directly verify the NP witness.

Recently, Gentry and Wichs [84] showed that some of the difficulty is indeed inherent by proving that no SNARG construction can be proved secure via a black-box reduction to an *efficiently falsifiable assumption* [107]. For example, the assertion that one-way functions exist or that fully homomorphic encryption exists are both falsifiable assumptions; in general, an assumption is efficiently falsifiable if it can be modeled as a game between an adversary and a challenger, where the challenger can *efficiently* decide whether the adversary has won the game. The impossibility result of Gentry and Wichs holds even for *designated-verifier* protocols, where the verifier may use secret randomness to verify. This suggests that nonstandard assumptions, such as the knowledge (extractability) assumptions described next, may be inherent.

Knowledge (extractability) Assumptions Knowledge (or extractability) assumptions capture our belief that certain computational tasks can be achieved efficiently only by (essentially) going through specific intermediate stages and thereby obtaining, along the way, some specific intermediate values. Such an assumption asserts that, for any efficient algorithm that achieves the task, there exists a *knowledge extractor* algorithm that efficiently recovers the said intermediate values.

A number of different extractability assumptions exist in the literature, most of which are specific number theoretic assumptions (such as several variants of the *knowledge of exponent* assumption [48]). It is hard to gain assurance regarding their relative strengths. Abstracting from such specific assumptions, one can formulate general notions of extractability for one-way functions and other basic primitives (see [37,47]). That is, say that a function family $\mathcal{F}$ is *extractable* if, given a random $f \leftarrow \mathcal{F}$, it is infeasible to produce $y \in \mathsf{Image}(f)$ without actually "knowing" x such that $f(x) = y$. This is expressed by saying that for any efficient adversary $\mathcal{A}$ there is an efficient extractor $\mathcal{E}_{\mathcal{A}}$ such that, if $\mathcal{A}(f) = f(x)$ for some x, then $\mathcal{E}_{\mathcal{A}}(f)$ almost always outputs x' such that $f(x') = f(x)$. Typically, for such a family to be interesting, $\mathcal{F}$ should also have some sort of hardness property, e.g., one-wayness. The assumption that a given function family is extractable is not efficiently falsifiable. In particular, the impossibility result of Gentry and Wichs [84] does not apply to such assumptions.

1.1. *Summary of Our Results*

- We formulate a general and relatively natural notion of *extractable collision-resistant hash functions* (ECRHs). We then show that if ECRHs exist then so do SNARGs for NP. Concretely, we revisit the PCP+MT+PIR approach of [49] and show that it can be modified in a way that allows us to prove (adaptive) soundness when using an ECRH to construct the Merkle tree.

Furthermore, we show that the construction is a *proof of knowledge*. We are thus able to obtain a *SNARG of knowledge* (SNARK) for NP.

- We propose a candidate ECRH construction, based on a Knowledge of Exponent assumption (that is a variant of the one found in [81]) and the hardness of taking discrete logs. We also define *proximity ECRHs*, a weaker variant of ECRHs that are still sufficient for the construction of SNARKs; we propose two candidate PECRH constructions, based on knapsack (subset sum) problems related to hard problems on lattices. Finally, we show how proximity ECRHs can be further weakened to obtain a (still sufficient) primitive for which it seems plausible to conjecture that "unstructured" hash functions such as (randomized) SHA-2 satisfy it.
- We show that existence of SNARKs for NP implies existence of (the weaker variants of) ECRHs, as well as extractable variants of some other cryptographic primitives; in other words (proximity), ECRHs are not only sufficient but also necessary for the existence of SNARKs.
- We describe some applications of SNARKs: (a) to delegation of computations (even with long delegator input and with worker input), (b) to constructing *zero-knowledge* SNARKs, and (c) for constructing succinct non-interactive secure computation (in the common reference string model). See more below.
- We consider the related notion of extractable one-way functions (EOWFs). We formulate two natural variants of EOWFs and show that as ECRHs they help circumvent previous black-box impossibility results. Namely, we construct a two-message selective-opening-attack-secure commitment scheme and a three-round public-coin concurrent zero-knowledge (ZK) argument of knowledge, which in turn implies a three-round simultaneously resettable ZK protocol (all assuming additionally the existence of enhanced trapdoor permutations). Previous works have shown that these protocols cannot be proven secure from standard assumptions using black-box reductions or simulation [41,111].

The rest of the introduction describes these results in more detail. Before proceeding, though, we wish to highlight a prominent application of SNARGs that further motivates the need for proofs of knowledge.

Delegation of Computation and Adaptive Arguments of Knowledge In the problem of delegation of computation, which has become ever more relevant with the advent of cloud computing, a client has some computational task (typically in P) and wishes to delegate the task to an untrusted worker, who responds with a claimed result along with a proof that the result is correct. Prior to our work, delegation schemes, such as [42,65,69,93,94], required either more than two messages of interaction between the client and the untrusted worker, or much work to be done by the verifier in a preprocessing stage.

A SNARG for NP could improve on these by minimizing both interaction and the verifier's computational effort. (Note that adaptive soundness of the SNARG seems crucial for this application since a cheating worker might choose a bogus result of the computation based on the delegator's first message.)[2]

[2]More precisely, this seems to be the case for delegation of non-deterministic computations where the worker contributes an input (witness) to the computation. In contrast, when delegating deterministic

However, the application to delegation schemes brings with it additional security concerns. For example, the untrusted worker may store for the delegator a long database z whose short Merkle hash $h = \mathrm{MT}(z)$ is kept by the delegator; the delegator may then ask the worker to compute $F(z)$ for some function F. However, from the delegator's perspective, merely being convinced that "there exists $\tilde{z}$ such that $h = \mathrm{MT}(\tilde{z})$ and $F(\tilde{z}) = f$" is not enough. The delegator should also be convinced that the worker *knows* such a $\tilde{z}$, which implies due to collision resistance of MTthat indeed $\tilde{z} = z$.

Thus, the delegator may not only be interested in establishing that a witness for a claimed theorem *exists*, but also want that such a witness can be *extracted* from a convincing prover. That is, we require *proof of knowledge* (or rather, an *argument of knowledge*) and thus SNARKs (rather than merely SNARGs) are needed. The ability to efficiently extract a witness for an adaptively chosen theorem seems almost essential for making use of a delegation scheme when the untrusted worker is expected to contribute its own input (such as a database, as in the above example, or a signature) to a computation.

Another application where adaptive proofs of knowledge are crucial is *recursive proof composition*, a technique that has already been shown to enable desirable cryptographic tasks, such as targeted non-malleability and proof-carrying data [13,32,39,45,124].

1.2. *ECRHs, SNARKs, and Applications*

(ii) Extractable collision-resistant hash functions. We start by defining a natural strengthening of collision-resistant hash functions (CRHs): A function ensemble $\mathcal{H} = \{\mathcal{H}_k\}_k$ is an extractable CRH (ECRH) if (a) it is collision-resistant in the standard sense and (b) it is extractable in the sense that for any efficient adversary that is able to produce a valid evaluation of the function there is an extractor that is able to produce a corresponding preimage. More precisely, extractability is defined as follows:

Definition 1. A function ensemble $\mathcal{H} = \{\mathcal{H}_k\}_k$ mapping $\{0,1\}^{\ell(k)}$ to $\{0,1\}^k$ is *extractable* if for any polynomial-size adversary $\mathcal{A}$ and polynomial m there exists a polynomial-size extractor $\mathcal{E}_{\mathcal{A}}$ such that for any security parameter $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$:

$$\Pr_{h \leftarrow \mathcal{H}_k}\left[\begin{array}{c} y \leftarrow \mathcal{A}(h,z) \\ \exists\, x : h(x) = y \end{array} \wedge \begin{array}{c} x' \leftarrow \mathcal{E}_{\mathcal{A}}(h,z) \\ h(x') \neq y \end{array}\right] \leq \mathrm{negl}(k) \ .$$

We do not require that there is an efficient way to tell whether a given string in $\{0,1\}^k$ is in the image of a given $h \in \mathcal{H}_k$. We note that:

- For extractability and collision resistance (or one-wayness) to coexist, it should be hard to "obliviously sample images"; in particular, this implies that the image of almost any $h \in \mathcal{H}_k$ should be sparse in $\{0,1\}^k$, i.e., with cardinality at most

Footnote 2 continued
computations, the delegator can use a SNARG with non-adaptive soundness by requesting a proof for each bit of the claimed output. Indeed, the statement that the ith output bit of a computation is "10" (respectively, "1") is a fixed statement, independent of the reference string. (The latter does not work for non-deterministic computations as there is no guarantee that the prover uses the same witness for each one of the output bits.)

$2^{k-\omega(\log k)}$. (This remark is a bit oversimplified and not entirely accurate; see discussion in Sect. 6.1.)

- For simplicity of exposition, the above definition accounts for the most general case of arbitrary polynomial-size auxiliary input; this is, in fact, too strong and cannot be achieved assuming indistinguishability obfuscation [16,28]. However, for our main result, we can actually settle for a relaxed definition that only considers a specific distribution over auxiliary inputs of a priori bounded size (PIR encryptions of random strings). See further discussion in Sect. 6.1.

(ii) From ECRHs to adaptive succinct arguments of knowledge, and back again. We modify the "PCP+MT+PIR" construction of [49] and show that the modified construction can be proven to be a SNARK based solely on the existence of ECRHs and PIR schemes with polylogarithmic complexity.[3]

Theorem 1. (Informal) *If there exist ECRHs and (appropriate) PIRs, then there exist SNARKs (for* NP*).*

A single VGRS in our construction suffices for only logarithmically many proofs; however, since the VGRS can be succinctly generated, the cost of occasionally resending a new one is limited.

We complement Theorem 1 by showing that ECRHs are in fact *essential* for SNARKs. More accurately, we show that SNARKs and CRHs imply a slightly relaxed notion of ECRHs that we call *proximity* ECRH*s*, and which is still sufficient for our construction of SNARKs.

Theorem 2. (Informal) *If there exist SNARKs and (standard) CRHs, then there exist proximity ECRHs.*

We defer the discussion of the details of this relaxation to Sect. 1.3.

We also show that SNARKs can be used to construct extractable variants of other cryptographic primitives. A naïve strategy to obtain this may be to "add a succinct proof of knowledge of a preimage to the output." While this strategy does not work as such because the proof may leak secret information, we show that in many cases this difficulty can be overcome by combining SNARKs with (non-extractable) *leakage-resilient* primitives [2]. For example, since CRHs and subexponentially hard defs are leakage-resilient, we obtain:

Theorem 3. (Informal) *Assume SNARKs and (standard) CRHs exist. Then there exist extractable one-way functions and extractable computationally hiding and binding commitments. Alternatively, if there exist SNARKs and (standard) subexponentially hard one-way functions, then there exist extractable one-way functions. Furthermore, if these functions are one-to-one, then we can construct perfectly binding computationally hiding extractable commitments.*

[3] More precisely, we shall require PIR schemes with polylogarithmic complexity where a fixed polynomial bound for the database size is not required by the query algorithm. See Sect. 1.4 for more details.

We believe that the approach of constructing extractable primitives based on SNARKs merits further investigation. One question, for example, is whether extractable pseudo-random generators and extractable pseudo-random functions can be constructed from generic extractable primitives (as was asked and left open in [37]). Plausibly, our SNARK-based approach can be used to obtain the weaker variants of extractable pseudo-entropy generators [86] and pseudo-entropy functions [24], by relying on previous results regarding leakage resilience of PRGs [55,84,121] and leakage-resilient pseudo-entropy functions [24]. We leave the investigation of these possibilities and their applicability for future work.

(iii) Applications of SNARK*s.* As discussed earlier, SNARKs directly enable non-interactive delegation of computation, including settings where the delegator has a very long input or where the worker supplies his own input to the computation. An important property of SNARK-based delegation is that it does not require expensive preprocessing and (as a result) soundness can be maintained even when the prover learns the verifier's responses between successive delegation sessions because a fresh VGRS can simply be resent for each time.

In addition, SNARKs can be used to obtain zkSNARKs, that is, *zero-knowledge* succinct non-interactive arguments of knowledge in the common reference string (CRS) model. We provide two constructions to do so, depending on whether the succinct argument is "on top or below" a *non-interactive zero-knowledge proof* (NIZK) [19].

In an additional step, it is possible to obtain non-interactive two-party secure computation (against malicious adversaries) between a powerful sender and a weak receiver. To do so, start with a non-interactive two-party computation protocol that is secure against "almost" honest-but-curious adversaries, who are allowed to choose arbitrary randomness; such protocols are known to exist, e.g., based on fully homomorphic encryption [63]. Then, to make the protocol resilient to malicious adversaries, let the sender attach to his message a zkSNARK that his computation was performed honestly. The succinct receiver can use a standard NIZK to prove knowledge of his inputs. As for the sender, if his input is short he can also prove knowledge using a standard NIZK (as also done in [52]); otherwise, he must rely on the zkSNARK proof of knowledge. In particular, in the latter case, we only get non-concurrent security (rather than UC security) as the simulation relies on the non-black-box SNARK extractor.

In summary, SNARKs can be used for a number of applications:

Corollary 1.2. (Informal) *If there exist SNARKs, then:*

1. *There exist non-interactive delegation schemes where the verifier's response need not remain secret.*
   (Furthermore, there are such schemes that allow the worker to contribute its own input, as well as ones allowing to delegate memory and data streams [40,46]*.)*
2. *In the CRS model, there exist zero-knowledge SNARKs.*
3. *In the CRS model, there exist succinct non-interactive secure computation schemes (with UC security for short sender input and non-concurrent security for long sender input).*

For more details on the aforementioned applications, see the respective Sects. 9, 10.1, and 10.2.

(iii) Applications of EOWF*s*. As demonstrated by the construction of SNARK from ECRHs, extractable functions can be used to circumvent impossibilities of constructing certain primitives using black-box security reductions. We explore this direction and show that new feasibility results can be shown using EOWFs. Specifically, we first formalize a stronger variant of EOWFs—called strong EOWFs (sEOWF)—where every function in the ensemble is one-to-one and every sequence of functions is one-way. Using sEOWFs, we construct a two-message selective-opening-attack-secure (SOA-secure) commitment scheme. It was shown by Pass [111] that such a commitment scheme cannot be proven secure via black-box security reductions from any assumptions that can be formulated as a game between a challenger and an adversary (a.k.a. falsifiable assumptions [107]) of bounded rounds. In contrast, our scheme is proven secure using a non-black-box security reduction from a non-falsifiable assumption, that is, the existence of sEOWFs.

Furthermore, sEOWFs can be used to construct a three-round zero-knowledge argument of knowledge (ZKAOK) for NP, and if the sEOWFs are "concurrently extractable," then the protocol is also concurrent ZK. Additionally, the three-round protocol is public-coin; thus by applying standard transformations [21,54], it yields a three-round simultaneously resettable ZK protocol. It is known that these protocols do not admit black-box simulation [41,68]; our constructions indeed have non-black-box simulation utilizing the extractability of EOWFs.

For more details on the definitions and applications of (variants of) EOWFs, see Sect. 11.

1.3. *ECRH Candidate Constructions and Sufficient Relaxations*

We propose a natural candidate ECRH based on a generalization of the Knowledge of Exponent assumption in large algebraic groups [48]. The assumption, which we call *t-Knowledge of Exponent assumption* (or t-KEA for short), proceeds as follows. For any polynomial-size adversary, there exists a polynomial-size extractor such that, on input $g_1, \ldots, g_t, g_1^\alpha, \ldots, g_t^\alpha$ where each g_i is a random generator (of an appropriate group) and α is a random exponent: If the adversary outputs (f, f^α), then the extractor finds a vector of "coefficients" $(x_1, \ldots, x_t)$ such that $f = \prod_{i \in [t]} g_i^{x_i}$. This assumption can be viewed as a simplified version of the assumption used by Groth in [81] (the formal relation between the assumptions is discussed in Sect. 8.1). Similarly to Groth's assumption, t-KEA holds in the *generic group model.*

Theorem 4. (Informal) *If t-KEA holds in a group where taking discrete logs is hard, then there exists an ECRH that compresses inputs by a factor proportional to t.*

The construction is straightforward: The function family is parameterized by $(g_1, \ldots, g_t, g_1^\alpha, \ldots, g_t^\alpha)$. Given input $(x_1, \ldots, x_t)$, the function outputs the two group elements $(\prod_{i \in [t]} g_i^{x_i}, \prod_{i \in [t]} g_i^{\alpha x_i})$. Extractability directly follows from t-KEA, while collision resistance is ensured by the hardness of taking discrete logs. See Sect. 8.1 for more details.

Next we proceed to propose ECRH candidates that are based on the subset sum problem in finite groups. Here, however, we are only able to construct candidates for somewhat weaker variants of ECRHs that are still sufficient for constructing SNARKs. While these variants are not as elegantly and concisely stated as the "vanilla" ECRH notion, they are still natural. Furthermore, we can show that these variants are necessary for SNARKs. We next proceed to formulate these weaker variants.

1.3.1. *Proximity ECRH*

We say that $\mathcal{H}$ defined on domain D is a *proximity* ECRH (PECRH) if (for any $h \in \mathcal{H}$) there exist a reflexive "proximity" relation $\overset{h}{\approx}$ on values in the range and an extension of the hash to a larger domain $D_h \supseteq D$ fulfilling the following: (a) proximity collision resistance: Given $h \leftarrow \mathcal{H}$, it is hard to find $x, x' \in D_h$ such that $h(x) \overset{h}{\approx} h(x')$ and (b) proximity extraction: For any poly-time adversary $\mathcal{A}$ there exists an extractor $\mathcal{E}$ such that, whenever $\mathcal{A}$ outputs $y \in h(D)$, $\mathcal{E}$ outputs $x \in D_h$ such that $h(x) \overset{h}{\approx} y$. (See Definition 6.2 for further details.)

Harder to Find Collisions, Easier to Extract The notions of proximity extraction and proximity collision resistance are the same as standard extraction and collision resistance in the "strict" case, where $x \overset{h}{\approx} y$ is the equality relation and the domain is not extended ($D_h = \{0,1\}^{\ell(k)}, \bar{h} = h$).

However, in general, proximity collision resistance is stronger than (standard) collision resistance, because even "near collisions" (i.e., $x \neq y$ such that $\bar{h}(x) \overset{h}{\approx} \bar{h}(y)$) must not be efficiently discoverable, not even over the extended domain D_h. Conversely, proximity extraction is weaker than (standard) extraction, since it suffices that the extractor finds a point mapping merely close the adversary's output (i.e., finds x' such that $\bar{h}(x') \overset{h}{\approx} y$); moreover, it suffices that the point is in the extended domain D_h. Thus, the notion of PECRH captures another, somewhat more flexible trade-off between the requirements of extractability and collision resistance.

We show that any point on this trade-off (i.e., any choice of $\overset{h}{\approx}$, D_h and $\bar{h}$ fulfilling the conditions) suffices for the construction of SNARKs:

Theorem 5. (Informal) *If there exist PECRHs and (appropriate) PIRs, then there exist SNARKs for* NP.

Candidate PECRH*s Based on Knapsack (subset sum) Problems* A necessary property of ECRHs is that the image should be sparse; knapsack-based CRHs can often be tweaked to obtain this essential property. For example, in the t-KEA-based ECRH that we already discussed, we start from a standard knapsack hash $f = \prod_{i \in [t]} g_i^{x_i}$ and extend it to a "sparsified" knapsack hash (f, f^{α}) for a secret α. While for t-KEA this step is enough for plausibly assuming precise extractability (leading to a full-fledged ECRH), for other knapsack-based CRHs this is not the case.

For example, let us consider the task of sparsifying modular subset sum [117]. Here, the hash function is given by random coefficients $l_1, \ldots, l_t \in \mathbb{Z}_N$ and the hash of $x \in \{0,1\}^t$ is simply the corresponding modular subset sum $\sum_{i:x_i=1} l_i \bmod N$. A standard way to sparsify the function is, instead of drawing random coefficients, drawing them from a distribution of noisy multiples of some secret integer. However, by doing so, we lose the "algebraic structure" of the problem. Hence, now we also have to deal with new “oblivious image-sampling attacks” that exploit the noisy structure. For example, slightly perturbing an honestly computed subset sum is likely to “hit” another image of the function. This is where the relaxed notion of proximity extraction comes into play: It allows the extractor to output the preimage of the nearby (honest) image and, more generally, to thwart “perturbation attacks.”

Sparsification of modular subset sum in fact introduces additional problems. For instance, an attacker may take “small-norm” combinations of the coefficients that are not $0/1$ and still obtain an element in the image (e.g., if there are two even coefficients); to account for this, we need to further relax the notion of extraction by allowing the extractor to output a preimage in an extended domain, while ensuring that (proximity) collision resistance still holds for the extended domain too. Additionally, in some cases a direct naïve sparsification is not sufficient and we also need to introduce a notion of amplified knapsacks.

The relaxations of extractability discussed above have to be matched by a corresponding strengthening of collision resistance following the definition of PECRH. Fortunately, this can still be done under standard hardness assumptions.

A similar approach can be taken in order to sparsify the modular matrix subset sum CRH [5,64], resulting in a noisy inner-product knapsack hash based on the LWE assumption [119]. Overall, we propose three candidates for PECRHs:

Theorem 6. (Informal) *There exist PECRHs under any of the following assumptions:*

1. *A Knowledge of Knapsack of Exponent assumption (which in fact follows from t-KEA) and hardness of discrete logs.*
2. *A Knowledge of Knapsack of Noisy Multiples assumption and lattice assumptions.*
3. *A Knowledge of Knapsack of Noisy Inner Products assumption and learning with errors.*

1.3.2. *Weak PECRHs*[4]

Our second weakening is essentially orthogonal to the first one and relates to the condition that determines when the extractor has to “work.” The ECRH and PECRH definitions required extraction whenever the adversary outputs a valid image; here the sparseness of the image appears to be key. In particular, unstructured CRHs where one can sample elements in the image obliviously of their preimage have no hope to be either ECRH or PECRH. However, for our purposes it seems sufficient to only require the extractor to “work” when the adversary outputs an image y together with extra encoding of a preimage that can be verified given proper trapdoor information; oblivious image-sampling, on its own, is no longer sufficient for “breaking” the extractor.

More formally, a family $\mathcal{H}$ of functions is weakly extractable if for any efficient adversary $\mathcal{A}$ there exists an efficient extractor $\mathcal{E}_{\mathcal{A}}^{\mathcal{H}}$ such that for any auxiliary input z and efficient decoder $\mathcal{Y}$, the probability of the event that $\mathcal{A}$ outputs, given z and a random function h from the family, values y and e such that: (a) $\mathcal{E}_{\mathcal{A}}^{\mathcal{H}}$, given z and h, does not find a preimage of y, but (b) $\mathcal{Y}$, given e, does find a preimage of y, is negligible. See Definition 6.3 for further details.

[4]This further weakening was inspired by private communication with Ivan Damgård.

We stress that the decoder circuit $\mathcal{Y}$ is allowed to arbitrarily depend on the auxiliary input z. This is $\mathcal{Y}$'s advantage over the extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$ (that must be efficient in z); in particular, the extractability condition is not trivially true. Another interpretation of the above definition is the following: The definition requires that there exists a single extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$ that does as well as any other efficient extractor (that may depend on z); namely, any given decoder circuit $\mathcal{Y}$ can be thought of as a candidate extractor and the extractor $\mathcal{E}^{H}_{A}$ has to succeed whenever $\mathcal{Y}$ does despite being efficient in the auxiliary input. In particular, whenever extraction is possible when given trapdoor information related to the auxiliary input, it should also be possible without such trapdoor information. Indeed, the ability of the extractor to forgo the trapdoor is the key to a successful use of the above definition in our construction of SNARKs:

Theorem 7. (Informal) *If there exist weak PECRHs and (appropriate) PIRs, then there exist SNARKs for* NP.

Unlike ECRHs and PECRHs, weak ECRHs may in principle not require sparsity of the image or algebraic structure. For example, it may be plausible to assume that (a properly keyed) SHA-2 is a weak ECRHs.

We remark that the notion of weak ECRH is somewhat reminiscent of the notion of *preimage awareness* considered by Dodis et al. [59]. Their notion, however, is set in an idealized model where the hash function can only be accessed as a black-box. When ported to such an ideal model, our definition can be viewed as a strengthening of the [59] definition to account for auxiliary input. Similarly to the [59] definition, the idealized version of our definition also holds in the *random oracle model*. (We do not know whether an idealized ECRH or PECRH can be constructed unconditionally in the random oracle model; however, assuming the existence of standard CRHs, so that a concise description for a CRH is available, it is possible to construct PECRHs in the random oracle model using CS proofs of knowledge.)

1.4. *High-Level Proof Strategy for Theorem 1*

In this section, we provide some high-level intuition for the proof of our main technical result, showing that the existence of ECRHs and (appropriate) PIR schemes imply the existence of SNARKs. At a very high level, our construction follows the "PCP+MT+PIR approach" of [49], which in turn is built upon Kilian's four-message succinct interactive argument system [89]. So let us start by reviewing them.

Kilian's Construction In [89], Kilian presents a succinct zero-knowledge argument for NP, $(\mathcal{P}, \mathcal{V})$, where the prover $\mathcal{P}$ uses a Merkle tree in order to provide to $\mathcal{V}$ "virtual access" to a PCP proof. More precisely, to prove a statement $x \in L$, the verifier $\mathcal{V}$ starts by sending the prover a CRHF H. The prover on private input a witness w constructs a PCP proof π. In order to yield efficient verifiability, $\mathcal{P}$ cannot send $\mathcal{V}$ the witness w nor π. Rather, $\mathcal{P}$ builds a Merkle tree with the proof π as the leaf values (using the collision-free hash function H from the verifier) producing a root value rv. It then "commits" itself to π by sending rv to the verifier $\mathcal{V}$. The verifier $\mathcal{V}$ tosses a fixed polynomial number of random coins r and sends them to the prover. Both the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ compute the PCP queries by internally running the PCP verifier on input x

and r. The prover $\mathcal{P}$ sends back answers to those queries, together with "proofs"—called authentication paths—that each answer is consistent with the root of the Merkle tree. Finally, the verifier accepts if all the answers are consistent with the root value rv, and convinces the PCP verifier.

Killan's protocol is succinct, because the verifier, invoking the PCP verifier, makes only a fixed polynomial number of queries and each query is answered with an authentication path of some fixed polynomial length, all independent of the length of the witness. At a very high level, the soundness follows from the fact that the Merkle tree provides the verifier "virtual access" to the PCP proof, in the sense that given the root value of the Merkle tree, for every query q, it is infeasible for a cheating prover to answer q differently depending on the queries. Therefore, interacting with the prover is "equivalent" to having access to a PCP proof oracle. Then it follows from the soundness of the PCP system that Kilian's protocol is sound.

The "PCP+MT+PIR approach" The work of [49] proposed the "PCP+MT+PIR approach" to "squash" Kilian's four-message protocol into a two-message protocol as follows. In Kilian's protocol, the verifier obtains from the prover a Merkle hash to a PCP oracle and only then asks the prover to locally open the queries requested by the PCP verifier. In [49]'s protocol, the verifier sends also in the first message, a PIR-encrypted version of the PCP queries (the first message of a PIR scheme can be viewed as an encryption of the queries); the prover then prepares the required PCP oracle, computes and sends a Merkle hash of it, and answers the verifier's queries by replying to the PIR queries according to a database that contains the answer (as well as the authentication path with respect to the Merkle hash) to every possible verifier's query. [49] proved the soundness of the above scheme based on the assumption that any convincing prover $\mathcal{P}^*$ must essentially behave as an honest prover: Namely, if a proof is accepting, then the prover must have in mind a *full* PCP oracle π, which maps under the Merkle hash procedure to the claimed root, and such a proof π can be obtained by an efficient extractor $\mathcal{E}_{\mathcal{P}^*}$.[5] [49] then showed that, if this is the case, the extracted string π must be consistent with the answers the prover provides to the PCP queries, for otherwise the extractor can be used to obtain collisions of the hash function underlying the Merkle tree. Therefore, the extracted string π also passes the PCP test, where the queries are encrypted under PIR. Then, it follows from the privacy of the PIR scheme that the string π is "computationally independent" of the query. Hence from the soundness of PCP, they conclude that the statement must be true.

1.4.1. *The Main Challenges and Our Solutions*

Our goal is to obtain the stronger notion of SNARKs, based on the more restricted assumption that ECRHs exist. At a very high-level, our construction follows the "PCP+MT+PIR approach" but replacing the CRH underlying the Merkle tree in their construction with an ECRH. Unlike [49] which directly assumed the "global extraction"

[5] Note that, as originally formulated, the assumption of [49] seems to be false; indeed, a malicious prover can always start from a good PCP oracle π for a true statement and compute an "almost full" Merkle hash on π, skipping very few branches—so one should at least formulate an analogous but more plausible assumption by only requiring "sufficient consistency" with the claimed root.

guarantees from a Merkle tree, we show that we can *lift* the "local extraction" guarantee provided by the ECRH, to the "global extraction" guarantee on the entire Merkle tree. More precisely, by relying on the (local) ECRH extraction guarantee, we show that for every convincing prover, there is an extractor that can efficiently extract a PCP proof $\tilde{\pi}$ that is "sufficiently satisfying" (i.e., a PCP verifier given $\tilde{\pi}$ accepts with high probability). Furthermore, to obtain the argument of knowledge property, we instantiate the underlying PCP system with PCPs of knowledge, which allows for extracting a witness from any sufficiently satisfying proof oracle. (See details for the requisite PCP system in Sect. 3.7.) We next describe the high-level ideas for achieving global extraction guarantees from the local extraction of ECRHs. Full details are contained in Sect. 5, and the construction is summarized in Table 1.

From Local to Global Extraction The main technical challenge lies in establishing a "global" knowledge feature (i.e., informally, extraction of a sufficiently satisfying proof $\tilde{\pi}$ from a convincing prover given that it outputs the root of a Merkle tree) from a very "local" one (namely, extraction of a preimage from a machine that outputs a valid hash value of the ECRH h). A natural attempt is to start from the root of the Merkle tree and extract the values of the intermediate nodes of the Merkle tree layer by layer toward the leaves; that is, to extract a candidate proof $\tilde{\pi}$ by recursively applying the ECRH extractor to extract the entire Merkle tree $\widetilde{\mathrm{MT}}$, where the leaves should correspond to $\tilde{\pi}$.

However, recursively composing ECRH extractors encounters a difficulty: Each time applying the ECRH extraction incurs a polynomial blowup in extraction time. Therefore (without making a very strong assumption on the amount of "blowup" incurred by the extractor), we can only apply the ECRH extraction recursively a constant number of times; as a result, we can only extract from Merkle trees of a constant depth. We address this problem by opting to use a "squashed" Merkle tree, where the fan-in of each intermediate node is polynomial rather than binary as in the traditional case. Consequently, for every NP statement, since its PCP proof is of polynomial length, the depth ℓ of the Merkle tree built over the PCP proof is a constant. Then, we can apply the ECRH extraction procedure recursively to extract the whole Merkle tree, while overall incurring polynomial overhead in the size of the extractor.

A technical problem that arises when applying the above recursive extraction procedure is that we might not be able to extract out a full Merkle tree (where all paths have the same length). More precisely, after applying the ECRH extraction recursively ℓ times, we obtain the values for the intermediate nodes up to level ℓ (thinking about the root as level zero). However, at each intermediate node, when applying the ECRH extractor, extraction could have failed to extract a preimage if the given intermediate node is not a valid hash image under the ECRH. Hence, the extracted tree might be at some points "disconnected."[6] Nevertheless, we show (relying solely on ECRH extraction) that the leaves in the extracted (perhaps partial) tree connected to the root are sufficiently satisfying for witness extraction.

Proof at High Level Given the foregoing discussion, we show the correctness of the extraction procedure in two steps:

[6]This captures for example the behavior of the prover violating the [49] assumption described above.

- *Step 1: "local consistency."* We first show that whenever the verifier is convinced, the recursively extracted string $\tilde{\pi}$ contains valid answers to the verifier's PCP queries specified in its PIR queries. Otherwise, it is possible to find collisions within the ECRH h as follows. A collision finder could simulate the PIR encryption on its own, invoke both the prover and the corresponding extractor for the ECRH h, and obtain two paths that map to the same root but must differ somewhere (as one is satisfying and the other is not) and therefore obtain a collision.
- *Step 2: "from local to global consistency."* Next, using the privacy guarantees of the PIR scheme, we show that, whenever we extract a set of leaves that are satisfying with respect to the PIR-encrypted queries, the same set of leaves must also be satisfying for almost all other possible PCP queries and is thus sufficient for witness extraction. Indeed, if this was not the case, then we would be able to use the polynomial-size extraction circuit to break the semantic security of the PIR.

For further details of the proof, we refer the reader to Sect. 5.2.

Our construction ensures that the communication complexity and the verifier's time complexity are bounded by a polynomial in the security parameter, the size of the instance, and the logarithm of the time it takes to verify a valid witness for the instance; furthermore, this polynomial is *independent* of the specific NP language at hand.

On Adaptive Soundness The soundness requirement of SNARK prevents a cheating prover from proving any false statement even if it is allowed to choose the statement *after* seeing the verifier's first message (or, rather, the verifier-generated reference string). In the above construction, the verifier sends in the first message the PIR-encrypted PCP queries. However, in general, the PCP queries may depend on the statement to be proven, and hence limit the construction to be only statically sound. To resolve this problem, we modify the above protocol as follows: In the first message, the verifier PIR-encrypts the PCP verifier's random coins rather than its actual queries (as the former are independent of the statement), and require the prover to prepare an appropriate database (containing all the possible answers for each setting of the coins, rather than a proof oracle). A subtle problem with this approach is that in general, the number of random coins tossed by the PIR verifier may depend on the size of the database, which is not known to the verifier a priori. This is solved in the body by enumerating over possible bounds 2^ρ on the database size for all values ρ upto some superlogarithmic function of the security parameter.

On Universal Soundness: As for soundness, our main construction is not universal, in the sense that the verifier needs to know a constant c such that the verification time of an instance y does not exceed $|y|^c$. Fortunately, this very weak dependence on the specific NP language at hand (weak because it does not even depend on the Turing machine verifying witnesses) does not affect the application to delegation of computation, because the delegator, having in mind a specific poly-time task to delegate, *knows* c. Furthermore, we also show how, by assuming the existence of exponentially hard one-way functions, our main construction can be extended to be a *universal* SNARK, that is, a single protocol that can simultaneously work with all NP languages.

1.5. *Discussion*

We conclude the introduction with an attempt to briefly motivate the premise of this work. Our main contribution can be seen as providing additional understanding of the security of a construction that has frustrated researchers. Toward this goal, we prove strong security properties of the scheme based on a new cryptographic primitive, ECRHs, that, while not fitting into the mold of "standard cryptographic primitives or assumptions," can be defined concisely and investigated separately.

Furthermore, we investigate a number of relaxations of ECRHs as well as a number of candidate constructions that have quite different underlying properties. Looking beyond the context of our particular protocol, this work can be seen as another step toward understanding the nature of extractability assumptions and their power in cryptography.

1.6. *Subsequent Work*

Subsequent to our paper, many works have contributed a greater understanding of SNARKs and delegation schemes. At present, there are many SNARK constructions in the literature, with different properties in efficiency and supported languages. In *pre-processing* SNARK*s*, the complexity of the setup of public parameters grows with the size of the computation being proved [8,14,15,18,38,51,61,66,81,92,97–99,112,126,127]; in *fully succinct* SNARK*s*, which are the ones that we study in this work, that complexity is independent of computation size [9,11,12,17,49,52,71,104,105,124]; in particular, some later works also achieve public verifiability, which we do not achieve in this work. Another line of research focused on delegation schemes for P computations (rather than NP as in the case of SNARKs), see [91,95,96,113] and references therein; all of these works do not rely on knowledge assumptions but, instead, rely on (often subexponential) reductions to falsifiable assumptions.

Finally, [16] construct extractable one-way functions against adversaries with any polynomial running time but whose auxiliary input is bounded by a fixed polynomial (and, on the negative side, they show that indistinguishability obfuscation rules out extractable one-way functions for the case where the auxiliary input is not bounded by a fixed polynomial).

1.7. *Organization*

In Sect. 2, we discuss more related work. In Sect. 3, we give basic definitions for the cryptographic primitives that we use (along with any nonstandard properties that we may need). In Sect. 4, we give the definitions for SNARKs. In Sect. 5, we give our main construction showing that ECRHs, along with appropriate PIRs, imply SNARKs. In Sect. 6, we discuss two relaxations of ECRHs, namely PECRHs and weak PECRHs, that still suffice for SNARKs. In Sect. 7, we explain how SNARKs imply proximity PECRHs, thus showing an almost tight relation between the existence of SNARKs and PECRHs. In Sect. 8, we propose candidate constructions for ECRHs and PECRHs. In Sect. 9, we show how to obtain zero-knowledge SNARKs. In Sect. 10, we provide further discussion for how SNARKs can be used in the setting of delegation of computation and secure computation. Finally, in Sect. 11 we define two notions of extractable one-way functions and show how to use them to construct, respectively, non-interactive (two-message) selective-opening-attack secure commitments and three-round concurrent zero-knowledge protocols.

2. Other Related Work

Knowledge Assumptions A popular class of knowledge assumptions, which have been successfully used to solve a number of (at times notoriously open) cryptographic problems, is that of *Knowledge of Exponent* assumptions. These have the following flavor: If an efficient circuit, given the description of a finite group along with some other public information, computes a list of group elements that satisfies a certain algebraic relation, then there exists a knowledge extractor that outputs some related values that "explain" how the public information was put together to satisfy the relation. Most such assumptions have been proven secure against generic algorithms (see Nechaev [108], Shoup [123], and Dent [50]), thus offering some evidence for their truth. In the following, we briefly survey prior works which, like ours, relied on Knowledge of Exponent assumptions.

Damgård [48] first introduced a Knowledge of Exponent assumption to construct a CCA-secure encryption scheme. Later, Hada and Tanaka [87] showed how to use two Knowledge of Exponent assumptions to construct the first three-round zero-knowledge argument. Bellare and Palacio [27] then showed that one of the assumptions of [87] was likely to be false, and proposed a modified assumption, which they used to construct a three-round zero-knowledge argument.

More recently, Abe and Fehr [3] extended the assumption of [27] to construct the first perfect NIZK for NP with "full" adaptive soundness. Prabhakaran and Xue [116] constructed statistically hiding sets for trapdoor DDHgroups [53] using a new Knowledge of Exponent assumption. Gennaro et al. [70] used another Knowledge of Exponent assumption (with an interactive flavor) to prove that a modified version of the Okamoto–Tanaka key agreement protocol [110] satisfies perfect forward secrecy against fully active attackers.

In a different direction, Canetti and Dakdouk [36,37,47] study *extractable functions*. Roughly, a function f is extractable if finding a value x in the image of f implies knowledge of a preimage of x. The motivation of Canetti and Dakdouk for introducing extractable functions is to capture the abstract essence of prior knowledge assumptions and to formalize the "knowledge of query" property that is sometimes used in proofs in the Random Oracle Model. They also study which security reductions are "knowledge preserving" (e.g., whether it possible to obtain extractable commitment schemes from extractable one-way functions).

[16,28] show that, assuming indistinguishability obfuscation [22], extractable one-way functions (and thus also ECRHs) cannot be constructed against adversaries with arbitrary polynomial-size auxiliary input if the (efficient) extractor is universally fixed before the adversary's auxiliary input. On the other hand, they show that, under standard assumptions, extractable one-way functions are achievable against adversaries with a priori bounded auxiliary input. (It is still not known whether such ECRHs can also be constructed under standard assumptions.)

Prior (somewhat) succinct arguments from Knowledge of Exponent assumptions. Knowledge of exponent assumptions have been used to obtain somewhat succinct arguments, in the sense the non-interactive proof is short, but the verifier's running time is long.

Groth [81] introduced a family of knowledge of exponent assumptions, generalizing those of [3], and used them to construct extractable length-reducing commitments, as a building block for short non-interactive perfect zero-knowledge arguments system for circuit satisfiability. These arguments have very succinct proofs (independent of the circuit size), though the public key is large: quadratic in the size of the circuit. Groth's assumption holds in the generic group model. For a comparison between our t-KEA assumption and Groth's assumptions, see Sect. 8.1.

Mie [105] observes that the PCP+MT+PIR approach works as long as the PIR scheme is *database aware*—essentially, a prover that is able to provide valid answers to PIR queries must "know" their decrypted values, or, equivalently, must "know" a database consistent with those answers (by arbitrarily setting the rest of the database). Mie then shows how to make the PIR scheme of Gentry and Ramzan [80] PIR aware, based on Damgård's Knowledge of Exponent assumption [48]. Unfortunately, while the communication complexity is very low, the receiver in [80] and thus also the verifier in [105] are inefficient relative to the database size.

Delegation of Computation An important application of succinct arguments is *delegation of computation* schemes, where one usually also cares about privacy, and not only soundness, guarantees. Specifically, a succinct argument can be usually combined in a trivial way with fully homomorphic encryption [63] (in order to ensure privacy) to obtain a delegation scheme where the delegator runs in time polylogarithmic in the running time of the computation (see Sect. 10.1).

Within the setting of delegation, however, where the same weak delegator may be asking a powerful untrusted worker to evaluate an expensive function on many different inputs, a weaker preprocessing approach may still be meaningful. In such a setting, the delegator performs a one-time function-specific expensive setup phase, followed by inexpensive input-specific delegations to amortize the initial expensive phase. Indeed, in the preprocessing setting a number of prior works have already achieved constructions where the online stage is only two messages [4,42,65]. These constructions do not allow for an untrusted worker to contribute his own input to the computation, namely they are "P-delegation schemes" rather than "NP-delegation schemes." Note that all of these works do not rely on any knowledge assumption; indeed, the impossibility results of [84] only apply for NP and not for P.

However, even given that the preprocessing model is very strong, all of the mentioned works maintain soundness over many delegations only as long as the verifier's answers remain secret. (A notable exception is the work of Benabbas et al. [23], though their constructions are not generic and are only for specific functionalities such as polynomial functions.)

Goldwasser et al. [69] construct interactive proofs for log-space uniform NC where the verifier running time is quasi-linear. When combining [69] with the PIR-based squashing technique of Kalai and Raz [93], one can obtain a two-message delegation scheme for log-space uniform NC. Canetti et al. [44] introduce an alternative way of squashing [69], in the preprocessing setting; their scheme is of the public coin type and hence the verifier's answers need not remain secret (another bonus is that the preprocessing state is publicly verifiable and can thus be used by anyone).

3. Preliminaries

In this section, we give basic definitions for the cryptographic primitives that we use (along with any nonstandard properties that we may need).

3.1. *Conventions*

The cryptographic definitions in the paper follow the convention of modeling security against non-uniform adversaries. An efficient adversary $\mathcal{A}$ is modeled as a sequence of circuits $\mathcal{A} = \{\mathcal{A}_k\}_{k\in\mathbb{N}}$, such that each circuit $\mathcal{A}_k$ is of polynomial size $k^{O(1)}$ with $k^{O(1)}$ input and output bits. We often omit the subscript k when it is clear from the context.

When we refer to a randomized algorithm $\mathcal{A}$, we typically do not explicitly denote its random coins, and use the notation $s \leftarrow \mathcal{A}$ or $s \leftarrow \mathcal{A}(x)$ if $\mathcal{A}$ has an extra input x. When we want to be explicit regarding the coins, we shall denote $s \leftarrow \mathcal{A}(r)$, or $s \leftarrow \mathcal{A}(x; r)$, respectively.

Whenever we refer to a circuit class $\mathcal{C} = \{\mathcal{C}_k\}$, we mean that each set $\mathcal{C}_k$ consists of Boolean circuits of size at most poly(k) for some polynomial poly$(\cdot)$, defined on the domain $\{0, 1\}^{n(k)}$. When referring to inputs $x \in \{0, 1\}^{n(k)}$, we often omit k from the notation.

Throughout, negl(k) is any negligible function in k.

3.2. *Collision-Resistant Hashes*

A *collision-resistant hash* (CRH) is a function ensemble for which it is hard to find two inputs that map to the same output. Formally:

Definition 3.1. A function ensemble $\mathcal{H}$ is a CRH if it is collision-resistant in the following sense: For every polynomial-size adversary $\mathcal{A}$ and any $k \in \mathbb{N}$,

$$\Pr_{h\leftarrow\mathcal{H}_k}\left[\begin{array}{c} x \neq y \\ h(x) = h(y) \end{array} : (x, y) \leftarrow \mathcal{A}(h)\right] \leq \text{negl}(k) \ .$$

We say that a function ensemble $\mathcal{H}$ is $(\ell(k), k)$-*compressing* if each $h \in \mathcal{H}_k$ maps strings of length $\ell(k)$ to strings of length $k < \ell(k)$.

3.3. *Merkle Trees*

Merkle tree (MT) hashing [103] enables a party to use a CRHto compute a succinct commitment to a long string π and later to locally open any bit of π (again in a succinct manner). Specifically, given a function $h: \{0, 1\}^{\ell(k)} \to \{0, 1\}^k$ randomly drawn from a CRHensemble, the committer divides π into $|\pi|/\ell(k)$ parts (padding with 0's if needed) and evaluates h on each of these; the same operation is applied to the resulting string, and so on, until one reaches the single k-bit root. For $|\pi| = (\ell/k)^{d+1}$, this results in a tree of depth d, whose nodes are all the intermediate k-bit hash images. An opening to a leaf in π (or any bit within it) includes all the nodes and their siblings along the path from the root to the leaf and is of size ℓd. Typically, $\ell(k) = 2k$, resulting in a binary tree of depth $\log \pi$. In this work, we shall also be interested in "wide trees" with polynomial fan-in (relying on CRHs with polynomial compression), see Sect. 5.1.

3.4. *Private Information Retrieval*

A (single-server) computational polylogarithmic *private information retrieval* (PIR) scheme [43,90] consists of a triple of algorithms $(\mathsf{PEnc}, \mathsf{PEval}, \mathsf{PDec})$ where:

- $\mathsf{PEnc}_R(1^k, i)$ outputs an encryption C of query $i \in \{0,1\}^n$ to a database DB with $N = 2^n$ entries using randomness R,
- $\mathsf{PEval}(\mathsf{DB}, C)$ outputs a string e "containing" the answer $\mathsf{DB}[i]$, and
- $\mathsf{PDec}_R(e)$ decrypts the string e to an answer $\mathsf{DB}[i]$.

Formally:

Definition 3.2. A triple of algorithms $(\mathsf{PEnc}, \mathsf{PEval}, \mathsf{PDec})$ is a PIR if it has the following properties:

1. **Correctness** For any database DB with $N = 2^n$ entries in $\{0,1\}^\ell$, any query $i \in \{0,1\}^n$, and security parameter $k \in \mathbb{N}$,

$$\Pr_R\left[\mathsf{PDec}_R(e) = \mathsf{DB}[i] : \begin{array}{c} C \leftarrow \mathsf{PEnc}_R(1^k, i) \\ e \leftarrow \mathsf{PEval}(\mathsf{DB}, C) \end{array}\right] = 1 \ ,$$

   where $\mathsf{PEval}(\mathsf{DB}, C)$ runs in $\mathrm{poly}(k, N, \ell)$ time.
2. **Succinctness** The running time of both $\mathsf{PEnc}_R(1^k, i)$ and $\mathsf{PDec}_R(e)$ is bounded by

$$\mathrm{poly}(k, n, \ell) \ ,$$

   for some fixed polynomial poly, independent of DB. In particular, the sizes of the two messages C and e are also so bounded.
3. **Semantic security** The query encryption is semantically secure for multiple queries, i.e., for any polynomial-size $\mathcal{A}$, any security parameter $k \in \mathbb{N}$ and any two tuples of queries $\mathbf{i} = (i_1 \cdots i_q)$, $\mathbf{i}' = (i'_1 \cdots i'_q) \in \{0,1\}^{\mathrm{poly}(k)}$,

$$\Pr\left[\mathcal{A}(\mathsf{PEnc}_R(1^k, \mathbf{i})) = 1\right] - \Pr\left[\mathcal{A}(\mathsf{PEnc}_R(1^k, \mathbf{i}')) = 1\right] \leq \mathrm{negl}(k) \ ,$$

   where $\mathsf{PEnc}_R(1^k, \mathbf{i}) = (\mathsf{PEnc}_{R_1}(1^k, i_1), \cdots, \mathsf{PEnc}_{R_q}(1^k, i_q))$ is the coordinate-wise encryption the tuple $\mathbf{i}$, using independent random blocks of $R = (R_1, \ldots, R_q)$.

PIR schemes with the above properties have been constructed under various hardness assumptions such as ΦHA [43] or LWE [33].

3.5. *The Complexity Class* NP *and Witness Relation*

We recall the class NP, i.e., the class of languages for which there exists a proof of membership that can be verified in polynomial time.

Definition 3.3. (Complexity Class NP) A language L is in NP if there exists a Boolean relation $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$ and a polynomial $p(\cdot)$ such that R_L is recognizable in polynomial time, and $x \in L$ if and only if there exists a string $y \in \{0,1\}^*$ such that $|y| \leq p(|x|)$ and $(x, y) \in R_L$. The relation R_L is called a NP *relation* and is a *witness relation* for L.

We say that y is a witness for the membership $x \in L$ if $(x, y) \in R_L$. We will also let $R_L(x)$ denote the set of witnesses for the membership $x \in L$, i.e., $R_L(x) = \{y : (x, y) \in L\}$.

3.6. *The Universal Relation*

The *universal relation* [20] is defined to be the set $\mathcal{R}_{\mathcal{U}}$ of instance–witness pairs (y, w), where $y = (M, x, t)$, $|w| \leq t$, and M is a Turing machine, such that M accepts (x, w) after at most t steps. While the witness w for each instance $y = (M, x, t)$ is of size at most t, there is *no a priori* polynomial bound on t in terms of $|x|$.

Also, for any $c \in \mathbb{N}$, we denote by $\mathcal{R}_c$ the subset of $\mathcal{R}_{\mathcal{U}}$ consisting of those pairs $(y, w) = \big((M, x, t), w\big)$ for which $t \leq |x|^c$. This is a "generalized" NP relation, where we do not insist on the same Turing machine accepting different instances, but only insist on a fixed polynomial bounding the running time in terms of the instance size.

3.7. *Probabilistically Checkable Proofs of Knowledge*

A (verifier-efficient) *probabilistically checkable proof* (PCP) of knowledge for the universal relation $\mathcal{R}_{\mathcal{U}}$ is a triple of algorithms $(P_{\mathsf{pcp}}, V_{\mathsf{pcp}}, E_{\mathsf{pcp}})$, where P_{pcp} is the prover, V_{pcp} is the (randomized) verifier, and E_{pcp} is the knowledge extractor.

Given $(y, w) \in \mathcal{R}_{\mathcal{U}}$, $P_{\mathsf{pcp}}(y, w)$ generates a proof π of length $\mathrm{poly}(t)$ and runs in time $\mathrm{poly}(|y|, t)$. The verifier $V^{\pi}_{\mathsf{pcp}}(y, r)$ runs in time $\mathrm{poly}(|y|) = \mathrm{poly}(|M| + |x| + \log t)$, while accessing $\mathrm{polylog}(t)$ locations in the proof π and using $\rho = O(\log t)$ random bits. We require:

1. **Completeness** For every $(y, w) = \big((M, x, t), w\big) \in \mathcal{R}_{\mathcal{U}}, \pi \leftarrow P_{\mathsf{pcp}}(y, w)$:

$$\Pr_{r \leftarrow \{0,1\}^{\rho(t)}} \left[V^{\pi}_{\mathsf{pcp}}(y, r) = 1 \right] = 1 \ .$$

2. **Proof of knowledge** (PoK). There is a constant $\varepsilon < 1$ such that for any $y = (M, x, t)$ if

$$\Pr_{r \leftarrow \{0,1\}^{\rho(t)}} \left[V^{\pi}_{\mathsf{pcp}}(y, r) = 1 \right] \geq 1 - \varepsilon \ ,$$

then $E_{\mathsf{pcp}}(y, \pi)$ outputs a witness w such that $(y, w) \in \mathcal{R_U}$, and runs in time $\mathrm{poly}(|y|, t)$.

(Note that proof of knowledge in particular implies that the soundness error is at most ε.)

PCPs of knowledge as defined above can be based on the efficient verifier PCPs of [30,31]. (See [124] for a simple example of how a PCP of proximity can yield a PCP with a proof of knowledge.) Moreover, the latter PCPs' proof length is quasi-linear in t; for simplicity, we shall settle for a bound of t^2.

We shall typically invoke the verifier V_{pcp} for $q(k)$ times to reduce the proof-of-knowledge threshold to $(1-\varepsilon)^{q(k)}$, where k is the security parameter and $q(k) = \omega(\log k)$. Namely, extraction should succeed whenever $\Pr_{\mathbf{r}}\left[V^{\pi}_{\mathsf{pcp}}(y, \mathbf{r}) = 1\right] \geq (1-\varepsilon)^q$, where $\mathbf{r} = (r_i)_{i\in[q]}$ and $V^{\pi}_{\mathsf{pcp}}(y, \mathbf{r}) = \bigwedge_{i\in[q]} V^{\pi}_{\mathsf{pcp}}(y, r_i)$.

3.8. *Indistinguishability*

The following definition of (computational) indistinguishability originates in the seminal paper of Goldwasser and Micali [72].

Definition 3.4. (Indistinguishability) Let X be a countable set. Two ensembles $\{A_x\}_{x\in X}$ and $\{B_x\}_{x\in X}$ are said to be *computationally indistinguishable over* X, if for every probabilistic "distinguishing" algorithm D whose running time is polynomial in its first input, there exists a negligible function $\nu(\cdot)$ so that for every $x \in X$:

$$|\Pr[a \leftarrow A_x : D(x, a) = 1] - \Pr[b \leftarrow B_x : D(x, b) = 1]| < \nu(|x|)$$

$\{A_x\}_{x\in X}$ and $\{B_x\}_{x\in X}$ are said to be *statistically close* over X if the above condition holds for all (possibly unbounded) algorithms D.

3.9. *Interactive Proofs, Zero Knowledge and Witness Indistinguishability*

We use the standard definitions of interactive proofs (and interactive Turing machines) [73] and arguments (also known as computationally sound proofs) [10]. Given a pair of interactive Turing machines, P and V, we denote by $\langle P(w), V\rangle(x)$ the random variable representing the final (local) output of V, on common input x, when interacting with machine P with private input w, when the random tape to each machine is uniformly and independently chosen.

Definition 3.5. (Interactive Proof System) A pair (P, V) of interactive machines is called an interactive proof system for a language L with respect to a witness relation R_L if the following two conditions hold :

- *Completeness:* For every $x \in L$ and $w \in R_L(x)$, $\Pr[\langle P(w), V\rangle(x) = 1] = 1$.
- *Soundness:* For every interactive machine B, there is a negligible function $\nu(\cdot)$, such that, for every $x \in \{0, 1\}^n \setminus L$, $\Pr[\langle B, V\rangle(x) = 1] \leq \nu(n)$.

In the case that the soundness condition is required to hold only with respect to a polynomial-size prover, the pair $\langle P, V\rangle$ is called an interactive *argument* system.

Zero Knowledge An interactive proof is said to be *zero knowledge* (ZK) if a (malicious) verifier V^* learns nothing beyond the validity of the assertion being proved. Because "feasible computation" is typically captured through the notion of probabilistic polynomial time, the ZK property is formalized by requiring that the output of every (possibly malicious) verifier interacting with the honest prover P can be *simulated* by a probabilistic (expected) polynomial-time machine S, known as the *simulator*. The idea behind this definition is that the anyone can learn by himself, by running the simulator S, whatever V^* learns by interacting with P.

The notion of ZK was introduced and formalized by Goldwasser, Micali, and Rackoff in [73]. We present their definition below.

Definition 3.6. (ZK) Let L be a language in NP, R_L a witness relation for L, (P, V) an interactive proof (argument) system for L. We say that (P, V) is *(computational) ZK*, if for every probabilistic polynomial-time interactive machine V there exists a probabilistic algorithm S whose expected running time is polynomial in the length of its first input, such that, for every ensemble $\{(x_n, y_n, z_n)\}_{n\in\mathbb{N}}$, where $x_n \in \{0, 1\}^n \cap L$, $y \in R_L(x)$, and $z \in \{0, 1\}^{\text{poly}(n)}$, the following two ensembles are computationally indistinguishable.

- $\left\{\text{View}_V[\langle P(y_n), V(z_n)\rangle(x_n)]\right\}_{n\in\mathbb{N}}$
- $\left\{S(x_n, z_n)\right\}_{n\in\mathbb{N}}$

where $\text{View}_V[\langle P(y_n), V(z_n)\rangle(x_n)]$ denote the random variable describing the view of V in interaction with P on common input x_n (the statement) and private inputs y_n (the witness) to P and z_n (the auxiliary input) to V respectively.

Witness indistinguishable proofs An interactive proof is said to be *witness indistinguishable* ($\mathcal{WI}$) if the verifier's output is "computationally independent" of the witness used by the prover for proving the statement. In this context, we focus on languages $L \in \mathcal{N}P$ with a corresponding witness relation R_L. Namely, we consider interactions in which, on common input x, the prover is given a witness in $R_L(x)$. By saying that the output is computationally independent of the witness, we mean that for any two possible NP witnesses that could be used by the prover to prove the statement $x \in L$, the corresponding outputs are computationally indistinguishable.

Definition 3.7. (*Witness indistinguishability*) Let (P, V) be an interactive proof system for a language $L \in \mathcal{N}P$. We say that (P, V) is *witness indistinguishable* for R_L, if for every probabilistic polynomial-time interactive machine V^* and for every ensemble $\{(x_n, w_n^1, w_n^2, z_n)\}_{n\in\mathbb{N}}$, such that, for every $n \in \mathbb{N}$, $x_n \in \{0, 1\}^n \cap L$, $w_n^1, w_n^2 \in R_L(x)$, and $z_n \in \{0, 1\}^{\text{poly}(n)}$, the following ensembles are computationally indistinguishable over $n \in \mathbb{N}$.

- $\{\text{View}_V[\langle P(w_n^1), V^*(z_n)\rangle(x_n)]\}_{n\in\mathbb{N}}$
- $\{\text{View}_V[\langle P(w_n^2), V^*(z_n)\rangle(x_n)]\}_{n\in\mathbb{N}}$

3.10. *Proofs and arguments of knowledge*

Given a language $L \in \mathsf{NP}$ and an instance x, a *proof or argument of knowledge* (POK or AOK) not only convinces the verifier that $x \in L$, but also demonstrates that the prover possesses an NP witness for x. This is formalized by the existence of an *extractor*: Given black-box access to a machine that can successfully complete the proof or argument of knowledge on input x, the extractor can compute a witness for x.

Definition 3.8. (*Proofs and arguments of knowledge*) An interactive proof $\Pi = (P, V)$ is a *proof of knowledge* (resp. An interactive argument $\Pi = (P, V)$ is an *argument of knowledge*) of NP language L with respect to witness relation R_L if there exist a positive constant c, a negligible function ν, and a polynomial-time oracle machine E, such that for every interactive machine P^* (respectively, for every polynomial-time machine P^*), every $x \in L$ and every auxiliary input $z \in \{0,1\}^{\mathrm{poly}(|x|)}$, the following holds: On input x and oracle access to $P^*(x, z)$, machine E outputs a string from $R_L(x)$ with probability at least $(\Pr[\langle P^*(z), V\rangle(x) = 1])^c - \nu(|x|)$. The machine E is called the knowledge extractor.

3.11. *Commitments*

Roughly speaking, a commitment scheme enables a party, called the *committer*, to commit itself to a value to another party, the *receiver*. At first the value is hidden from the receiver; this property is called *hiding*. At a later stage when the commitment is opened, it can only reveal a single value as determined in the committing phase; this property is called *binding*. First we define the structure of a commitment scheme.

Definition 3.9. (*Commitment schemes*) A *commitment scheme* is an interactive protocol (C, R) with the following properties:

1. Both the committer C and the receiver R are polynomial-time machines.
2. The commitment scheme has two stages: a commit stage and a reveal stage. In both stages, C and R receive a security parameter 1^n as common input. C additionally receives a private input $v \in \{0,1\}^n$ that is the string to be committed.
3. The commit stage results in a joint output c, called the commitment and a private output d for C, called the decommitment string. Without loss of generality, c can be the full transcript of the interaction between C and R.
4. In the reveal stage, the committer C sends the pair (v, d) to the receiver R, who decides to accept or reject the decommitment (c, v, d) deterministically.

If C and R do not deviate from the protocol, then R should accept (with probability 1) during the reveal stage.

Next we define the binding and hiding property of a commitment scheme.

Definition 3.10. (*Binding*) A commitment scheme (C, R) is *statistically (respectively, computationally) binding* if for every interactive machine (respectively, non-uniform polynomial-size interactive machine) C^* (a malicious committer), there exists a negligible function ν such that C^* succeeds in the following game with probability at most $\nu(n)$:

> On security parameter 1^n, C^* first interacts with R in the commit stage to produce commitment c. Then C^* outputs two decommitments (c, v_0, d_0) and (c, v_1, d_1), and succeeds if $v_1 \in \{0,1\}^n$, $v_2 \in \{0,1\}^n$, $v_1 \neq v_2$ and R accepts both decommitments.

The commitment scheme is perfectly binding if no machine C^* can ever succeed at the above game.

Definition 3.11. (*Hiding*) A commitment scheme (C, R) is *computationally (respectively, statistically)* hiding if for every non-uniform polynomial-size machine (respectively, every interactive machine) R^* (a malicious receiver), every ensemble $\{(v_{n,0}, v_{n,1}, z_n)\}_{n\in\mathbb{N}}$ where $v_{n,0} \in \{0,1\}^n$, $v_{n,1} \in \{0,1\}^n$, and $z_n \in \{0,1\}^{\mathrm{poly}(n)}$, the following ensembles are computationally indistinguishable (resp. statistically close) over $n \in \mathbb{N}$:

$$\{\langle C(v_{n,0}), R^*(z_n)\rangle(1^n)\}_{n\in\mathbb{N}} \approx \{\langle C(v_{n,1}), R^*(z_n)\rangle(1^n)\}_{n\in\mathbb{N}} \ .$$

4. SNARKs

In this section, we formally introduce the main cryptographic primitive studied in this paper—the SNARK.

4.1. *Succinct Non-Interactive Arguments*

A *succinct non-interactive argument* (SNARG) is a triple of algorithms $(\mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$. For a security parameter k, the verifier runs $\mathcal{G}_{\mathcal{V}}(1^k)$ to generate $(\mathsf{vgrs}, \mathsf{priv})$, where vgrs is a (public) verifier-generated reference string and priv are corresponding private verification coins; the honest prover $\mathcal{P}(y, w, \mathsf{vgrs})$ produces a proof Π for the statement $y = (M, x, t)$ given a witness w; then $\mathcal{V}(\mathsf{priv}, y, \Pi)$ verifies the validity of Π. The SNARG is *adaptively sound* if the prover may choose the statement *after* seeing the vgrs, otherwise, it is *non-adaptively sound.*

Definition 4.1. A triple of algorithms $(\mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$ is a SNARG for the relation $\mathcal{R} \subseteq \mathcal{R}_{\mathcal{U}}$ if the following conditions are satisfied:

1. **Completeness** For any $(y, w) \in \mathcal{R}$,

$$\Pr\left[\mathcal{V}(\mathsf{priv}, y, \Pi) = 1 : \begin{array}{c}(\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ \Pi \leftarrow \mathcal{P}(y, w, \mathsf{vgrs})\end{array}\right] = 1 \ .$$

   In addition, $\mathcal{P}(y, w, \mathsf{vgrs})$ runs in time $\mathrm{poly}(k, |y|, t)$.
2. **Succinctness** The length of the proof Π that $\mathcal{P}(y, w, \mathsf{vgrs})$ outputs, as well as the running time of $\mathcal{V}(\mathsf{priv}, y, \Pi)$, is bounded by

$$p(k + |y|) = p(k + |M| + |x| + \log t) \ ,$$

   where p is a universal polynomial that does not depend on $\mathcal{R}$. In addition, $\mathcal{G}_{\mathcal{V}}(1^k)$ runs in time $p(k)$; in particular, $(\mathsf{vgrs}, \mathsf{priv})$ are of length $p(k)$.

Remark 4.2. (The size of the proof) Although this is not required in applications such as delegation, one may further require that the size of the proof Π is a universal polynomial in k also independent of $|y|$. Our constructions satisfy this property.

3. **Soundness** Depending on the notion of adaptivity:
   - **Non-adaptive soundness** For all polynomial-size provers $\mathcal{P}^*$ and polynomials m, any $k \in \mathbb{N}$, and $y \in \{0,1\}^{m(k)} \setminus \mathcal{L}_{\mathcal{R}}$,

$$\Pr\left[\mathcal{V}(\mathsf{priv}, y, \Pi) = 1 : \begin{array}{c} (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ \Pi \leftarrow \mathcal{P}^*(y, \mathsf{vgrs}) \end{array}\right] \leq \mathrm{negl}(k) \ .$$

   - **Adaptive soundness** For all polynomial-size provers $\mathcal{P}^*$ and any $k \in \mathbb{N}$,

$$\Pr\left[\begin{array}{c} \mathcal{V}(\mathsf{priv}, y, \Pi) = 1 \\ y \notin \mathcal{L}_{\mathcal{R}} \end{array} : \begin{array}{c} (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ (y, \Pi) \leftarrow \mathcal{P}^*(\mathsf{vgrs}) \end{array}\right] \leq \mathrm{negl}(k) \ .$$

A SNARG *of knowledge*, or SNARK for short, is a SNARG where soundness is strengthened as follows:

Definition 4.3. A triple of algorithms $(\mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$ is a SNARK if it is a SNARG where adaptive soundness is replaced by the following stronger requirement:

- **Adaptive Proof of Knowledge**[7] For any polynomial-size prover $\mathcal{P}^*$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}_{\mathcal{P}^*}$ such that for any $k \in \mathbb{N}$ and all auxiliary inputs $z \in \{0,1\}^{m(k)}$,

$$\Pr\left[\begin{array}{c} \mathcal{V}(\mathsf{priv}, y, \Pi) = 1 \\ w \notin \mathcal{R}(y) \end{array} : \begin{array}{c} (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ (y, \Pi) \leftarrow \mathcal{P}^*(z, \mathsf{vgrs}) \\ (y, w) \leftarrow \mathcal{E}_{\mathcal{P}^*}(z, \mathsf{vgrs}) \end{array}\right] \leq \mathrm{negl}(k) \ .$$

Universal Arguments Versus Weaker Notions A SNARG for the relation $\mathcal{R} = \mathcal{R}_{\mathcal{U}}$ is called a *universal argument*.[8] A weaker notion that we will also consider is a SNARG for the relation $\mathcal{R} = \mathcal{R}_c$ for a constant $c \in \mathbb{N}$. In this case, soundness is only required to hold with respect to $\mathcal{R}_c$; in particular, the verifier algorithm may depend on c. Nevertheless, we require (and achieve) *universal succinctness*, where a universal polynomial p, independent of c, upper bounds the length of every proof and the verification time.

Designated Verifiers Versus Public Verification In a *publicly verifiable* SNARG, the verifier does not require a private state priv. In this work, however, we concentrate on *designated-verifier* SNARGs, where priv must remain secret for soundness to hold.

The Verifier-Generated Reference String A very desirable property is the ability to generate the verifier-generated reference string vgrs once and for all and then reuse it in polynomially many proofs (potentially by different provers). In publicly verifiable SNARGs, this *multi-theorem soundness* is automatically guaranteed; in designated-verifier SNARGs, however, multi-theorem soundness needs to be required explicitly as an additional property. Usually, this is achieved by ensuring that the verifier's response

[7] One can also formulate weaker proof-of-knowledge notions; in this work, we focus on the above strong notion.

[8] Barak and Goldreich [20] define universal arguments for $\mathcal{R}$ with a black-box "weak proof-of-knowledge" property; in contrast, our proof of knowledge property is not restricted to black-box extractors and does not require the extractor to be an implicit representation of a witness.

"leaks" only a negligible amount of information about priv. (Note that $O(\log k)$-theorem soundness always holds; the "non-trivial" case is obtaining $\omega(\log k)$-theorem soundness. Weaker solutions to support more theorems include assuming that the verifier's responses remain secret, or regenerating vgrs every logarithmically many rejections, e.g., as in [42,65,69,93,94,105].)

The SNARK *Extractor $\mathcal{E}$ and Auxiliary Input* Above, we require that any polynomial-size family of circuits $\mathcal{P}^*$ has a specific polynomial-size family of extractors $\mathcal{E}_{\mathcal{P}^*}$; in particular, we allow the extractor to be of arbitrary polynomial size and to be "more non-uniform" than $\mathcal{P}^*$. In addition, we require that for any prover auxiliary input $z \in \{0,1\}^{\mathrm{poly}(k)}$, the polynomial-size extractor manages to perform its witness extraction task given the same auxiliary input z. As shown in [16,28], this aspect of the definition is too strong assuming indistinguishability obfuscation [22]. The definition can be naturally relaxed to consider only specific distributions of auxiliary inputs according to the required application. (In our setting, the restrictions on the auxiliary input handled by the knowledge extractor will be related to the auxiliary input that the underlying ECRH extractor can handle. See further discussion in Sect. 6.1.)

5. From ECRHs to SNARKs

In this section, we describe and analyze our construction of adaptive SNARKs for NP based on ECRHs. (Recall that an ECRH is a CRH as in Definition 3.1 that is extractable as in Definition 1.)

In Sect. 5.3, we discuss the universal features of our constructions and the difficulties in extending it to a full-fledged universal argument; we propose a solution that can overcome the difficulties based on exponentially hard one-way functions.

Our Modified Approach We modify the PCP+MT+PIR approach of [49] and show that the knowledge assumption of [49] (which involves the entire PIR+MT structure) can be replaced by the simpler generic assumption that ECRHs exist. Furthermore, our modification enables us to improve the result from a two-message succinct argument with non-adaptive soundness to an adaptive SNARG of knowledge (SNARK)—this improvement is crucial in cryptographic applications. Specifically, we perform two modifications:

1. We instantiate the Merkle tree hash using an ECRH and, unlike the traditional construction where a $(2k,k)$-CRH is used, we use a polynomially compressing (k^2,k)-ECRH; in particular, for k^{d+1}-long strings the resulting tree will be of depth d (rather than $d\log k$).[9] As we shall see later, doing so allows us to avoid superpolynomial blowup of the final knowledge extractor that will be built via composition of ECRH extractors. The initial construction we present will be specialized for "generalized" NP relations $\mathcal{R}_c$; after presenting and analyzing it, we propose an extension to the universal relation $\mathcal{R}_{\mathcal{U}}$ by further assuming the existence of exponentially hard one-way functions.
2. In order to ensure that the first message of the verifier does not depend on the theorem being proved, the database that we use does not consist of bits of π;

[9]We note that any $(k^{\varepsilon},k^{\varepsilon'})$-compressing ECRH would have sufficed (for any constants $\varepsilon > \varepsilon'$); for the sake of simplicity, we stick with (k^2,k)-compression.

rather, the rth entry of the database corresponds to the answers to the queries of $V^{\pi}_{\mathsf{pcp}}(y, r)$ with their corresponding authentication paths in the Merkle tree. Here y is chosen by the prover and the authentication is relative to a single string π (to avoid cheating provers claiming one value for a particular location of π in one entry of the database, and another value for the same location of π in another entry of the database). Also, a priori the verifier does not know the exact size of this database (the number of coins that will be needed by the PCP verifier), since this will depend on the prover choice of statement y. However, we do know that the number of coins is bounded by any superlogarithmic function. Thus, the verifier can send PIR encryptions with respect to each of the polylogarithmically many possibilities.

5.1. *Construction Details*

We start by providing a short description of our MT and then present the detailed construction of the protocol in Table 1.

The Shallow Merkle Tree By padding when required, we assume without loss of generality that the compressed string π is of size k^{d+1} (where d is typically unknown to the verifier). A node in the MT of distance j from the root can be represented by a string $\mathbf{i} = i_1 \cdots i_j \in [k]^j$ containing the path traversal indices (and the root is represented by the empty string). We then label the nodes with k-bit strings according to π and the hash $h : \{0,1\}^{k^2} \to \{0,1\}^k$ as follows:

- The leaf associated with $\mathbf{i} = i_1 \cdots i_d \in [k]^d \cong [k^d]$ is labeled by the $\mathbf{i}$th k-bit block of π, denoted by $\ell_{\mathbf{i}}$ (here $\mathbf{i}$ is interpreted as a number in $[k^d]$).
- An internal node associated with $\mathbf{i} = i_1 \cdots i_j \in [k]^j$ is labeled by $h(\ell_{\mathbf{i}1}\ell_{\mathbf{i}2} \cdots \ell_{\mathbf{i}k})$, denoted by $\ell_{\mathbf{i}}$.
- Thus, the label of the root is $h(\ell_1\ell_2 \ldots \ell_k)$, which we denote by ℓ_ϵ.

The MT commitment is the pair (d, ℓ_ϵ). An opening $\mathsf{dcom}_{\mathbf{i}}$ to a leaf $\mathbf{i}$ consists of all the labels of all the nodes and their siblings along the corresponding path. To verify the opening information, evaluate the hash h from the leaves upwards. Specifically, for each node $\mathbf{i}' = \mathbf{i}j$ along the opening path labeled by $\ell_{\mathbf{i}'} = \ell_{\mathbf{i}j}$ and his siblings' labels $\ell_{\mathbf{i}1}, \ell_{\mathbf{i}2}, \ldots, \ell_{\mathbf{i}(j-1)}, \ell_{\mathbf{i}(j+1)}, \ldots, \ell_{\mathbf{i}k}$, verify that $h(\ell_{\mathbf{i}1}, \ldots, \ell_{\mathbf{i}k}) = \ell_{\mathbf{i}}$.

We shall prove the following theorem:

Theorem 5.1. *For any* NP *relation* $\mathcal{R}_c$*, the construction in Table 1 yields a SNARK that is secure against adaptive provers. Moreover, the construction is universally succinct: The proof's length and verifier's running time are bounded by the same universal polynomials for all* $\mathcal{R}_c \subseteq \mathcal{R}_{\mathcal{U}}$.

The completeness of the construction follows directly from the completeness of the PCP and PIR. In Sect. 5.2, we give a security reduction establishing the PoK property (against adaptive provers). In Sect. 5.3, we discuss *universal succinctness* and possible extensions of our construction to a full-fledged universal argument.

Table 1. A SNARK for the relation$\mathcal{R}_c$.

Ingredients:
- •A universal efficient verifier PCPof knowledge $(P_{\mathsf{pcp}}, V_{\mathsf{pcp}}, E_{\mathsf{pcp}})$ for $\mathcal{R}_{\mathcal{U}}$; for $((M, x, t), w) \in \mathcal{R}_{\mathcal{U}}$, where any proof π is s.t. $|\pi| \leq t^2$ and the non-repeated verifier uses $\rho(t) = O(\log t)$ coins and $O(1)$ queries.
- •A succinct PIR (PEnc, PEval, PDec).
- •A (k^2, k)-ECRH$\mathcal{H} = \{\mathcal{H}_k\}_{k\in\mathbb{N}}$.

Setup $\mathcal{G}_{\mathcal{V}}(1^k)$:
- •Generate private verification state:
  - – Sample coins for $q = \omega(\log k)$ repetitions of V_{pcp}: $\mathbf{r} = (r_1, \ldots, r_q) \overset{U}{\leftarrow} \{0,1\}^{(\log^2 k)\times q}$.
  - – Sample coins for encrypting $q \times \log^2 k$ PIR-queries: $R \overset{U}{\leftarrow} \{0,1\}^{\mathrm{poly}(k)}$.
  - –Sample an ECRH: $h \leftarrow \mathcal{H}_k$.
  - – Set $\mathsf{priv} := (h, \mathbf{r}, R)$.
- •Generate corresponding verifier-generated reference string:
  - – For $j \in [\log^2 k]$, let $\mathbf{r}^{(j)} = (r_1^{(j)}, \ldots, r_q^{(j)})$ where $r_i^{(j)}$ is the j-bit prefix of r_i.
  - – Let $\mathbf{r}^* = (\mathbf{r}^{(j)} : j \in [\log^2 k])$
  - – Compute $C_{\mathbf{r}^*} = (C_{\mathbf{r}^{(j)}} : j \in [\log^2 k])$, where $C_{\mathbf{r}^{(j)}} \leftarrow \mathsf{PEnc}_R(1^k, \mathbf{r}^{(j)})$.
  - – Set $\mathsf{vgrs} := (h, C_{\mathbf{r}^*})$.

Proof generation by $\mathcal{P}$:
- •Input: 1^k, vgrs, $(y, w) \in \mathcal{R}_c$ where $y = (M, x, t)$, and $t \leq |x|^c$ is such that $\rho(t) \leq \log^2 k$.
- •Proof generation:
  - – Compute a PCP proof $\pi \leftarrow P_{\mathsf{pcp}}(y, w)$ of size $|\pi| = k^{d+1} \leq t^2$.
  - – Compute an MT commitment for π: $\ell_\epsilon = \mathrm{MT}_h(\pi)$ of depth d.
  - – Compute a database DB with 2^ρ entries; in each entry $r^{(\rho)} \in \{0,1\}^\rho$ store the openings $\mathsf{dcom}_{r^{(\rho)}}$ for all locations of π queried by $V^{\pi}_{\mathsf{pcp}}(y, r^{(\rho)})$.
  - –Compute $C_{\mathsf{dcom}_{\mathbf{r}^{(\rho)}}} \leftarrow \mathsf{PEval}(\mathsf{DB}, C_{\mathbf{r}^{(\rho)}})$.
  - –The proof is set to be $\Pi := (d, \ell_\epsilon, C_{\mathsf{dcom}_{\mathbf{r}^{(\rho)}}})$.

Proof verification by $\mathcal{V}$:
- •Input: 1^k, priv, y, Π, where $y = (M, x, t)$, $\Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}_{\mathbf{r}^{(\rho)}}})$.
- •Proof verification:
  - – Verify that $k^{d+1} \leq t^2 \leq |x|^{2c}$.
  - – Decrypt PIR answers $\mathsf{dcom}_{\mathbf{r}^{(\rho)}} = \mathsf{PDec}_R(C_{\mathsf{dcom}_{\mathbf{r}^{(\rho)}}})$, and verify opened paths (against h and ℓ_ϵ).
  - – Let $\pi|_{\mathbf{r}^{(\rho)}}$ be the opened values in the locations corresponding to $\mathbf{r}^{(\rho)}$. Check if $V_{\mathsf{pcp}}^{\pi|_{\mathbf{r}^{(\rho)}}}(y, \mathbf{r}^{(\rho)})$ accepts.
  - – In case any of the above fail, reject.

[a] V_{pcp}'s queries might be adaptive; such behavior can be simulated by the prover.

[b] This is the single place where the verification algorithm depends on c. See further discussion in Sect. 5.3

5.2. *Proof of Security*

A high-level overview of the proof and main technical challenges are presented in Sect. 1.4. We now turn to the detailed proof, which concentrates on establishing and proving the validity of the knowledge extractor.

Proposition 5.2. (Adaptive proof of knowledge) *For any polynomial-size $\mathcal{P}^*$ and polynomial m there exists a polynomial-size extractor $\mathcal{E}_{\mathcal{P}^*}$, such that for any $k \in \mathbb{N}$ and any* *auxiliary input $z \in \{0,1\}^{m(k)}$:*

$$\Pr_{h,\mathbf{r},R}\left[\begin{array}{c}(y,\Pi)\leftarrow\mathcal{P}^*(z,h,\mathsf{PEnc}_R(\mathbf{r}^*))\\ \mathcal{V}((1^k,h,R,\mathbf{r}),y,\Pi)=1\end{array}\wedge\begin{array}{c}(y,w)\leftarrow\mathcal{E}_{\mathcal{P}^*}(z,h,\mathsf{PEnc}_R(\mathbf{r}^*))\\ w\notin\mathcal{R}_c(y)\end{array}\right]\leq\mathrm{negl}(k)\ ,$$

where $\mathcal{V}$ is the verifier described in Table 1, h is an ECRH, $\mathbf{r}$ *are the PCP coins,* $\mathbf{r}^* = (\mathbf{r}^{(j)} : j \in [\log^2 k])$*, and R are the PIR coins.*

We start by describing how the extraction circuit is constructed and then prove that it satisfies Proposition 5.2. In order to simplify notation, we will address provers $\mathcal{P}^*$ that get as input only $(h, C_{\mathbf{r}^*})$, where $C_{\mathbf{r}^*} = \mathsf{PEnc}_R(\mathbf{r}^*)$; the analysis can be extended to the case that $\mathcal{P}^*$ also gets additional auxiliary input $z \in \{0,1\}^{m(k)}$ for any polynomial m.

The Extraction Procedure As discussed in Sect. 1.4, extraction is done in two phases: First, we recursively extract along all the paths of the Merkle tree (MT); doing so results in a string (of leaf labels) $\tilde{\pi}$; then, we apply to $\tilde{\pi}$ the PCP witness extractor E_{pcp}. As we shall see, $\tilde{\pi}$ will exceed the knowledge threshold ε of the PCP and hence E_{pcp} will produce a valid witness.

We now describe the recursive extraction procedure of the of the ECRH-based MT. Given a polynomial-size prover $\mathcal{P}^*$, let c^* be such that $|\mathcal{P}^*| \leq k^{c^*}$. We derive $2cc^*$ circuit families of extractors, one for each potential level of the MT. Define $\mathcal{E}_1 := \mathcal{E}^{\mathcal{H}}_{\mathcal{P}^*}$ to be the ECRH extractor for $\mathcal{P}^*$; like $\mathcal{P}^*$, $\mathcal{E}_1$ also expects input $(h, C_{\mathbf{r}^*}) \in \{0,1\}^{\mathrm{poly}(k)}$ and returns a string $(\tilde{\ell}_1, \ldots, \tilde{\ell}_k) \in \{0,1\}^{k\times k}$ (which will be a preimage in case $\mathcal{P}^*$ produces a valid image). We can interpret the string output by $\mathcal{E}_1$ as k elements in the range of the hash. Since the ECRH extraction guarantee only considers a single image, we define an augmented family of circuits $\mathcal{E}'_1$ that expects input $(h, C_{\mathbf{r}^*}, i_1)$, where $i_1 \in [k]$, and returns $\tilde{\ell}_{i_1}$, which is the i_1th k-bit block of $\mathcal{E}_1(h, C_{\mathbf{r}^*})$.

Next, we define $\mathcal{E}_2 := \mathcal{E}^{\mathcal{H}}_{\mathcal{E}'_1}$ to be the extractor for $\mathcal{E}'_1$. In general, we define $\mathcal{E}_{j+1} := \mathcal{E}^{\mathcal{H}}_{\mathcal{E}'_j}$ to be the extractor for $\mathcal{E}'_j$, and $\mathcal{E}'_j$ expects an input $(h, C_{\mathbf{r}^*}, \mathbf{i})$, where $\mathbf{i} \in [k]^j$. For each $\mathbf{i} \in [k]^j$, $\mathcal{E}_{j+1}(h, C_{\mathbf{r}^*}, \mathbf{i})$ is meant to extract the labels $\tilde{\ell}_{\mathbf{i}1}, \ldots, \tilde{\ell}_{\mathbf{i}k}$.

Recall, however, that the ECRH extractor $\mathcal{E}_{j+1}$ is only guaranteed to output a preimage whenever the corresponding circuit $\mathcal{E}'_j$ outputs a true image. For simplicity, we assume that in case $\mathcal{E}'_j$ does not output a true image, $\mathcal{E}_{j+1}$ still outputs some string of length k^2 (without any guarantee on this string).

Overall, the witness extractor $\mathcal{E}_{\mathcal{P}^*}$ operates as follows. Given input $(1^k, h, C_{\mathbf{r}^*})$, (a) first invoke $\mathcal{P}^*(h, C_{\mathbf{r}^*})$ and obtain (y, Π); (b) obtain the depth d from Π, and abort if $d > 2cc^*$; (c) otherwise, for each $\mathbf{i} \in [k]^{d-1}$, extract the labels $(\tilde{\ell}_{\mathbf{i}1}, \ldots, \tilde{\ell}_{\mathbf{i}k}) \leftarrow \mathcal{E}_d(h, C_{\mathbf{r}^*}, \mathbf{i})$; (d) letting $\tilde{\pi}$ be the extracted PCP-proof given by the leaves, apply the PCP witness extractor $\tilde{w} \leftarrow E_{\mathsf{pcp}}(y, \tilde{\pi})$ and output $\tilde{w}$.

We now turn to prove that (except with negligible probability), whenever the verifier is convinced, the extractor $\mathcal{E}_{\mathcal{P}_*}$ outputs a valid witness. The proof is divided into two main claims as outlined in Sect. 1.4.

A Reminder and Some Notation Recall that prior to the prover's message, the randomness for the PCP verifier is of the form $\mathbf{r} = (r_i)_{i\in[q]} \in \{0,1\}^{(\log^2 k)\times q}$ (and $q = \omega(\log k)$ is some fixed function). Once the verifier receives (y, Π), where $y = (M, x, t)$ and $\Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}})$, he computes the amount of coins required

$\rho = O(\log t) < \log^2 k$ and uses $r_j^{(\rho)} \in \{0,1\}^\rho$, the ρ-bit prefix of r_j. The corresponding PCP proof π (or the extracted $\tilde{\pi}$) is of size k^{d+1}. We shall denote by $\tilde{\pi} = \mathcal{E}_d(h, \mathsf{PEnc}_R(\mathbf{r}^*)) = \cup_{\mathbf{i}\in[k]^{d-1}} \mathcal{E}_d(h, \mathsf{PEnc}_R(\mathbf{r}^*), \mathbf{i})$ the extraction of the full set of leaves.

Using collision resistance and ECRH extraction, we show that (almost) whenever the verifier is convinced, we extract a proof $\tilde{\pi}$ that locally satisfies the queries induced by the encrypted $\mathsf{PEnc}_R(\mathbf{r}^*)$.

Claim 5.3. (Local consistency) *Let $\mathcal{P}^*$ be a polynomial-size prover strategy, where $|\mathcal{P}| \le k^{c^*}$, and let $(\mathcal{E}_1, \dots, \mathcal{E}_{2cc^*})$ be its ECRH extractors as defined above. Then for any $k \in \mathbb{N}$,*

$$\Pr_{(h,R,\mathbf{r})\leftarrow\mathcal{G}_\mathcal{V}(1^k)}\left[\begin{array}{c}(y,\Pi) \leftarrow \mathcal{P}^*(h, \mathsf{PEnc}_R(\mathbf{r}^*)) \\ y = (M,x,t), \Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}}) \\ \mathcal{V}(1^k, (h,R,\mathbf{r}), y, \Pi) = 1\end{array} \wedge \begin{array}{c}\tilde{\pi} \leftarrow \mathcal{E}_d(h, \mathsf{PEnc}_R(\mathbf{r}^*)) \\ V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}^{(\rho)}) = 0\end{array}\right]$$
$$\le \operatorname{negl}(k) ,$$

where $\mathbf{r} = (r_i : i \in [q])$, $\mathbf{r}^{(j)} = (r_i^{(j)} : i \in [q])$, and $\mathbf{r}^* = (\mathbf{r}^{(j)} : j \in [\log^2 k])$, and ρ is the amount of coins required to verify the statement y.

Proof. Let us say that a tuple $(h, R, \mathbf{r})$ is "bad" if it leads to the above event. Assume toward contradiction that for infinitely many $k \in \mathbb{N}$, there is a noticeable fraction $\varepsilon(k)$ of bad tuples $(h, R, \mathbf{r})$. We show how to find collisions in $\mathcal{H}$.

Given $h \leftarrow \mathcal{H}$, sample coins R for the PIR encryption and coins $\mathbf{r}$ for the PCP verifier to simulate $\mathsf{PEnc}_R(\mathbf{r}^*)$. Given that the resulting $(h, R, \mathbf{r})$ is bad, let us show how to produce a collision in h.

First, invoke $\mathcal{P}^*(h, \mathsf{PEnc}_R(\mathbf{r}^*))$ to obtain (y, Π), where $y = (M, x, t)$ and $\Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}})$. Next, decrypt C_{dcom} (using R) and obtain a set S of $O(q)$ opened paths (each $r_j^{(\rho)}$ in $\mathbf{r}^{(\rho)} = (r_j^{(\rho)} : j \in [q])$ induces a constant amount of queries). Each path corresponds to some leaf $\mathbf{i} \in [k]^d$ and contains d k^2-bit strings $\mathbf{l}_1^\mathbf{i}, \dots, \mathbf{l}_d^\mathbf{i} \in \{0,1\}^{k^2 \times d}$; each string $\mathbf{l}_j^\mathbf{i}$ contains the label of the jth node along the path and the labels of all its siblings.

Next, note that $d \le 2cc^*$. Indeed, if the verifier accepts then: $k^d \le |x|^{2c}$, and in our case $|x| \le |\mathcal{P}^*| \le k^{c^*}$. Accordingly, we can use our extractors as follows: for each opened path in $\mathbf{i} \in S$, where $\mathbf{i} = i_1 \cdots i_d \in [k]^d$, invoke:

$$\begin{array}{l}\mathcal{E}_1(h, \mathsf{PEnc}_R(\mathbf{r}^*)) \\ \mathcal{E}_2(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1) \\ \qquad\vdots \\ \mathcal{E}_d(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1 \cdots i_{d-1})\end{array}$$

and obtain $\tilde{\mathbf{l}}_1^\mathbf{i}, \dots, \tilde{\mathbf{l}}_d^\mathbf{i} \in \{0,1\}^{k^2 \times d}$.

Let $\pi|_S = \left(\mathbf{l}^{\mathbf{i}}_d\right)_{\mathbf{i}\in S}$ be the leaves $\mathcal{P}^*$ opened and let $\tilde{\pi}|_S = \left(\tilde{\mathbf{l}}^{\mathbf{i}}_d\right)_{\mathbf{i}\in S}$ be the extracted leaves. Since $(h, R, \mathbf{r})$ are bad, it holds that $V^{\pi|_S}_{\mathsf{pcp}}(x, \mathbf{r}^{(\rho)}) = 1$ while $\mathcal{V}^{\tilde{\pi}|_S}_{\mathsf{pcp}}(x, \mathbf{r}^{(\rho)}) = 0$; in particular, there exist some $\mathbf{i} \in S$ such that $\mathbf{l}^{\mathbf{i}}_d \neq \tilde{\mathbf{l}}^{\mathbf{i}}_d$. We focus from hereon on this specific $\mathbf{i}$. Let $j \in [d]$ be the smallest index such that $\mathbf{l}^{\mathbf{i}}_j \neq \tilde{\mathbf{l}}^{\mathbf{i}}_j$ (we just established that such an index exists); then it holds that $\mathbf{l}^{\mathbf{i}}_{j-1} = \tilde{\mathbf{l}}^{\mathbf{i}}_{j-1}$. Furthermore, since $(h, R, \mathbf{r})$ are bad, $\mathcal{V}$ accepts; this in turn implies that h compresses $\mathbf{l}^{\mathbf{i}}_j$ to the i_{j-1}th block of $\mathbf{l}^{\mathbf{i}}_{j-1} = \tilde{\mathbf{l}}^{\mathbf{i}}_{j-1}$, which we will denote by ℓ^*. However, the latter is also the output of $\mathcal{E}'_{j-1}(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1 \cdots i_{j-1})$,[10] which in turn implies that $\mathcal{E}_j(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1 \cdots i_{j-1}) = \tilde{\mathbf{l}}^{\mathbf{i}}_j$ is also compressed by h to the same ℓ^* (except when extraction fails, which occurs with negligible probability). It follows that $\mathbf{l}^{\mathbf{i}}_j \neq \tilde{\mathbf{l}}^{\mathbf{i}}_j$ is a collision in h.

The second step in the proof of Proposition 5.2 is to show that if the aforementioned extractor outputs a proof $\tilde{\pi}$ that convinces the PCP verifier with respect to the encrypted randomness, then the same proof $\tilde{\pi}$ must be globally satisfying (at least for witness extraction); otherwise, the polynomial-size extractor can be used to break the semantic security of the PIR.

Claim 5.4. (From local satisfaction to extraction) *Let $\mathcal{P}^*$ be a polynomial-size prover and let $\mathcal{E}_{\mathcal{P}^*}$ be its polynomial-size extractor.*[11]*. Then for any $k \in \mathbb{N}$,*

$$\Pr_{(h,R,\mathbf{r})\leftarrow\mathcal{G}_{\mathcal{V}}(1^k)}\left[\begin{array}{c} t \leq |x|^c \\ V^{\tilde{\pi}}_{\mathsf{pcp}}(y, \mathbf{r}^{(\rho)}) = 1 \\ E_{\mathsf{pcp}}(y, \tilde{\pi}) \notin \mathcal{R}_c(y) \end{array} : \begin{array}{c} (y, \Pi) \leftarrow \mathcal{P}^*(h, \mathsf{PEnc}_R(\mathbf{r}^*)) \\ y = (M, x, t),\, \Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}}) \\ \tilde{\pi} \leftarrow \mathcal{E}_{\mathcal{P}^*}(h, \mathsf{PEnc}_R(\mathbf{r}^*)) \end{array}\right] \leq \mathrm{negl}(k)\ ,$$

where $\mathbf{r} = (r_i : i \in [q])$, $\mathbf{r}^{(j)} = (r^{(j)}_i : i \in [q])$, and $\mathbf{r}^* = (\mathbf{r}^{(j)} : j \in [\log^2 k])$, and ρ is the amount of coins required to verify the statement y.

Proof. Assume toward contradiction that for infinitely many $k \in \mathbb{N}$ the above event occurs with noticeable probability $\delta = \delta(k)$; we show how to break the semantic security of PEnc. First note that whenever the event occurs, it holds that $\Pr_{\mathbf{r}^{(\rho)} \overset{U}{\leftarrow} \{0,1\}^{\rho\times q}}[V^{\tilde{\pi}}_{\mathsf{pcp}}(y, \mathbf{r}^{(\rho)}) = 1] \leq (1-\varepsilon)^q$, where ε is the (constant) knowledge threshold of the PCP (see Sect. 3.7), and $q = \omega(\log k)$ is the number of repetitions.

To break semantic security, we consider the following CPA game, where a breaker $\mathcal{B}$ hands its challenger two independent strings of PCP randomness, $(\mathbf{r}_0, \mathbf{r}_1) \in \{0,1\}^{(\log^2 k)\times 2}$, and gets back $\mathsf{PEnc}_R(\mathbf{r}_b)$ for a random $b \in \{0,1\}$. Now, $\mathcal{B}$ samples a random h, and runs $\mathcal{P}^*(h, \mathsf{PEnc}_R(\mathbf{r}_b))$ and $\mathcal{E}_{\mathcal{P}^*}(1^k, h, \mathsf{PEnc}_R(\mathbf{r}_b))$ to obtain an instance $y = (M, x, t)$ from the prover and an extracted PCP proof $\tilde{\pi}$ from the extractor.

[10]Recall that $\mathcal{E}'_{j-1}(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1 \cdots i_{j-1})$ returns the i_{j-1}th block of the output of $\mathcal{E}_{j-1}(h, \mathsf{PEnc}_R(\mathbf{r}^*), i_1 \cdots i_{j-2})$, which is exactly $\mathbf{l}^{\mathbf{i}}_{j-1}$ and that $\mathcal{E}_j$ is the extractor for $\mathcal{E}'_{j-1}$.

[11]The claim actually holds for any circuit family $\mathcal{E}$, but we'll be interested in the extractor of $\mathcal{P}^*$.

Then, $\mathcal{B}$ computes the amount of coins required for V_{pcp}, $\rho = \rho(t)$ and derives the corresponding ρ-prefixes $(\mathbf{r}_0^{(\rho)}, \mathbf{r}_1^{(\rho)})$ of $(\mathbf{r}_0, \mathbf{r}_1)$.

The breaker now runs the PCP extractor E_{pcp} on input $(y, \tilde{\pi})$ to obtain a string $\tilde{w}$ and verifies whether it is a valid witness for y (which can be done in $\mathrm{poly}(|x|) = \mathrm{poly}(k)$ time). In case the witness $\tilde{w}$ is valid or $V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_0^{(\rho)}) = V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_1^{(\rho)})$, the breaker $\mathcal{B}$ outputs a random guess for the bit b. Otherwise, the breaker outputs the single b' such that $V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{b'}^{(\rho)}) = 1$.

We now analyze the success probability of $\mathcal{B}$. We define two events F and E over a random choice of $(h, R, \mathbf{r}_0^{(\rho)}, \mathbf{r}_1^{(\rho)}, b)$; note that any choice of $(h, R, \mathbf{r}_0^{(\rho)}, \mathbf{r}_1^{(\rho)}, b)$ induces a choice of $y = (M, x, t)$ and $\tilde{\pi}$. Define F to be the event that $t \leq |x|^c$ and $E_{\mathsf{pcp}}(y, \tilde{\pi})$ fails to output a valid witness $\tilde{w}$; next, define E to be the event that $V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_0^{(\rho)}) \neq V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_1^{(\rho)})$.

First, since we have assumed by way of contradiction that the event in the statement of the claim occurs with probability δ, we know that

$$\Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_b^{(\rho)}) = 1 \;\middle|\; F \,\right] = \frac{\delta}{\Pr[F]} \;.$$

Second, since E_{pcp} cannot extract a valid witness from $\tilde{\pi}$ and $\mathbf{r}_{1-b}^{(\rho)}$ are random coins independent of $(\tilde{\pi}, y)$, we also know that

$$\Pr_{\mathbf{r}_{1-b}^{(\rho)} \overset{U}{\leftarrow} \{0,1\}^{\rho \times q}}\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{1-b}^{(\rho)}) = 1 \;\middle|\; F \,\right] \leq (1-\varepsilon)^q \;.$$

Combining these two facts, we deduce that

$$\begin{aligned}
&\Pr\left[\, E \;\middle|\; F \,\right] \\
&\quad\geq \Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_b^{(\rho)}) = 1 \wedge V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{1-b}^{(\rho)}) = 0 \;\middle|\; F \,\right] \\
&\quad\geq \Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_b^{(\rho)}) = 1 \;\middle|\; F \,\right] - \Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{1-b}^{(\rho)}) = 1 \;\middle|\; F \,\right] \\
&\quad\geq \Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_b^{(\rho)}) = 1 \;\middle|\; F \,\right] - (1-\varepsilon)^q = \frac{\delta}{\Pr[F]} - \mathrm{negl}(k) \;,
\end{aligned}$$

so that, in particular, we can also deduce that

$$\Pr\left[\, F \wedge E \,\right] \geq \delta - \mathrm{negl}(k) \;.$$

Therefore,

$$\begin{aligned}
&\Pr\left[\mathcal{B} \text{ guesses } b \;\middle|\; F \wedge E\right] \\
&\quad\geq 1 - \Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{1-b}^{(\rho)}) = 1 \;\middle|\; F \wedge E \,\right] \\
&\quad= 1 - \frac{\Pr\left[\, V_{\mathsf{pcp}}^{\tilde{\pi}}(y, \mathbf{r}_{1-b}^{(\rho)}) = 1 \wedge E \;\middle|\; F \,\right]}{\Pr\left[\, E \;\middle|\; F \,\right]}
\end{aligned}$$

$$\geq 1 - \frac{(1-\varepsilon)^q}{\delta / \Pr\left[\, F \,\right]}$$
$$\geq 1 - \mathrm{negl}(k) \ .$$

We now deduce that the breaker $\mathcal{B}$ guesses b with a noticeable advantage; indeed,

$$\begin{aligned}
\Pr\left[\, \mathcal{B} \text{ guesses } b \,\right] &= \Pr\left[\, F \wedge E \,\right] \Pr\left[\, \mathcal{B} \text{ guesses } b \mid F \wedge E \,\right] + \left(1 - \Pr\left[\, F \wedge E \,\right]\right) \\
&\quad \Pr\left[\, \mathcal{B} \text{ guesses } b \mid \bar{F} \vee \bar{E} \,\right] \\
&= \Pr\left[\, F \wedge E \,\right] \Pr\left[\, \mathcal{B} \text{ guesses } b \mid F \wedge E \,\right] + \left(1 - \Pr\left[\, F \wedge E \,\right]\right) \cdot \frac{1}{2} \\
&= \frac{1}{2} + \Pr\left[\, F \wedge E \,\right] \left(\Pr\left[\, \mathcal{B} \text{ guesses } b \mid F \wedge E \,\right] - \frac{1}{2} \right) \\
&\geq \frac{1}{2} + (\delta - \mathrm{negl}(k)) \left(\frac{1}{2} - \mathrm{negl}(k) \right) \\
&\geq \frac{1}{2} + \frac{\delta - \mathrm{negl}(k)}{2} \ ,
\end{aligned}$$

thus completing the proof of the claim.

Putting it All Together By Claim 5.3, we conclude that whenever the verifier accepts, $\mathcal{E}_{\mathcal{P}^*}$ almost always extracts a proof $\tilde{\pi}$ which locally satisfies the PCP verifier on the encrypted randomness. By Claim 5.4, we deduce that whenever this occurs, $\tilde{\pi}$ must satisfy sufficiently many queries for PCP witness extraction. This completes the proof of Proposition 5.2 and thus of Theorem 5.1.

Efficiency: "Universal Succinctness" For input $y = (M, x, t)$ (where $t < k^{\log k}$ for a security parameter k), the proof $\Pi = (d, \ell_\epsilon, C_{\mathsf{dcom}_{\mathsf{r}(\rho)}})$ is essentially dominated by the PIR answers $C_{\mathsf{dcom}_{\mathsf{r}(\rho)}}$; this includes $q = \mathrm{polylog}(k)$ PIR answers for entries of size $\tilde{O}(k^2)$.[12] In the PIR scheme of [33], the size of each PIR answer is bounded by $E \cdot k \cdot \mathrm{polylog}(k) + \log D$, where E is the size of an entry and D is the number of entries in DB. Hence, the overall length of the proof is bounded by a fixed polynomial $\tilde{O}(k^2)$, independently of $|x|$, $|w|$ or c. The verifier's and prover's running time is bounded, respectively, by fixed universal polynomials $\mathrm{poly}(|y|, k)$, $\mathrm{poly}(k, t)$, again independently of c. Moreover, the size of the proof itself is a universal polynomial in k, also independent of $|y|$.

Parameter Scaling In Kilian's original protocol, succinctness of the proof could be improved by making stronger hardness assumptions. For example, for security parameter k, if one is willing to assume collision-resistant hash functions with a $\mathrm{polylog}(k)$-long output, the proof length would be $\mathrm{polylog}(k)$, rather than $\mathrm{poly}(k)$. Unfortunately, in our construction we use a Merkle tree with $\mathrm{poly}(k)$ fan-in; therefore, we cannot afford the same scaling. Specifically, even if we assume that our hash and PIR scheme have polylogarithmic-size output, each node in the Merkle tree still has $\mathrm{poly}(k)$ siblings.[13]

[12] Recall that $d = \log_k t < \log k$.

[13] We thank Kai-Min Chung and the anonymous referees of ITCS for pointing out the scaling problem.

Nonetheless, scaling can be performed if we make a stronger extractability assumption, such as the interactive one of [52], because in such a case there is no need to consider Merkle trees with polynomial fan-in as binary Merkle trees suffice for the security reduction.

5.3. *Extension to Universal Arguments*

We now discuss the possibility of extending our construction to a full-fledged universal argument, namely an argument for the universal relation $\mathcal{R}_{\mathcal{U}}$ as defined in Sect. 3.6.

Indeed, Theorem 5.1 tells us that for every $c \in \mathbb{N}$ we obtain a specific protocol that is sound with respect to $\mathcal{R}_c$. The dependence on c, however, only appears in the first step of $\mathcal{V}$, where it is checked that $k^{d+1} \leq |x|^{2c}$. In particular, as we already discussed, the running time of both the prover and verifier, as well as the proof length, are universal and do *not* depend on c.

Toward a Full-Fledged Universal Argument To obtain a full-fledged universal argument we might try to omit the above c-dependent size check. However, now we encounter the following difficulty: For the proof of knowledge to go through, we must ensure that the number of recursive extractions is a priori fixed to some constant d (that may depend on the prover). In particular, we need to prevent the prover $\mathcal{P}^*$ from convincing the verifier of statements $y = (M, x, t)$ with $t > k^d$. The natural candidate for d is typically related to the polynomial-size bound on the size of $\mathcal{P}^*$. Indeed, any prover that actually "writes down" a proof of size t should be of size at least t; intuitively, one could hope that being able to convince the verifier should essentially amount to writing down the proof and computing a Merkle hash of it. However, we have not been able to prove this. Instead, we propose the following modification to the protocol to make it a universal argument.

Proofs of Work For the relation $\mathcal{R}_c$, the above problem of $\mathcal{P}^*$ claiming an artificially large t can be avoided by ensuring that the size of a convincing proof t can only be as large as $|x|^c$, where $|x|$ is a lower bound on the prover's size. More generally, to obtain a *universal argument*, we can omit the verifier's check (thus collapsing the family of protocols to a *single* protocol) and enhance the protocol of Table 1 with a *proof of work* attesting that the prover has size at least t^ε for some constant $\varepsilon > 0$. Concretely, if we are willing to make an additional (though admittedly quite strong) assumption, we can obtain such proofs of work:

Theorem 5.5. *If there exist $2^{\varepsilon n}$-hard one-way functions (where n is the input size), then, under the same assumptions as Theorem 5.1, we can modify the protocol in Table 1 to obtain a universal argument.*

Proof sketch. Let $f : \{0, 1\}^* \to \{0, 1\}^*$ a $2^{\varepsilon n}$-hard one-way function. Modify $\mathcal{G}_{\mathcal{V}}(1^k)$ to also output $z_1, \ldots, z_\ell$, with $\ell := \log^2 k$ and $z_i := f(s_i)$, where each s_i is drawn at random from $\{0, 1\}^i$. Then, when claiming a proof Π for an instance $y = (M, x, t)$, the prover must also present s'_i such that $f(s'_i) = z_i$ where $i > \log t$.[14] The verifier $\mathcal{V}$ can

[14] Note that in any case the verifier will reject any claim for t above the superpolynomial universal bound $2^{\log^2 k}$; hence, $\ell = \log^2 k$ challenges are sufficient for any polynomial-size prover.

easily check that this is the case by evaluating f. (Note also that the honest prover will have to pay at most an additive factor of $\tilde{O}(t)$ in its running time when further requested to present this challenge.) Then, by the hardness of f, we know that if the prover has size k^d, then it must be that $k^d > 2^{\varepsilon i} > t^{\varepsilon}$, so that we conclude that $k^{d/\varepsilon} > t$. Therefore, in the proof of security, we know that the claimed depth of the prover is a constant depending only on d and ε, and thus the same proof of security as that of Theorem 5.1 applies.

Admittedly, assuming exponentially hard one-way functions is unsatisfying, and we hope to remove the assumption with further analysis; in the meantime, we would like to emphasize that this assumption has already been made, e.g., in natural proofs [120] or in works that improve PRG constructions [85].

6. ECRHs: a Closer Look

In this section, we take a closer look at the notion of ECRH and propose relaxations of this notion that still suffice for constructing SNARKs. These relaxations are crucial to expand our set of candidate constructions; for more details on the constructions, see Sect. 8.

6.1. *ECRHs*

We begin by discussing several technical aspects regarding the definition of ECRH. Recall that an ECRH is a collision-resistant function ensemble $\mathcal{H}$ that is extractable in the sense of Definition 1, which we reproduce:

Definition 6.1. An efficiently samplable function ensemble $\mathcal{H} = \{\mathcal{H}_k\}_k$ is an $(\ell(k), k)$-*compressing* ECRH if it is $(\ell(k), k)$-compressing, collision-resistant, and moreover *extractable*: For any polynomial-size adversary $\mathcal{A}$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$, such that for any $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$:

$$\Pr_{h \leftarrow \mathcal{H}_k}\left[\begin{array}{c} y \leftarrow \mathcal{A}(h,z) \\ \exists\, x : h(x) = y \end{array} \wedge \begin{array}{c} x' \leftarrow \mathcal{E}^{\mathcal{H}}_{\mathcal{A}}(h,z) \\ h(x') \neq y \end{array}\right] \leq \operatorname{negl}(k)\ . \qquad (1)$$

In other words, the only way an adversary $\mathcal{A}$ can sample elements in the image of the hash is by knowing a corresponding preimage (which an extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$ could in principle find).

Image Verification In known applications of extractable primitives (e.g., three-round zero knowledge [27,37,87]), an extra image-verifiability feature is required. Namely, given $y \in \{0,1\}^k$ and h, one should be able to efficiently test whether $y \in \mathsf{Image}(h)$. Here, there are two flavors to consider: (a) public verifiability, where to verify an image all that is required is the (public) seed h; and (b) private verifiability; that is, the seed h is generated together with private verification parameters priv, so that anyone in hold of priv may perform image verification. We emphasize that our main

ECRH-based construction (presented in Sect. 5.1) *does not require any verifiability features*.

Necessity of Sparseness For the compressing family $\mathcal{H}$ to be collision-resistant, it must also be one-way [77]; namely, the image distribution

$$\mathcal{I}_h = \left\{ h(x) : x \overset{U}{\leftarrow} \{0,1\}^{\ell(k)} \right\}$$

should be hard to invert (except with negligible probability over h). In particular, $\mathcal{I}_h$ must be very far from the uniform distribution over $\{0,1\}^k$ (for almost all h).

Indeed, suppose that the statistical distance between $\mathcal{I}_h$ and uniform is $1 - 1/\mathrm{poly}(k)$ and consider an adversary $\mathcal{A}$ that simply outputs range elements $y \in \{0,1\}^k$ uniformly at random, and any $\mathcal{E}^{\mathcal{A}}_{\mathcal{H}}$. In this case, there is no "knowledge" to extract from $\mathcal{A}$, so $\mathcal{E}^{\mathcal{A}}_{\mathcal{H}}$ has to invert uniformly random elements of the range $\{0,1\}^k$. Thus, the success probability of $\mathcal{E}^{\mathcal{A}}_{\mathcal{H}}$ will differ by at most $1 - 1/\mathrm{poly}(k)$ from its success probability had the distribution been $\mathcal{I}_h$, which is negligible (by one-wayness); hence $\mathcal{E}^{\mathcal{A}}_{\mathcal{H}}$ will still fail with probability $1 - 1/\mathrm{poly}(k)$, thereby violating Equation (1).

A simple way to ensure that the image distribution $\mathcal{I}_h$ is indeed far from uniform is to make the support of $\mathcal{I}_h$ sparse. We will take this approach when constructing candidates, making sure that all $h(x)$ fall into a superpolynomially sparse subset of $\{0,1\}^k$: $\mathsf{Image}(h) < 2^{k-\omega(\log k)}$ (except with negligible probability over $h \leftarrow \mathcal{H}_k$).

Of course, this merely satisfies one necessary condition and is a long way off from implying extractability. Still, this rules out one of the few generic attacks about which we can reason without venturing into the poorly charted territory of non-black-box extraction. Moreover, the sparseness (or more generally, statistical distance) requirement rules out many natural constructions; for example, traditional cryptographic CRH ensembles, and heuristic constructions such as the SHA family have an image distribution $\mathcal{I}_h$ that is close to uniform (by design) and are thus not extractable.

On Auxiliary Input The ECRH definition requires that for any adversary auxiliary input $z \in \{0,1\}^{\mathrm{poly}(k)}$, the polynomial-size extractor manages to perform its extraction task given the same auxiliary input z. As observed by Hada and Tanaka [87], and by Goldreich [79], this requirement is rather strong considering the fact that z could potentially encode arbitrary circuits. Indeed, [16,28] show that such a definition cannot be satisfied, assuming indistinguishability obfuscation [22]. The mentioned impossibility results strongly rely on the fact that the auxiliary input can be of arbitrary polynomial size, and can be taken from an arbitrary distribution.

While for presentational purposes the above definition may be simple and convenient, for our main application (i.e., SNARKs) we can actually settle for a definition that is weaker in both aspects—it can be restricted to a specific "benign distribution" on auxiliary inputs, which is also of a priori bounded polynomial size, a setting in which no negative results are known. (Naturally, when relying on ECRHs with respect to such restricted auxiliary input distributions, the resulting SNARK will account for the same auxiliary input distributions.)

Specifically, in our setting the extractor is required to handle an auxiliary input (C, I) consisting of (honestly generated) PIR encryptions C of random strings, whose length is a priori bounded by a fixed polynomial $\mathrm{poly}(k)$ in the security parameter k, and a short path index I of length $O(\log k)$. The logarithmically short auxiliary input I can be shown not to give any additional power to the adversary. Specifically, an ECRH for auxiliary input distribution C is also an ECRH for auxiliary input (C, I) as long as $|I| = O(\log k)$. (Roughly, the extractor of an adversary $\mathcal{A}$ for auxiliary input (C, I) can be constructed from a polynomial number of extractors, obtained by considering the extractor of $\mathcal{A}_I(C) := \mathcal{A}(C, I)$ for every possible value of I; for the formal argument, one has to adapt this intuition to circuit families.)

As for the "main" part C of the auxiliary input, we note that by using a PIR scheme where honestly generated ciphers are pseudo-random, we can actually restrict the "main" part C of the auxiliary input to be a random string (e.g., the LWE-based PIR scheme of [33] has this property), as long as the ECRH has efficient (even private) image verification. Indeed, an extractor that does noticeably better for random auxiliary input than for pseudo-random auxiliary input can be used to distinguish the two cases. For this, we should be able to efficiently check when that adversary produces an actual image (whereas the extractor fails to produce a corresponding preimage), which is why efficient image verification is needed.

6.2. *PECRHs*

As discussed in Sect. 1.3.1, our first weakening of ECRH, namely proximity ECRH, captures a more flexible trade-off between the requirements of extractability and collision resistance. Formally:

Definition 6.2. An efficiently samplable function ensemble $\mathcal{H} = \{\mathcal{H}_k\}_k$ is an $(\ell(k), k)$-*compressing* PECRH if it is $(\ell(k), k)$-compressing and, for every h in the support of $\mathcal{H}_k$, there exist a reflexive "proximity relation" $\overset{h}{\approx}$ over pairs in $\{0,1\}^k \times \{0,1\}^k$, an "extended domain"$D_h \supseteq \{0,1\}^{\ell(k)}$, and an extension $\bar{h} : D_h \to \{0,1\}^k$ consistent with h (i.e., $\forall x \in \{0,1\}^{\ell(k)}$ it holds that $h(x) = \bar{h}(x)$), such that:

1. $\mathcal{H}$ is *proximity extractable* in the following weakened sense: For any polynomial-size adversary $\mathcal{A}$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$ such that for any security parameter $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$:

$$\Pr_{h \leftarrow \mathcal{H}_k}\left[\begin{array}{c} y \leftarrow \mathcal{A}(h, z) \\ \exists\, x \in \{0,1\}^{\ell(k)} : y = h(x) \end{array} \wedge \begin{array}{c} x' \leftarrow \mathcal{E}^{\mathcal{H}}_{\mathcal{A}}(h, z) \\ \neg\left(x' \in D_h \;\wedge\; \bar{h}(x') \overset{h}{\approx} y\right) \end{array}\right] \le \mathrm{negl}(k) \ .$$

2. $\mathcal{H}$ is *proximity-collision-resistant* in the following strengthened sense: For any polynomial-size adversary $\mathcal{A}$,

$$\Pr_{h \leftarrow \mathcal{H}_k}\left[(x, x') \leftarrow A(h) \;\wedge\; x, x' \in D_h \;\wedge\; x \neq x' \;\wedge\; \bar{h}(x) \overset{h}{\approx} \bar{h}(x')\right] \le \mathrm{negl}(k) \ .$$

We now discuss why any point on the trade-off (i.e., any choice of $\overset{h}{\approx}$, D_h and $\bar{h}$ fulfilling the conditions) suffices for the construction of SNARKs as claimed in Theorem 5 in Sect. 1.3.1.

Proof sketch for Theorem 5. We argue that the *same construction* the we use in the proof of Theorem 1 to construct SNARKs from ECRHs still suffices even when the underlying hash function is only a PECRH.

First, observe that moving from ECRHs to PECRHs only affects the "local consistency" step of our proof (as described in our high-level description in Sect. 1.4 and then formally as Claim 5.3). Indeed, in the proof based on ECRHs, the local consistency step is where we employ collision resistance to claim that the Merkle tree output by the extractor locally agrees with the opened paths (except with negligible probability).

The same argument holds. By the proximity extraction guarantee, it must be that the hash image of every node label that appears in an opened path is "close" to the image of the corresponding node label in the extracted tree. By the proximity collision resistance, however, these two node labels must in fact *be the same*; for if they were not, then we could utilize the prover and extractor for finding "proximity collisions." The rest of the proof of Theorem 1 remains unchanged.

We emphasize that the proximity relation $\overset{h}{\approx}$ need not be efficiently computable for the above to hold.

6.3. *Weak PECRHs*

As discussed in Sect. 1.3.2, our second weakening of ECRH, namely weak PECRH, relaxes the condition that determines when the extractor has to work. Formally:

Definition 6.3. An efficiently samplable function ensemble $\mathcal{H} = \{\mathcal{H}_k\}_k$ is a $(\ell(k), k)$-*compressing weak* PECRH if it satisfies Definition 6.2 with the following modified first condition:

1. $\mathcal{H}$ is *weak proximity extractable* in the following sense: For any polynomial-size adversary $\mathcal{A}$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}^{\mathcal{H}}_{\mathcal{A}}$ such that for any security parameter $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$ and polynomial-size decoder circuit $\mathcal{Y}$:

$$\Pr_{h \leftarrow \mathcal{H}_k}\left[\begin{array}{c} (y,e) \leftarrow \mathcal{A}(h,z) \\ h(\mathcal{Y}(e)) = y \end{array} \wedge \neg \begin{array}{c} x' \leftarrow \mathcal{E}^{\mathcal{H}}_{\mathcal{A}}(h,z) \\ \left(x' \in D_h \;\wedge\; \bar{h}(x') \overset{h}{\approx} y\right) \end{array}\right] < \mathrm{negl}(k) \ .$$

We show that even weak PECRHs are enough for the construction of SNARKs as claimed in Theorem 7.

Proof sketch of Theorem 7. We argue that the *same construction* that we use in the proof of Theorem 1 to construct SNARKs from ECRHs also obtains SNARKs even when the underlying hash function is only a weak PECRH. As was the case when moving from ECRHs to PECRHs, moving from PECRHs to weak PECRHs only affects the "local consistency" step of our proof (as described in our high-level description in Sect. 1.4 and then formally as Claim 5.3); specifically, we must still be able to guarantee local consistency even when the condition under which the extractor is guaranteed to work is weakened to the case where the adversary outputs an encoding of a preimage (as opposed to when the adversary merely outputs a value in the image).

In the construction, this is always the case, because preimages along the opened path are provided as encrypted authentication paths, which can be "decoded" by a decoder that knows the secret key (i.e., the PIR private coins). Therefore, we are still able to show local consistency.

Unlike PECRHs, weak PECRHs may in principle not require sparsity of the image or special algebraic structure. While an attacker trying to fool a PECRH extractor only has to obliviously sample an image of the function, now, to fool a weak PECRH extractor, it needs to simultaneously obliviously sample an image of the function and an encoding of a corresponding preimage. This raises the following natural question: "is any CRH also a weak PECRH?" We believe this to be unlikely; indeed, the following example shows that (assuming the existence of one-way permutations) there exists a CRH that is not a weak PECRH when the proximity relation is forced to be equality. (To extend this to an actual counter-example, one would have to rule out all proximity relations.)

Let $\mathcal{H} = \{\mathcal{H}_k\}$ be any CRH mapping $\{0, 1\}^{2k}$ to $\{0, 1\}^k$ and P any one-way permutation mapping $\{0, 1\}^k$ to itself, for every $k \in \mathbb{N}$. Consider the (contrived) new CRH $\mathcal{H}'$, mapping $\{0, 1\}^{2k}$ to $\{0, 1\}^{k+1}$, that is defined as follows. A seed $h' \in \mathcal{H}'_k$ corresponds to a seed $h \in \mathcal{H}_k$; each $(x_1||0^k) \in \{0, 1\}^k \times \left\{0^k\right\}$ is mapped by h' to $(0||P(x_1))$ and each $(x_1||x_2) \in \{0, 1\}^k \times (\{0, 1\}^k \setminus \left\{0^k\right\})$ is mapped by h' to $(1||h(x_1||x_2))$.

Since P is one-to-one and the intersection of the image set $h'(\{0, 1\}^k \times \left\{0^k\right\})$ with the image set $h'(\{0, 1\}^k \times (\{0, 1\}^k) \setminus \left\{0^k\right\})$ is empty, any collision in h' implies a corresponding collision in h and, therefore, $\mathcal{H}'$ is also a CRH (i.e., a proximity CRH relative to the equality proximity relation). However, $\mathcal{H}'$ is not weakly proximity extractable relative to the equality proximity relation; indeed, consider the auxiliary input distribution $z = P(x_1)$ for a random $x_1 \leftarrow \{0, 1\}^k$, with a corresponding (correlated) decoder $\mathcal{Y}$ that always outputs $x_1||0^k$. In addition, consider a dummy adversary that given z, h simply outputs $(0||z) = (0||P(x_1)) = h(x_1||0^k)$. Note that since the decoder always outputs a valid preimage, any extractor would have to do the same. However, any efficient extractor that manages to do so has to invert the one-way permutation P.

7. From SNARKs to PECRHs (and More)

In this section, we provide more details about Theorems 2 and 3, which were only informally stated in the introductory discussion of Sect. 1.2. That is, we show that (proximity) extractable collision-resistant hash functions (PECRHs) are in fact not only sufficient (together with appropriate polylog PIRs) but also necessary for SNARKs (assuming standard CRHs). We then describe a general method for obtaining additional (non-interactive) extractable primitives.

Extractability and Proximity Extractability We say that a function ensemble $\mathcal{F} = \{\mathcal{F}_k\}_k$ is *extractable* if, given a random $f \leftarrow \mathcal{F}_k$, it is infeasible to produce $y \in \mathsf{Image}(f)$ without actually "knowing" $x \in \mathsf{Domain}(f)$ such that $f(x) = y$. This is formalized by the requirement that for any polynomial-size adversary $\mathcal{A}$ there is a polynomial-size extractor $\mathcal{E}_\mathcal{A}$ such that for any auxiliary input z and randomly chosen f: If $\mathcal{A}(z, f)$ outputs a proper image, $\mathcal{E}_\mathcal{A}(z, f)$ outputs a corresponding preimage. Typically, for such a family to be interesting, it is required that $\mathcal{F}$ also has some hardness property, e.g., one-wayness; in particular, for the two features (of hardness and extractability) to coexist, $\mathsf{Image}(f)$ must be sparse in the (recognizable) range of the function.

As explained (for ECRHs) in Sect. 1.3.1 and later in Sect. 6.1, the above notion of extraction can be relaxed to consider proximity extraction, according to which it is infeasible to produce $y \in \mathsf{Image}(f)$ without knowing an x such that $f(x)$ is proximate to y relative to some given reflexive proximity relation (e.g., relative hamming distance at most $1/2$). For such a notion of extraction to be useful, the corresponding hardness of f should be upgraded accordingly. For example, a *proximity extractable one-way function* $(f, \approx)$, where $\approx$ is some proximity relation, is such that, given $f(x)$ for a random x, it is infeasible to find an x' for which $f(x') \approx f(x)$ (and it is proximity extractable in the sense that we just described).

In Sect. 6, we discussed even further relaxations: In one, the extractor also had the freedom to output elements x from an extended domain $D_f \supseteq \mathsf{Domain}(f)$; in another, the extractor only had to work when the adversary manages to output not only an image but also an encoding of a corresponding preimage. In this section, however, we do not consider these further relaxations.

Note that, naturally, known cryptographic schemes (e.g., three-round zero knowledge) have relied on standard extraction rather than proximity extraction; however, to the best of our knowledge we can safely replace standard extraction in these schemes with proximity extraction (as we did for the purpose of constructing SNARKs).

Verification and Proximity Verification Another extraction-related issue is *image verification*; here, there are two notions that can be considered:

- *Public verification:* Given f and $y \in \mathsf{Range}(f)$, one can efficiently test whether $y \in \mathsf{Image}(f)$.
- *Private verification:* Together with the function f, $\mathcal{F}_k$ also generates a private verification state priv_f. Given f, priv_f and $y \in \mathsf{Range}(f)$, one can efficiently test whether $y \in \mathsf{Image}(f)$.

In addition, when considering proximity extractability, we can consider a corresponding notion of *proximity verifiability*, where the verifier should only check whether $y \in_{\approx} \mathsf{Image}(f)$, namely there is some element $y' \in \mathsf{Image}(f)$ for which $y \approx y'$. Again we note that for the known applications of (verifiable) extractable primitives, proximity verification is sufficient.

Also note that the weaker notion of *extractability with no efficient verification* might also be meaningful in certain scenarios. Indeed, for our main ECRH-based construction of SNARKs (presented in Sect. 5.1), this weak notion of extractability with no efficient verification suffices and in fact ultimately allows us to deduce, through the SNARK construction, many extractable primitives with efficient private (proximity) verification.

7.1. *From SNARKs to PECRHs*

We now present the implications of SNARKs to the existence of extractable primitives, starting with the necessity of PECRHs:

Proposition 7.1. *If there exist SNARKs and (standard) CRHs, then there exist proximity verifiable PECRHs.*

Proof sketch. We show that designated-verifier SNARKs imply PECRHs with private proximity verification. The proof can be easily extended to the case of public verifiability (where publicly verifiable SNARKs imply PECRHs with public proximity verification). Let $\mathcal{H}$ be a (tk, k)-compressing CRH, where $t = t(k) > 1$ is a compression parameter. Let $(\mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$ be an (adaptive) SNARK such that, given security parameter $\hat{k}$, the length of any proof is bounded by $\hat{k}^c$.[15] We define a $(tk, 2k)$-compressing ECRH, $\widetilde{\mathcal{H}} = \{\widetilde{\mathcal{H}}_k\}_k$. A function $\tilde{h}$ and private verification state $\mathsf{priv}_{\tilde{h}}$ are sampled by $\widetilde{\mathcal{H}}_k$ as follows:

1. Draw a function $h \leftarrow \mathcal{H}_k$,
2. Draw public and private parameters $(\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(k^{1/c})$, and
3. Set $\tilde{h} = (h, \mathsf{vgrs})$, $\mathsf{priv}_{\tilde{h}} = \mathsf{priv}$.

Then, for an input x and defining $y = h(x)$, we define $\tilde{h}(x) = (y, \Pi)$ where $\Pi = \mathcal{P}(\mathsf{vgrs}, \mathsf{thm}, x)$ is a proof of knowledge for the NP statement thm = "there exists an $x \in \{0,1\}^{tk}$ such that $h(x) = y$."

We now show that the above is a PECRH with respect to the relation $\approx$, where $(y, \Pi) \approx (y', \Pi')$ if and only if $y = y'$.

The proximity collision resistance of $\widetilde{\mathcal{H}}$ follows directly from the collision resistance of $\mathcal{H}$, because any proximity-colliding pair (x, x') for $\widetilde{\mathcal{H}}$ is a colliding pair for $\mathcal{H}$. The proximity extractability property of $\widetilde{\mathcal{H}}$ follows from the (adaptive) proof of knowledge of the SNARK $(\mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$; that is, for any image-computing polynomial-size adversary $\mathcal{A}$, the ECRH extractor is set to be the SNARK witness extractor $\mathcal{E}_{\mathcal{A}}$. In addition, an image can be proximity-verified (with respect to $\approx$) by invoking the SNARK verifier $\mathcal{V}$ with the private verification state priv, the proof Π, and the corresponding statement. We note that, for the proposition to go through, it is crucial for the SNARK to hold against adaptive provers; indeed, the adversary gets to choose on which inputs to compute the hash function, and these may very well depend on the public parameters.

Why PECRH *and not* ECRH? At first glance, it is not clear why the above does not imply an ("exact") ECRH rather than a PECRH. The reason lies in the fact that the extractor is only guaranteed to output one of many possible witnesses (preimages). In particular, given an honest image $(h(x), \Pi_x)$ (corresponding to some preimage x), the extractor may output x' such that $h(x') = h(x)$ but applying the honest prover to x' (or, more precisely, the NP statement proving knowledge of x') results in a proof $\Pi_{x'} \neq \Pi_x$.

We now can immediately deduce that SNARKs also imply *proximity extractable one-way functions* (PEOWFs) and *proximity extractable computationally binding and hiding commitments* (PECOMs). These are one-way functions and commitments (respectively) in the standard sense with an additional extraction property: extraction of a (proximate) preimage in the one-way function case and (proximity) extraction of a plain text in the commitment case.

[15]More precisely, the length of any proof is bounded by $(\hat{k} + \log t)^c$, where t is the computation time; however, we only address statements where the computation is poly-time and in particular $\log t < \hat{k}$.

Corollary 7.2. *If there exist SNARKs and (standard) CRHs, then there exist PEOWFs and PECOMs. Moreover, the verifiability features of the SNARK carry over to the implied primitives.*

Proof Sketch. First, note that any $(\ell(k), k)$-compressing PECRH is also a (keyed) PEOWF (with respect to the same image proximity relation). Indeed, it is a proximity def since it is a proximity CRH and independently of that it is also proximity extractable (and verifiable).

Second, to get an extractable bit-commitment scheme, one can use the classic CRH plus hardcore bit construction of Blum [26]. Specifically, the commitment scheme is keyed by a seed h for the PECRH and a commitment to a bit b is obtained by sampling $r, s \overset{U}{\leftarrow} \{0,1\}^{\ell(k)}$ and computing

$$\mathsf{Eval}_{\mathsf{Com}}(b; h; r, s) := (h(r), s, b \oplus \langle r, s \rangle) \ .$$

The fact that this is a computationally binding and hiding commitment holds for any CRH (note that any proximity CRH is in particular a CRH). Moreover, any adversary that computes a valid commitment $c = (y, s, b)$ (under the random seed h) also computes a valid image y under h; hence, we can use the PECRH extractor to extract the commitment randomness $\hat{r}$, such that $y \approx \hat{y} = h(\hat{r})$. We can accordingly define the proximity relation on commitments: $(y, s, b) \approx (\hat{y}, \hat{s}, \hat{b})$ if and only if $y \approx \hat{y}$, $s = \hat{s}$, and $b = \hat{b}$.

In addition, verifying a proximity commitment is done by verifying that y is proximate to an image under h.

7.2. *From Leakage-Resilient Primitives and SNARKs to Extractable Primitives*

Given the results in the previous section, naïvely, it seems that non-interactive adaptive arguments of knowledge offer a generic approach toward constructing extractable primitives: "simply add a non-interactive proof of preimage knowledge" (which might seem to be applicable even without succinctness when compression is not needed). However, this approach may actually compromise privacy because the attached proofs may leak too much information about the preimage.

One may try to overcome this problem by using non-interactive *zero-knowledge* proofs of knowledge. However, this can only be done in the common reference string model and will accordingly lead to extractable functions in the common reference string model. However, when adding a common reference string, some of the applications of extractable functions (for instance, three-message zero knowledge presented in Sect. 11.2) become trivial.

In this section, we consider a different approach toward overcoming the problem of proof-induced preimage leakage: We suggest to consider stronger (non-extractable) primitives that are resilient to bounded amounts of leakage on the preimage. Then, we can leverage the succinctness of SNARKs to claim that proving knowledge of a preimage does not leak too much and hence does not compromise the security of the primitive. Indeed, CRHs are in a sense optimally leakage-resilient defs; hence, the first part of Corollary 7.2 can be viewed as an application of this paradigm. Moreover, in this

approach, there is no need to assume a trusted third party to set up a common reference string (as would be the case if we were to use zero-knowledge techniques).
We exemplify how to apply this paradigm to subexponentially hard defs.

Proposition 7.3. *Given any $2^{|x|^{\varepsilon}}$-hard def $f : \{0,1\}^* \to \{0,1\}^*$ and SNARKs, there exist PEOWFs (against polynomial-size adversaries). Moreover, the verifiability features of the SNARK carry over to the implied PEOWF.*

The proposition follows directly from the following two claims.

Claim 7.4. (Leakage resilience) *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a $2^{|x|^{\varepsilon}}$-hard def. Then f is $|x|^{0.99\varepsilon}$-leakage-resilient in the sense that for any function $\ell(\cdot)$ such that $|\ell(x)| \leq |x|^{0.99\varepsilon}$ no polynomial-size adversary given $(f(x), \ell(x))$ for a random x, can find a preimage of f.*

Claim 7.5. (From leakage resilience and SNARK to Extractable OWFs) *Assume the existence of SNARKs and a OWF $f : \{0,1\}^* \to \{0,1\}^*$ that is $|x|^{\delta}$-leakage-resilient for some constant δ. Then there is a PEOWF. The verifiability features of the SNARK carry over to the implied PEOWF.*

Proof Sketch. The first claim follows directly from the fact that f is $2^{|x|^{\varepsilon}}$-hard. Any polynomial-size adversary that inverts $f(x)$, given $\ell(x)$, can be transformed to a $2^{|x|^{\varepsilon}}$-size adversary that inverts f by simply enumerating all possible values $\ell(x)$.

We now prove the second claim. As in Proposition 5.2, we define an extractable function $\mathcal{F} = \{\mathcal{F}_k\}_k$. Let c be the constant such that any SNARK proof is bounded by $\hat{k}^c$ for security parameter $\hat{k}$. The functions generated by $\mathcal{F}_k$ are defined on the domain $\{0,1\}^{k^{c/\delta}}$ and are indexed by a $\mathsf{vgrs} \leftarrow \mathcal{G}_{\mathcal{V}}(1^k)$. For $x \in \{0,1\}^{k^{c/\delta}}$, $f_{\mathsf{vgrs}}(x) = (f(x), \Pi)$, where Π is a SNARK for the statement that "there exists an $x \in \{0,1\}^{k^{c/\delta}}$ such that $f(x) = y$." As for PECRHs, we define the proximity relation to be $(y, \Pi) \approx (y', \Pi')$ if and only if $y = y'$. As in the proof of Proposition 7.1, proximity extraction and verifiability follow directly from the extraction and verifiability of the SNARK. We claim that $\mathcal{F}$ is one-way with respect to $\approx$; namely, given the image (y, Π) of a random x in the domain, it is infeasible to come up with an x' such that $f(x') = y$. This follows directly from the fact that f can withstand leakage of size $(k^{c/\delta})^{\delta} = k^c$ and the fact that $|\Pi| \leq k^c$.

Note that, given SNARK with proof size polylog(k), one can start from f that is only hard against quasi-polynomial-size adversaries. (As noted in Sect. 5.3 our SNARK security proof does not scale in this way, but given stronger extractability assumptions it would.) We also note that the above reduction essentially preserves the structure of the original def f; in particular, if f is one-to-one so is $\mathcal{F}$. Moreover, in such a case in the NP language corresponding to f, any theorem, claiming that $y = f(x)$ is a proper image, has a single witness x. In this case, we would get (exact) EOWF rather than PEOWF. We thus get:

Corollary 7.6. *Given any $2^{|x|^{\varepsilon}}$-hard one-to-one def and SNARKs, there exist (exactly) extractable commitments that are perfectly binding and computationally hiding (against polynomial-size adversaries).*

Proof sketch. Indeed, the EOWFs given by Proposition 7.3 would now be one-to-one, which in turn imply perfectly binding ECOMs, using the hardcore bit construction as in Corollary 7.2 instantiated with a one-to-one EOWF. (The fact that one-to-one EOWFs imply perfectly binding ECOMs was already noted in [37].)

More Extractable Primitives Based on SNARK*s and Leakage Resilience* We believe there is room to further investigate the above approach toward obtaining more powerful extractable primitives. In this context, one question that was raised by [37] is whether extractable *pseudo-random generators* and *pseudo-random functions* can be constructed from generic extractable primitives, e.g., EOWFs. (They show that the generic constructions of [86] are not knowledge preserving.)

Our SNARK-based approach can plausibly be used to obtain two weaker variants, namely extractable *pseudo-entropy generators* and *pseudo-entropy functions*. Specifically, the results of [55,84,121] imply that any strong enough PRG is inherently also leakage-resilient, in the sense that, even given leakage on the seed, the PRG's output still has high pseudo-entropy (specifically, *HILL entropy*). The results of Braverman et al. [24] show how to obtain the more general notion of leakage-resilient pseudo-entropy functions. We leave the investigation of these possibilities for future work.

Non-Verifiable Extractable Primitives Perfectly binding ECOMs (as given by Corollary 7.6) provide a generic way of obtaining limited extractable primitives that do not admit efficient image verification (and if compression is needed SNARKs can be used on top). Specifically, one can transform a function $\mathcal{F}$ to an extractable $\widetilde{\mathcal{F}}$ as follows. The seed $\tilde{f}_{(f,g)}$ generated by $\widetilde{\mathcal{F}}_k$ includes $f \leftarrow \mathcal{F}_k$ and a seed for the perfectly binding ECOM $g \leftarrow \mathsf{Gen}_{\mathsf{Com}}(1^k)$. To apply the sampled function on x, sample extra randomness r for the commitment, and define $\tilde{f}_{(f,g)}(x;r) = (f(x), \mathsf{Eval}_{\mathsf{Com}}(g;x;r))$. That is, add to $f(x)$ a perfectly binding commitment to a preimage. The hiding property of the commitment clearly prevents the problem of leakage on x. The fact that the commitment is perfectly binding and extractable implies that $\widetilde{\mathcal{F}}$ is also extractable. Indeed, any adversary that produces a valid image, also produces a valid perfectly binding commitment to a valid preimage; hence, using the extractor for the commitment, we obtain a valid preimage. A major caveat of this approach is that the resulting $\widetilde{\mathcal{F}}$ does not support efficient image verification; indeed, the commitment is never opened, and the seed generator does not have any trapdoor on it. At this time, we are not aware of applications for non-verifiable extractable primitives other than our non-verifiable ECRH-based SNARK construction. We leave an investigation of other possible applications of non-verifiable extractable primitives for future work.

8. Candidate ECRH and PECRH Constructions

In this section, we discuss:

- a candidate construction for an ECRH, based on a Knowledge of Exponent assumption and the hardness of discrete logs; and

- a generic technique for obtaining candidate constructions for PECRHs, which we instantiate in three different ways.

As already discussed, the relaxation of ECRHs to PECRHs is crucial for (a) obtaining more candidate constructions and (b) arguing the necessity of PECRHs to the construction of SNARKs.

8.1. *ECRHs from t-Knowledge of Exponent*

Recall that ECRHs are formally discussed in Definition 6.1. The *Knowledge of Exponent assumption* (KEA) [48] states that any adversary that, given a generator g and a random group element g^{α}, manages to produce $g^{x}, g^{\alpha x}$, must "know" the exponent x. The assumption was later extended in [27,87], by requiring that given $g^{r_1}, g^{r_1\alpha}, g^{r_2}, g^{r_2\alpha}$ it is infeasible to produce f, f^{α} without "knowing" x_1, x_2 such that $f = g^{x_1 r_1} g^{x_2 r_2} = g^{x_1 r_1 + x_2 r_2}$. The t-KEA assumption is a natural extension to $t = \mathrm{poly}(k)$ pairs $g^{r_i}, g^{\alpha r_i}$.

Assumption 8.1. (t-KEA) There exists an efficiently samplable ensemble $\mathcal{G} = \{\mathcal{G}_k\}$ where each $(\mathbb{G}, g) \in \mathcal{G}_k$ consists of a group of prime order $p \in (2^{k-1}, 2^k)$ and a generator $g \in \mathbb{G}$, such that the following holds. For any polynomial-size adversary $\mathcal{A}$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}_{\mathcal{A}}$ such that for any $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$,

$$\Pr_{\substack{(\mathbb{G},g)\leftarrow\mathcal{G}_k \\ (\alpha,\mathbf{r})\overset{U}{\leftarrow}\mathbb{Z}_p\times\mathbb{Z}_p^t}}\left[\begin{array}{c}(f,f')\leftarrow\mathcal{A}(g^{\mathbf{r}},g^{\alpha\mathbf{r}},z)\\ f'=f^{\alpha}\end{array}\wedge\begin{array}{c}\mathbf{x}\leftarrow\mathcal{E}_{\mathcal{A}}(g^{\mathbf{r}},g^{\alpha\mathbf{r}},z)\\ g^{\langle\mathbf{x},\mathbf{r}\rangle}\neq f\end{array}\right]\leq \mathrm{negl}(k)\ ,$$

where $|\mathbb{G}| = p$, $\mathbf{r} = (r_1, \ldots, r_t)$, $g^{\mathbf{r}} = (g^{r_1}, \ldots, g^{r_t})$, $\mathbf{x} = (x_1, \ldots, x_t)$, and $\langle\cdot,\cdot\rangle$ denotes inner product.

A related assumption was made by Groth [81]; there, instead of random $r_1, \ldots, r_t$, the exponents are powers of the same random element, i.e., $r_i = r^i$. (As formalized in [81], the assumption does not account for auxiliary inputs, but it could naturally be strengthened to do so.)

Our assumption can be viewed as a simplified version of Groth's assumption; in particular, we could use Groth's assumption directly to get ECRHs. Furthermore, Groth's assumption is formally stated in bilinear groups, while in our setting bilinearity is not necessary. When considered in (non-bilinear) groups where t-DDH is assumed to hold, the two assumptions are actually equivalent.[16] Therefore, as Groth shows that his assumption holds in the generic group model [123] (independently of the bilinear structure) and as t-DDH is also known to hold in this model, our assumption holds in the generic group model as well.

[16] t-DDH asserts that, over suitable groups, tuples of the form $g^{x}, g^{x^2}, \ldots, g^{x^t}$ are indistinguishable from random tuples.

A Candidate ECRH *from* t-KEA. Let $\ell(k) = \mathrm{poly}(k)$ be the size of group elements for any $\mathbb{G} \in \mathcal{G}_k$. A $(\ell(k) \cdot t(k), 2\ell(k))$-compressing ECRH $\mathcal{H}$ can now be constructed in the natural way:

- To sample from $\mathcal{H}_k$: sample $(\mathbb{G}, g) \leftarrow \mathcal{G}_k$ and $(\alpha, \mathbf{r}) \overset{U}{\leftarrow} \mathbb{Z}_p \times \mathbb{Z}_p^t$, and output $h := (\mathbb{G}, g^{\mathbf{r}}, g^{\alpha\mathbf{r}})$.
- To compute $h(x_1, \ldots, x_t)$: output the pair $(g^{\langle \mathbf{r},\mathbf{x}\rangle}, g^{\langle \alpha\mathbf{r},\mathbf{x}\rangle}) = \big(\prod_{i\in[t]} g^{r_i x_i}, \prod_{i\in[t]} g^{\alpha r_i x_i}\big)$.

The extractability of $\mathcal{H}$ easily follows from the t-KEA assumption. We show that $\mathcal{H}$ is collision-resistant based on the hardness of computing discrete logarithms in $\mathcal{G}$.

Claim 8.2. *Given an oracle $\mathcal{A}$ that finds a collision within $\mathcal{H}$ with probability ε, we can compute discrete logarithms in polynomial time with probability ε/t.*

Proof Sketch. Given g^r, where $r \overset{U}{\leftarrow} \mathbb{Z}_p$, choose a random $i \in [t]$ and sample $\alpha, r_1, \ldots r_{i-1}, r_{i+1}, \ldots, r_t$. Denote $r_i = r$ and $\mathbf{r} = (r_1, \ldots, r_t)$. Feed $\mathcal{A}$ with $g^{\mathbf{r}}, g^{\alpha\mathbf{r}}$. By our initial assumption and the independent choice of i, $\mathcal{A}$ outputs $\mathbf{x}, \mathbf{x}'$ such that $x_i \neq x_i'$ and $g^{\langle \mathbf{x},\mathbf{r}\rangle} = g^{\langle \mathbf{x}',\mathbf{r}\rangle}$ w.p. at least ε/t. It follows that $r_i = (x_i - x_i')^{-1} \sum_{j\in[k]\setminus\{i\}} (x_j - x_j')r_j$. □

8.2. *PECRHs from Knowledge of Knapsack*

In Sect. 8.1, we presented a candidate ECRH based on a generalization of the Knowledge of Exponent assumption in large algebraic groups. We are now going to introduce a class of knowledge assumptions with a "lattice flavor," which we call *Knowledge of Knapsack*, to construct candidates for the weaker notion of a proximity ECRH (PECRH). Recall that PECRHs are formally discussed in Definition 6.2.

Indeed, we are not able to achieve the strict notion of ECRH from "lattice-flavor" Knowledge of Knapsack assumptions; instead, we only obtain the "noisy" notion of ECRH that we have formalized as a PECRH (which yet is still sufficient and essentially necessary, for constructing SNARKs, as discussed in Sect. 6.2). This might not be surprising, given that problems about lattices tend to involve statements about noise distributions, rather than about exact algebraic relations as in the case of t-KEA.

At high level, we define a candidate PECRH family based on knowledge assumptions of the following form: Given a set of elements $l_1, \ldots, l_t$ in some group, the only way to compute a subset sum is (essentially) to pick a subset $S \subseteq [t]$ and output the subset sum $\sum_{i\in S} l_i$. As before, this is expressed by saying that for any adversary there exists an extractor such that whenever the adversary outputs a value y which happens to be a subset sum, the extractor "explains" this y by outputting a corresponding subset.

For convenience of exposition, we first define a very general "Knowledge of Knapsack" template, where the set size t, the group, and the distribution of l_i are left as parameters, along with an amplification factor λ (saying how many such subset sum instances are to be solved simultaneously).

Hashes From Knapsacks A *knapsack* is a tuple $K = (\mathbb{H}, l_1, \ldots, l_t)$, such that $\mathbb{H}$ is (the description of) an additive finite group and $l_1 \ldots, l_t \in \mathbb{H}$.

We construct hash function ensembles out of knapsack ensembles in a natural way. Given a size parameter $t = t(k)$, amplification parameter $\lambda = \lambda(k)$, and an ensemble of knapsacks $\mathcal{K} = \{\mathcal{K}_k\}_k$, we define the hash function ensemble $\mathcal{H}^{t,\lambda,\mathcal{K}} = \left\{\mathcal{H}_k^{t,\lambda,\mathcal{K}}\right\}_k$ as follows. For $K = (\mathbb{H}, l_1, \ldots, l_t) \leftarrow \mathcal{K}_k$, let $h^{t,K} : \{0,1\}^t \to \mathbb{H}$ be given by $h^{t,K}(\mathbf{s}) := \sum_{i:s_i=1} l_i$ represented in $\{0,1\}^{\lceil \log |\mathbb{H}| \rceil}$, where the summation is over $\mathbb{H}$. Then to sample $\mathcal{H}_k^{t,\lambda,\mathcal{K}}$, draw $K^1, \ldots, K^\lambda \leftarrow \mathcal{K}_k$ and output the hash function $h(x) := (h^{t,K^1}(x), \ldots, h^{t,K^\lambda}(x))$. (That is, h is the λ-wise repetition of $h^{t,K}$.)

Knowledge of Knapsack. The *Knowledge of Knapsack* assumption with respect to $(t, \lambda, \mathcal{K}, \overset{h}{\approx}, D_h)$ asserts that the function ensemble $\mathcal{H}^{t,\lambda,\mathcal{K}}$ is proximity extractable with respect to some proximity relation $\overset{h}{\approx}$, some extended domain $D_h \subseteq \mathbb{Z}^t$, and extended function $\bar{h} : D_h \to \mathbb{H}$ defined by taking a linear combinations with coefficients in D_h (rather than just subset sums). Explicitly:

Definition 8.3. (Knowledge of Knapsack) Let $t = t(k) \in \mathbb{N}$ (size parameter) and let $\lambda = \lambda(k) \in \mathbb{N}$ (amplification parameter). Let $\mathcal{K} = \{\mathcal{K}_k\}_k$ be an efficiently samplable ensemble of knapsacks. For each h in the support of $\mathcal{K}_k$, let $\overset{h}{\approx}$ be a relation on the image of h and let D_h be an extended domain $D_h \subseteq \mathbb{Z}^t$ where $D_h \supseteq \{0,1\}$.

The *Knowledge of Knapsack* assumption with respect to $(t, \lambda, \mathcal{K}, \overset{h}{\approx}, D_h)$ states the following: For any polynomial-size adversary $\mathcal{A}$ and polynomial m, there exists a polynomial-size extractor $\mathcal{E}_\mathcal{A}$ which outputs subsets of $[t]$ such that for any $k \in \mathbb{N}$ and any auxiliary input $z \in \{0,1\}^{m(k)}$,

$$\Pr_{(\mathbb{H}^j, l_1^j, \ldots, l_t^j)_{j=1}^\lambda \leftarrow \mathcal{K}_k} \left[\begin{array}{c} (y^1, \ldots, y^\lambda) \leftarrow \mathcal{A}(K^1, \ldots, K^\lambda, z) \\ \exists \mathbf{x} \in \{0,1\}^t \ \ \forall j : y^j = \sum_i x_i l_i^j \end{array} \wedge \begin{array}{c} \mathbf{x}' \leftarrow \mathcal{E}_\mathcal{A}(K^1, \ldots, K^\lambda, z) \\ \neg\left(\mathbf{x}' \in D_h \ \wedge \ \forall j : y^j \overset{h}{\approx} \sum_{i \in [t]} x'_i l_i^j\right) \end{array} \right] \le \mathrm{negl}(k)$$

where j ranges over $\{1, \ldots, \lambda\}$, the summations are in the group $\mathbb{H}$, and the multiplications mean adding an (integer number of) elements of $\mathbb{H}$.

Compression If the groups in all the knapsacks in $\mathcal{K}$ are of size $s = s(k)$, then the function ensemble $\mathcal{H}^{t,\lambda,\mathcal{K}}$ compresses t-bit strings to $(\lambda \log s)$-bit strings.

Discussion: Sparseness and Amplification As discussed in Sect. 6.1, we wish the candidate PECRH (just like a candidate ECRH) to be superpolynomially sparse. Sparseness grows exponentially with the amplification parameter λ: If each knapsack $K \leftarrow \mathcal{K}_k$ is ρ-sparse (i.e., $|\mathsf{Image}(h^{t,K})|/|\mathbb{H}| < \rho$), then with amplification λ we obtain the candidate PECRH $\mathcal{H}^{t,\lambda,\mathcal{K}}$ that is ρ^λ-sparse. Thus, for example, as long as ρ is upper-bounded by some non-trivial constant, $\lambda > \omega(\log k)$ suffices to get superpolynomial sparseness. We will indeed use this below, in candidates where the basic knapsacks $\mathcal{K}$ must be just polynomially sparse for the proof of (proximity) collision resistance to go through.

We now proceed to propose instantiations of the Knowledge of Knapsack approach.

8.2.1. *Knowledge of Knapsack of Exponents*

We first point out that the Knowledge of Knapsack template can be used to express also the Knowledge of Exponent assumptions, by considering subset sums on pairs of the form (f, f^α). The result is similar to the t-KEA assumption (see Sect. 8.1), albeit with inferior parameters:

Assumption 8.4. (t-KKE) For $t = t(k) \in \mathbb{N}$, the t-KKE (Knowledge of Knapsack of Exponents) states that there exists an efficiently samplable ensemble $\mathcal{G} = \{\mathcal{G}_k\}$ where each $(\mathbb{G}, g) \in \mathcal{G}_k$ consists of a multiplicative group of prime order p in $(2^{k-1}, 2^k)$ and a generator $g \in \mathbb{G}$, such that the Knowledge of Knapsack assumption with respect to $(t, 1, \mathcal{K}^{\mathsf{E}}, \equiv_{\mathbb{H}}, \{0,1\}^t)$ holds for the ensemble $\mathcal{K}^{\mathsf{E}} = \{\mathcal{K}_k^{\mathsf{E}}\}_k$ defined as follows (where $\equiv_{\mathbb{H}}$ is equivalence in the group $\mathbb{H}$ given below):

To sample from $\mathcal{K}_k^{\mathsf{E}}$, draw $(\mathbb{G}, g) \leftarrow \mathcal{G}_k$, let $\mathbb{H} = \mathbb{G} \times \mathbb{G}$ considered as an additive group, draw $\alpha \leftarrow \mathbb{Z}_p$ and $\mathbf{r} \leftarrow \mathbb{Z}_p^t$, let $l_i = (g^{r_i}, g^{\alpha r_i}) \in \mathbb{H}$, and output $(\mathbb{H}, l_1, \ldots, l_t)$.

The hash function ensemble $\mathcal{H}^{t,1,\mathcal{K}^{\mathsf{E}}}$ is readily verified to be $(t(k), 2k)$-compressing, and collision-resistant assuming the hardness of taking discrete logs. Note that its range is indeed sparse, as prescribed in Sect. 6.1: for $h \leftarrow \mathcal{H}^{t,1,\mathcal{K}^{\mathsf{E}}}$, $|\mathsf{Image}(h)|/|\mathbb{H}| = |\mathbb{G}|/|\mathbb{G} \times \mathbb{G}| \approx 1/2^k$. Alas, we lost a factor of k in the compression compared to directly using t-KEA, since we hash t bits as opposed to t group elements as in t-KEA.

8.2.2. *Knowledge of Knapsack of Noisy Multiples*

Next, we propose a new knowledge assumption based on the following goal: Given noisy integer multiples $L = (l_i, \ldots, l_t)$ in $\mathbb{Z}_N$ of a secret real number α (of magnitude about $\sqrt{N}$), find a subset sum of these multiples.[17] The knowledge assumption says (roughly) that whenever an efficient adversary produces such a subset sum, it knows the corresponding subset. This, however, requires care, since, taken literally, the assumption is clearly false. To motivate our definition, we describe several attempted attacks, and how the definition avoids them.

- *Perturbation Attack* Any small integer is close to a multiple of α (i.e., 0), and is thus likely to be a sum of *some* subset of L (when L is long enough, as it is in our setting). Thus, the adversary $\mathcal{A}$ could simply output a random small integer and thereby challenge the extractor $\mathcal{E}$ to find a corresponding subset. We let the extractor avoid this difficult task by using the notion of PECRHs defined above, with the proximity relation $\overset{h}{\approx}$ chosen so that the extractor only needs to output a subset that sums to *approximately* the adversary's output (in the above example, the extractor can output the empty set).
- *Integer Coefficients Attack* An adversary $\mathcal{A}$ could pick an integer combination of L with coefficients that are small but not all 0 and 1. Even though this is not a valid

[17] Our construction is inspired by a cryptosystem of Regev [117,118], where the public key is sampled from a similar distribution, and indeed our analysis of collision resistance and sparsity invokes Regev's. This is elaborated below.

computation of a sum over a subset of L, the result y is still close to a multiple of the secret real number and thus, as above, is likely to be a subset sum of for *some* subset, so the extractor $\mathcal{E}$ must "explain" y. We aid $\mathcal{E}$ by enlarging the extended domain D_h to allow small integer coefficients, so that the (non-blackbox) extractor may output the coefficients used by the adversary.

- *Fractional Coefficients Attack* An adversary $\mathcal{A}$ could pick a fractional combination of elements of L. For example, $l_1/2$ will be close to a multiple of α whenever l_1 happens to be close to an even multiple of α (i.e., with probability half). However, we amplify our knapsack to consider λ instances concurrently (each consisting of noisy multiples L of some different α), so the extractor is challenged only in the exponentially unlikely event that all λ instances have l_1 that is close to an even multiple.

Comparison to t-KKE. The above complications arise due to the addition of noise to l_i in the generation of the knapsack instances (otherwise α would be found computing the greatest common divisor on L, easily leading to collisions). Thus the collection of resulting subset sums constitutes a train of "hills" (each clustered around a multiple of α), which an adversary can traverse by the aforementioned attacks. Conversely, in t-KKE from Sect. 8.2.1, the underlying discrete log problem does not require injection of noise, hence the subset sums constitute a set of distinct "well-spaced" points in $\mathbb{G} \times \mathbb{G}$, and so (one may hope) the adversary can navigate the structure of the image only by algebraic operations that the extractor can unravel.

Definition 8.5. Let $N \in \mathbb{Z}$, $\alpha \in \mathbb{R}$ and $\bar{\sigma} \in (0, 1)$. We define the distribution $\mathsf{NM}_{\alpha,\bar{\sigma},N}$ of noisy multiples of α in the range $[0, \ldots, N-1)$, with relative noise of standard deviation $\bar{\sigma}$, as follows. Draw an integer $x \overset{U}{\leftarrow} \{0, \ldots, \lfloor N/\alpha \rfloor\}$ and a noise fraction $y \leftarrow \mathcal{N}_{0,\bar{\sigma}^2}$ (the normal distribution with mean 0 and variance $\bar{\sigma}^2$). Output $\lfloor \alpha(x+y) \bmod N \rfloor$.

Assumption 8.6. ((t, σ)-KKNM) For $t = t(k) > k \in \mathbb{N}$ and noise parameter $\sigma = \sigma(k) \in (0, 1)$, the (t, σ)-KKNM (Knowledge of Knapsack of Noisy Multiples) states that the Knowledge of Knapsack assumption with respect to $(t, \mathcal{K}^{\mathsf{NM},t,\sigma}, \overset{h}{\approx}, D_h)$ holds for the following distribution of knapsack elements.

To sample from $\mathcal{K}_k^{\mathsf{NM},t,\sigma}$ do the following: Let $N = 2^{8k^2}$, draw $h \overset{U}{\leftarrow} \left\{h \in [\sqrt{N}, 2\sqrt{N}) : |h - \lfloor h \rceil| < \frac{1}{16t}\right\}$ and draw $\bar{\sigma}$ such that $\bar{\sigma}^2 \overset{U}{\leftarrow} [\sigma^2, 2\sigma^2)$. Let $\alpha = N/h$. Draw t values $l_1, \ldots, l_t \leftarrow \mathsf{NM}_{\alpha,\bar{\sigma},N}$. Output $(\mathbb{Z}_N, l_1, \ldots, l_t)$.

For $h \leftarrow \mathcal{K}_k^{\mathsf{NM},t,\sigma}$, let $D_h = \left\{\mathbf{x} \in \mathbb{Z}^t : ||\mathbf{x}||_2 < t \log^2 t\right\}$, and let $\overset{h}{\approx}$ be s.t. for $y, y' \in \mathbb{Z}_N$, $y \overset{h}{\approx} y'$ if their distance in $\mathbb{Z}_N$ is at most $\sqrt{N}/9$.

Relation to Regev's cryptosystem [117,118]. The above distributions are essentially the same as in Regev's cryptosystem, with minor changes for clarity in the present context. Explicitly, the mapping is as follows. The distribution $Q_\beta = (\mathcal{N}_{0,\beta/2\pi} \bmod 1)$ from [118, Section 2.1] is replaced by $\mathcal{N}_{0,\bar{\sigma}^2}$, for $\beta = 2\pi\bar{\sigma}^2$ (the statistical difference between the two is negligible because $\bar{\sigma}$ will be polynomially small). The distribution $\mathsf{NM}_{\alpha,\bar{\sigma},N}$ is a scaling up by N of $T_{h,\beta}$ as defined in [118, above Definition 4.3], for

$h = N/d$ (except for the above deviation, and a deviation due to the event $x + y > h$ which is also negligible in our setting). Thus, the distribution $(l_1, \ldots, l_t)$ sampled by $\mathcal{K}_{\mathsf{NM}}$ is negligibly close to that of public keys in [118, Section 5] on parameters $n = k$, $m = t$, $\gamma(n) = \sqrt{2/\pi}\,/\,\sigma(k)$.

Collision Resistance We show that the hash function ensemble $\mathcal{H}^{\mathsf{KKNM}} = \mathcal{H}^{t,\lambda,\mathcal{K}^{\mathsf{NM},t,\sigma}}$ is proximity-collision-resistant for any $t = O(k^2)$ and suitable λ and σ, assuming on the hardness of the unique shortest vector problem (uSVP) in lattices. Recall that $f(\mu)$-uSVPis the computational problem of finding a shortest vector in a lattice of dimension μ given that the shortest vector is at least $f(\mu)$ times shorter than any other (non-parallel) lattice vector (see [100,118]).

Claim 8.7. *The samples $l_1, \ldots, l_t$ drawn by $\mathcal{K}^{\mathsf{NM},t,\sigma}$ are pseudo-random (i.e., indistinguishable from t random integers in the interval $\{0, \ldots, N-1\}$), assuming hardness of $(\sqrt{2/\pi\,\mu}/\sigma(\mu))$-uSVP.*

Proof Sketch. It suffices to show pseudo-randomness for the distribution obtained by modifying $\mathcal{K}^{\mathsf{NM},t,\sigma}$ to sample $h \overset{U}{\leftarrow} [\sqrt{N}, 2\sqrt{N})$ (for the same reason as in [118, Lemma 5.4]). This pseudo-randomness follows from [118, Theorem 4.5] with $g(n) = \sqrt{2\mu/\pi}/\sigma(\mu)$.

Claim 8.8. *The function ensemble $\mathcal{H}^{\mathsf{KKNM}}$ is proximity-collision-resistant, with $\overset{h}{\approx}$, $D_h, \bar{h}$ defined as in Assumption 8.6, for $t = O(k^2)$, assuming hardness of $\tilde{O}\left(\max\left(\mu^{3/2},\right.\right.$ $\left.\left.\sqrt{\mu}/\sigma(\mu)\right)\right)$-uSVP.*

Proof Sketch. In the following, adapting the notation of [118], for $x \in \mathbb{R}$, we denote by $\mathsf{frc}\,(x) = |x - \lfloor x \rceil|$ the fractional part of x. By Claim 8.7, the hash functions drawn by $\mathcal{H}^{\mathsf{KKNM}}$ are indistinguishable from the ensemble $\mathcal{U}$ of uniformly random modular subset sums (as defined in [118, Section 6]), assuming $\tilde{O}(\sqrt{\mu}/\sigma(\mu))$-uSVP. It thus suffices to show that $\mathcal{U}$ is proximity-collision-resistant, since this implies finding collisions in $\mathcal{H}^{\mathsf{KKNM}}$ would distinguish it from $\mathcal{U}$. The ensemble $\mathcal{U}$ is collision-resistant assuming $\tilde{O}(\mu^{3/2})$-uSVP, by [118, Theorem 6.5]. Moreover, the proximity relation $\overset{h}{\approx}$ is accommodated by noting that the theorem still holds if in its statement, $\sum_{i=1}^{m} b_i a_i \equiv 0 \pmod{N}$ is generalized to $\mathsf{frc}\left((\sum_{i=1}^{m} b_i a_i)/N\right) < 1/9\sqrt{N}$; inside that theorem's proof, this implies $\mathsf{frc}\left((\sum_{i=1}^{m} b_i z_i)/N\right) < 1/8\sqrt{N}$ and thus, in the penultimate displayed equation, $h \cdot \mathsf{frc}\left((\sum_{i=1}^{m} b_i z_i)/N\right) < h/9\sqrt{N} < 1/9$ so the last displayed equation still holds and the proof follows. The extended domain D_h and induced $\bar{h}$ are accommodated by noting that Regev's bound $||b|| \leq \sqrt{m}$ (in his notation) generalizes to $||b|| \leq \tilde{O}(\sqrt{m})$.

Sparseness and Parameter Choice To make the extractability assumption plausible, we want the function's image to be superpolynomially sparse within its range, as discussed in Sect. 6.1. Consider first the distribution $\mathcal{H}^{\mathsf{KKNM}} = \mathcal{H}^{t,1,\mathcal{K}^{\mathsf{NM},t,\sigma}}$ (i.e., $\lambda = 1$, meaning no amplification). The image of h drawn from $\mathcal{H}^{\mathsf{KKNM}}$ becomes "wavy" (hence sparse) when the noise (of magnitude $\sigma\alpha$) added to each multiple of α is sufficiently small, resulting in distinct peaks, so that any subset sum of t noisy multiples is still a noisy multiple:

Claim 8.9. *For $\sigma(k) = 1/16t\log^2 k$, the ensemble $\mathcal{K}^{\mathsf{NM},t,\sigma}$ is $\frac{1}{2}$-sparse:*

$$\Pr_{h \leftarrow \mathcal{H}_k^{t,1,\mathcal{K}^{\mathsf{NM},t,\sigma}}} \big[|\mathsf{Image}(h)|/N > 1/2\big] < \operatorname{negl}(k)$$

Proof sketch. In terms of the corresponding Regev public key, this means decryption failure becomes impossible with all except negligible probability over the keys. For this, it clearly suffices that each of the t noisy multiples is at most $\alpha/16t$ away from a multiple of α, so that any sum of them will have accumulated noise at most $\alpha/16$ (plus another $\alpha/16$ term due to modular reductions, as in Regev's decryption lemma [118, Lemma 5.2]). This indeed holds for $\sigma(k) = 1/16t\log^2 k$, by a tail bound on the noise terms $\alpha\mathcal{N}_{0,\sigma}$ followed by a union bound over the t samples.

Thus, the image becomes somewhat sparse when $\sigma = \tilde{o}(1/t)$. However, superpolynomial sparseness would require making σ superpolynomially small (and likewise a tighter distribution over h), in which case Claim 8.8 would require assuming hardness of $\mu^{\omega(1)}$-uSVP; this assumption is unmerited in light of the excellent heuristic performance of LLL-type lattice reduction algorithms on lattices with large gaps (e.g., [75] conjecture, from experimental evidence, that 1.02^{μ}-uSVPis easy). Instead, we can set $\sigma = \tilde{\Omega}(1/k^2)$ so that Claim 8.8 needs to assume merely hardness of $\tilde{O}(\mu^{3/2})$-uSVP, and then amplify via repetition, by choosing sufficiently large λ. In particular, by setting $\sigma(k) = \tilde{o}(1/t)$, $\lambda = \omega(\log(k))$ and $t = O(k^2)$, we indeed obtain superpolynomial sparseness.

Regarding the aforementioned integer-coefficient attack, note that the extended domain D_h allows $\mathcal{E}$ to explain y via any vector using a linear combination whose coefficients have ℓ_2 norm at most $t\log^2 t$, since beyond this norm, the linear combination is unlikely to be in the image of h.

Lastly, note that $k = n^2$ (or, indeed, any $k = n^{1+\varepsilon}$) suffices for the SNARK construction.

Relation to Other Lattice Hardness Assumptions The collision resistance is shown assuming hardness of the uSVP lattice problem. This can be generically translated to other (more common) lattice hardness assumptions, such as GAPSVP and BDD (bounded distance decoding), following Lyubashevsky and Micciancio [100].

8.2.3. *Knowledge of Knapsack of Noisy Inner Products*

Further PECRH candidates can be obtained from Knowledge of Knapsack problems on other lattice-based problems. In particular, the Learning with Errors problem [119] problem leads to a natural knapsack ensemble, sampled by drawing a random vector $\mathbf{s} \in \mathbb{Z}_p^n$ and then outputting a knapsack $K = (\mathbb{Z}_p^{n+1}, l_1, ..., l_t)$ where each l_i consists of a random vector $\mathbf{x} \overset{U}{\leftarrow} \mathbb{Z}_p^n$ along with the inner product $\mathbf{s} \cdot \mathbf{x} + \varepsilon$ where ε is independently drawn noise of small magnitude in $\mathbb{Z}_p$. For suitable parameters this ensemble is sparse, and proximity-collision-resistant following an approach similar to KKNM above: First show pseudo-randomness assuming hardness of LWE [119] and then rely on the collision resistance of the uniform case (e.g., [5,64,106]).

In this case, amplification can be done more directly, by reusing the same $\mathbf{x}$ with multiple s_i instead of using the generic amplification of Definition 8.3.

9. Zero-Knowledge SNARKs

In this section, we consider the problem of constructing *zero-knowledge* SNARK*s* (zkSNARKs); that is, we want to ensure that the succinct proof does not leak information about the witness used to generate it.

Recall that SNARKs do not require any setup assumptions (i.e., are in the plain model). However, now that we seek the additional property of zero knowledge, we cannot proceed in the plain model because otherwise we would obtain a two-message zero-knowledge protocol in the plain model, which is impossible [76]. We thus work in the standard common reference string (CRS) model.

There are two natural candidate zkSNARK constructions that one could consider, both starting with a NIZK system in the CRS model (see [78]) and making it succinct:

- "SNARK *on top of* NIZK." At high level, the prover first produces a (non-succinct) NIZK argument π_{ZK} for the statement y (given a valid witness w) and then produces a non-interactive succinct argument π for the statement y' that the ZK verifier would have accepted the proof π_{ZK} for y.
- "NIZK *on top of* SNARK." At high level, the prover first produces a non-interactive succinct argument π for the statement y (given a valid witness w) and then produces a NIZK argument π_{ZK} for the statement y'' that the verifier would have accepted the (succinct) proof π for y.

In both constructions, one needs the NIZK system to be adaptively sound. We now describe in further detail each of the above approaches.

9.1. *Zero-Knowledge SNARKs*

We define zero-knowledge SNARGs in the CRS model.

Definition 9.1. A triple of algorithms $(\mathsf{Setup}, \mathcal{P}, \mathcal{G}_{\mathcal{V}}, \mathcal{V})$ is a zero-knowledge SNARG for the relation $\mathcal{R} \subseteq \mathcal{R}_{\mathcal{U}}$ if the following conditions are satisfied:

1. *Completeness* For any $(y, w) \in \mathcal{R}$,

$$\Pr\left[\mathcal{V}(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{priv}, y, \Pi) = 1 : \begin{array}{c} (\mathsf{crs}_{\mathsf{ZK}}, \mathsf{trap}) \leftarrow \mathsf{Setup}(1^k) \\ (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ \Pi \leftarrow \mathcal{P}(y, w, \mathsf{vgrs}, \mathsf{crs}_{\mathsf{ZK}}) \end{array}\right] = 1\ .$$

   In addition, $\mathcal{P}(y, w, \mathsf{vgrs}, \mathsf{crs}_{\mathsf{ZK}})$ runs in time $\mathrm{poly}(k, |y|, t)$.
2. *Succinctness* The length of the proof Π that $\mathcal{P}(y, w, \mathsf{vgrs}, \mathsf{crs}_{\mathsf{ZK}})$ outputs, as well as the running time of $\mathcal{V}(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{priv}, y, \Pi)$, is bounded by

$$p(k + |y|) = p(k + |M| + |x| + \log t)\ ,$$

where p is a universal polynomial that does not depend on $\mathcal{R}$. In addition, $\mathcal{G}_{\mathcal{V}}(1^k)$ runs in time $p(k)$; in particular, $(\mathsf{vgrs}, \mathsf{priv})$ are of length $p(k)$.

3. *Adaptive soundness* For all polynomial-size provers $\mathcal{P}^*$ and any $k \in \mathbb{N}$,

$$\Pr\left[\begin{array}{c} \mathcal{V}(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{priv}, y, \Pi) = 1 \\ y \notin \mathcal{L}_{\mathcal{R}} \end{array}\ c : \begin{array}{c} (\mathsf{crs}_{\mathsf{ZK}}, \mathsf{trap}) \leftarrow \mathsf{Setup}(1^k) \\ (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{G}_{\mathcal{V}}(1^k) \\ (y, \Pi) \leftarrow \mathcal{P}^*(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{vgrs}) \end{array}\right] \leq \mathrm{negl}(k)\ .$$

4. *Zero knowledge* There is an (expected) polynomial-time simulator $\mathcal{S}$ such that for all polynomial-size verifiers $\mathcal{V}^*$ and any $k \in \mathbb{N}$, for every $(y, w) \in \mathcal{R}$, the following probabilities are negligibly close

$$\Pr\left[\mathcal{V}^*(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{priv}, y, \Pi) = 1 : \begin{array}{c} (\mathsf{crs}_{\mathsf{ZK}}, \mathsf{trap}) \leftarrow \mathsf{Setup}(1^k) \\ (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{V}^*(1^k) \\ \Pi \leftarrow \mathcal{P}(y, w, \mathsf{vgrs}, \mathsf{crs}_{\mathsf{ZK}}) \end{array}\right]$$

and

$$\Pr\left[\mathcal{V}^*(\mathsf{crs}_{\mathsf{ZK}}, \mathsf{priv}, y, \Pi) = 1 : \begin{array}{c} (\mathsf{crs}_{\mathsf{ZK}}, \mathsf{trap}) \leftarrow \mathsf{Setup}(1^k) \\ (\mathsf{vgrs}, \mathsf{priv}) \leftarrow \mathcal{V}^*(1^k) \\ \Pi \leftarrow \mathcal{S}(y, \mathsf{vgrs}, \mathsf{crs}_{\mathsf{ZK}}, \mathsf{trap}) \end{array}\right]\ .$$

A *zero-knowledge* zkSNARG *of knowledge*, or zkSNARK for short, is a zkSNARG where soundness is strengthened to proof of knowledge, similarly to a SNARK.

9.2. *SNARK on top of NIZK*

Theorem 9.2. *If there exist adaptively sound NIZK arguments and SNARKs, then there exist zkSNARGs. If furthermore the NIZK argument is a proof of knowledge, then we obtain zkSNARKs.*

Proof Sketch. The setup phase consists of generating a common reference string $\mathsf{crs}_{\mathsf{ZK}}$ and publishing it. A verifier then generates $(\mathsf{vgrs}, \mathsf{priv})$ and sends vgrs to the prover, and keeps the private verification state priv for later use. In order to prove membership for an instance y with valid witness w, the prover performs the following steps:

1. Generate, using $\mathsf{crs}_{\mathsf{ZK}}$, a (non-succinct) NIZK argument of knowledge π_{ZK} for the instance y using the valid witness w.
2. Generate, using vgrs, a (succinct) SNARK proof π for the NP statement "there exists a proof π_{ZK} that makes the NIZK verifier accept it as a valid proof for the instance y, relative to $\mathsf{crs}_{\mathsf{ZK}}$."
3. Send (y, π) to the verifier.

The verifier can now use $(\mathsf{vgrs}, \mathsf{priv})$ and $\mathsf{crs}_{\mathsf{ZK}}$ to verify (y, π) by running the SNARK verifier on the above NP statement.

By using the SNARK extractor, we can obtain (efficiently) a valid NIZK proof π_{ZK} for the claimed theorem y. Invoking the (computational) soundness of the NIZK argument, it must be that y is a true theorem (with all except negligible probability). If the NIZK argument also guarantees an extractor, we could use it to also extract a witness for y.

As for the zero-knowledge property, it follows from the zero knowledge property of the NIZK argument: the proof π_{ZK} is "already" zero knowledge, and thus using it as a witness in the SNARK implies that the resulting succinct proof will also be zero knowledge. More formally, we can first run the simulator of the NIZK system to obtain a simulated proof π'_{ZK} for y, and then honestly generate the proof π using π'_{ZK} as the witness to the NP statement. (We note that as long as the simulator for the NIZK is black-box, so will be the simulator of the zkSNARK.)

We note that:

- If the NIZK argument extractor requires a trapdoor for the common reference string $\mathsf{crs}_{\mathsf{ZK}}$, so will the extractor for the resulting zkSNARK.
- The common reference string $\mathsf{crs}_{\mathsf{ZK}}$ of the NIZK argument needs to be of size polynomial in the security parameter (and must not depend on the theorem being proved).
- Even if zkSNARKs "live" in the common reference string model, proofs are still only privately verifiable (if indeed the SNARK that we start with requires a designated verifier): The verifier generates $(\mathsf{vgrs}, \mathsf{priv})$ and sends vgrs to the prover; the prover uses both the vgrs and the common reference string $\mathsf{crs}_{\mathsf{ZK}}$ to produce a proof π for a theorem of his choice; the verifier then uses both $(\mathsf{vgrs}, \mathsf{priv})$ and $\mathsf{crs}_{\mathsf{ZK}}$ to verify the proof. In other words, the $\mathsf{crs}_{\mathsf{ZK}}$ can be used by multiple verifiers, each one of which will generate a $(\mathsf{vgrs}, \mathsf{priv})$ pair "on the fly" whenever they want to contact a prover. (Moreover, if the vgrs of the underlying SNARK can be reused, so can the vgrs of the resulting zkSNARK.)

For example, to obtain zkSNARKs, one may combine any SNARKs with the NIZK arguments of knowledge of Abe and Fehr [3] (which are based on an extended Knowledge of Exponent assumption, and happen to have an extractor that does not require a trapdoor).

Proof of Knowledge Strikes Again We emphasize that, in the "SNARK on top of NIZK" approach, the proof of knowledge of the SNARK was crucial for obtaining Theorem 9.2, even when only aiming for (only-sound) zkSNARGs. (Other instances where proof of knowledge played a crucial role were the results of Sect. 7, and thus in particular the "converse" of our main technical theorem, as well as some applications discussed in Sect. 10.)

9.3. *NIZK on top of SNARK*

Theorem 9.3. *If there exist adaptively sound NIZK arguments of knowledge, SNARGs, and function-hiding FHE, then there exist zkSNARGs. If furthermore there exist SNARKs, then we obtain zkSNARKs.*

Proof sketch. The setup phase again consists of generating a common reference string $\mathsf{crs}_{\mathsf{ZK}}$ and publishing it. A verifier then generates $(\mathsf{sk}, \mathsf{pk})$ for the FHE and $(\mathsf{vgrs}, \mathsf{priv})$ for a SNARG; then, it sends vgrs and $e := \mathsf{Enc}_{\mathsf{pk}}(\mathsf{priv})$ to the prover and keeps e and sk for later use. In order to prove membership for an instance y with valid witness w, the prover performs the following steps:

1. Generate, using vgrs, a (succinct) SNARG proof π for y,
2. Sample randomness R for the (function hiding) homomorphic evaluation and compute $\hat{e} = \mathsf{Eval}_R(e, C_{y,\pi})$, where $C_{y,\pi}$ is a circuit that given input priv computes $\mathcal{V}(\mathsf{priv}, y, \pi)$, where $\mathcal{V}$ is the SNARG verifier.
3. Generate using $\mathsf{crs}_{\mathsf{ZK}}$, a NIZK argument of knowledge π_{ZK} for the NP statement "there exist R, π such that $\hat{e} = \mathsf{Eval}_R(e, C_{y,\pi})$." Note that the size of the corresponding NP computation depends only on $|y|$ (and the security parameter).
4. Send $(y, \pi_{\mathsf{ZK}}, \hat{e})$ to the verifier.

The verifier can now use $y, e, \hat{e}$ and $\mathsf{crs}_{\mathsf{ZK}}$ to verify the proof, by running the NIZK verifier and then verifying that $\hat{e}$ decrypts to 1.

By using the NIZK extractor, we can obtain (efficiently) a valid SNARG proof π for the claimed theorem y. Invoking the semantic security of the FHE and the (computational) soundness of the SNARG, it must be that y is a true theorem (with all except negligible probability). If the SNARG also guarantees an extractor (i.e., it is a SNARK), we could use it to also extract a witness for y.

The proof $(\pi_{\mathsf{ZK}}, \hat{e})$ is zero knowledge, because we can simulate $\hat{e}$ (from the evaluation result "1") by the function hiding of the FHE and then simulate π_{ZK} by running the NIZK simulator. (We note that as long as the simulator for the NIZK is black-box, so will be the simulator of the zkSNARK.)

Unlike in the "SNARK on top of NIZK" approach, in the "NIZK on top of SNARG" approach the knowledge property of the SNARG was not needed, but we had to additionally assume the existence of function-hiding FHE. Had the SNARG been publicly verifiable this assumption would not have been needed.

10. Applications of SNARKs and zkSNARKs

In this section, we discuss applications of SNARKs and zkSNARKs to delegation of computation (Sect. 10.1) and secure computation (Sect. 10.2).

10.1. *Delegation of Computation*

Recall that, in a *two-message delegation scheme* (in the plain model): To delegate a T-time function F on input x, the delegator sends a message σ to the worker; the worker computes an answer (z, π) to send back to the delegator; the delegator outputs z if π is a convincing proof of the statement "$z = F(x)$." The delegator and worker time complexity are respectively bounded by $p(|F| + |x| + |F(x)| + \log T)$ and $p(|F| + |x| + |F(x)| + T)$, where p is a universal polynomial (not depending on the specific function being delegated).

(Throughout this section, we ignore the requirement for input privacy because it can always be achieved by using a semantically secure fully homomorphic encryption scheme.)

10.1.1. *Folklore Delegation from Succinct Arguments*

There is a natural method to obtain a two-message delegation scheme from a designated-verifier non-interactive succinct argument for NP with *adaptive soundness*: The delegator sends the desired input x and function F to the worker (along with the verifier-generated reference string vgrs, which is independent of the statement being proved), and asks him to prove that he evaluated the claimed output z for the computation $F(x)$ correctly.

In the above paragraph, "for NP" indicates that it is enough for there to exist a protocol specialized for each NP relation (as the delegator, at delegation time, does know which NP relation is relevant for the function being delegated), but the succinctness requirement is still required to be universal, as captured by our Definition 9.1.

We note that, as long as one uses FHE in order to obtain input privacy, designated-verifier SNARGs, as opposed to publicly verifiable SNARGs, suffice, since public verification is lost anyhow upon using FHE. In fact, there is also no need to insist that the verifier-generated reference string is reusable (beyond the logarithmically many theorems that it can always support anyway), as a fresh string can be very quickly generated and "sent out" along with the function and input being delegated. Thus, starting with designated-verifier non-interactive succinct arguments, is usually "enough" for delegation of computation.

Furthermore, the use of succinct arguments, in a sense, provides the "best" properties that one could hope for in a delegation scheme:

- There is *no need for preprocessing* and *no need to assume that the verifier's answers remain secret*. All existing work providing two-message *generic* delegation schemes are in the preprocessing setting and assume that the verifier's answers remain secret [42,65,69,93,94,105]. (Notable exceptions are the works of Benabbas et al. [23] and Papamanthou et al. [115], which, however, only deal with delegation of specific functionalities, such as polynomial functions or set operations.)
- The delegation scheme can also support inputs of the worker: One can delegate functions $F(x, x')$ where x is supplied in the first message by the delegator, and x' is supplied by the worker. (Indeed, x' acts as a "witness.") This extension is *delegation with worker input*.

We note that when delegating deterministic computations, the delegator can use a SNARG with non-adaptive soundness by requesting a proof for each bit of the claimed output. Indeed, the statement that the ith output bit of a computation is "0" (respectively, "1") is a fixed statement, independent of the reference string. However, to support worker input, adaptive soundness seems to be needed. Indeed, the previous solution does not work for non-deterministic computations as there is no guarantee that the prover uses the same witness for each one of the output bits.

10.1.2. *Our Instantiation*

Our main technical result, Theorem 1, provides an instantiation, based on a simple and generic knowledge assumption (instantiated by several quite different candidates), of the designated-verifier non-interactive succinct argument for NP with adaptive soundness required for constructing a two-message delegation scheme.

Corollary 10.1. *Assume that there exists (weak proximity) extractable collision-resistant hash functions. Then there exists a two-message delegation scheme.*

We note that previous two-message arguments for NP [49,105] did not provide strong enough notions of succinctness or soundness to suffice for constructing delegation schemes.

Our specific instantiation also has additional "bonuses":

- Not only is the delegation sound, *but also has a proof of knowledge*. Therefore, $F(x, x')$ could involve cryptographic computations, which would still be meaningful because the delegator would know that a "good" input x' can be found in efficient time.
  For example, the delegated function $F(x, x')$ could first verify whether the hash of a long x' is x and, if so, proceed to conduct an expensive computation; if the delegation were merely sound, the delegator would not be able to delegate such a computation, for such an x' may always exist!
  We discuss more consequences of this point in the paragraph below.
- Even if the construction from our Theorem 1 formally requires the argument to depend on a constant $c \in \mathbb{N}$ bounding the time to verify the theorem, the only real dependence is a simple verification by the verifier (i.e., checking that $t \leq |x|^c$), and thus in the setting of delegation of computation, we obtain a *single* protocol because the delegator gets to choose c. Of course, despite the dependence on c, our construction still delivers the "universal succinctness" (as already remarked in Sect. 5.2, satisfying our Definition 9.1) required at the beginning of this section.
  (Indeed, note that if the polynomial bounding the verifier time complexity is allowed to depend on the function being delegated, then a trivial solution is to just let the verifier compute the function himself, as the function is assumed to be poly-time computable!)
- When our construction is instantiated with the quasi-linear-time PCPs of Ben-Sasson et al. [29], we get essentially optimal efficiency (up to polylogarithmic factors):
  - the delegator's first message requires time complexity $\mathrm{poly}(k, \log T)\tilde{O}(|F|+|x|)$;
  - the worker's computation requires time complexity $\mathrm{poly}(k)\tilde{O}(|F|+|x|+|F(x)|+T)$; and
  - the delegator's verification time requires time complexity $\mathrm{poly}(k, \log T)\tilde{O}(|F|+|x|+|F(x)|)$.

Delegating Memory, Streams, and Authenticated Data We would like to point out that the SNARK's adaptive proof of knowledge enables the delegator to handle "large inputs."

Indeed, if we are interested in evaluating many functions on a large input x, we could first in an offline stage compute a Merkle hash c_x for x and then communicate x to the worker; later, in order to be convinced of $z = F(x)$, we simply ask the worker to prove to us that there is some $\tilde{x}$ such that the Merkle hash of $\tilde{x}$ is c_x and, moreover, $z = F(\tilde{x})$. By the collision resistance of the Merkle hash, the delegator is convinced that indeed $z = F(x)$—the proof of knowledge is crucial to invoke collision resistance!

Note that x, which now "lies in the worker's untrusted memory," can be updated by letting the worker compute the updated x and prove that the new Merkle hash is a "good" one. In this way, we are able to give simple *two-message* constructions for the task of *memory delegation* [40]—again getting the "best" that we can hope for here. The resulting schemes, based on SNARKs, are simpler than previously proposed (but of course, to instantiate the SNARKs, one may have to invoke stronger assumptions).

Another task is that of *streaming delegation* [40,46,114], where new pieces of x stream by the verifier, and the verifier needs to update the hash c_x on the fly, *without* help from the prover. This can be done if the verifier maintains, in addition to c_x, a logarithmic number of other hashes, representing the roots of other subtrees; in essence, this relies on applying known Merkle tree traversal techniques (see, e.g., [102, Ch. 5] or [60, Ch. 3]).

Of course, special cases of delegating "large datasets" such as [23] and [115] are also implied by our instantiation. (Though, in the case of [115], we only get a variant where proofs are for a designated verifier rather than being publicly verifiable.) While our construction is definitely not as practically efficient, it provides the only other construction with two messages (i.e., is "non-interactive").

As yet another example of an application, consider the following scenario. A third party publishes on a public database a large amount of authenticated data (e.g., statistics of public interest) along with his own public key, denoted by x' and pk respectively. A worker comes along and wants to compute a function F over this data, but, because the data is so large, is only interested in learning the result z of the computation but not seeing the data itself. Relying on the proof of knowledge property (and making the inconsequential simplifying assumption that the third party only ever published a single authenticated database), the worker could ask the database to prove that there is some (x'', σ) such that $z = F(x'')$ *and* each entry of x'' is accompanied by a corresponding signature in σ relative to pk. In other settings, one may prefer to think of the authenticated database as private, and thus a zero-knowledge property would be needed; this can be guaranteed by either Theorems 9.2 or 9.3 in the CRS model.

In all of the above examples, the delegator is only "paying" polylogarithmically in the size of the data upon verification time.

10.2. *Succinct Non-Interactive Secure Computation*

Non-interactive secure computation (NISC) [88] allows a receiver R to publish a string containing an encryption of his secret input x, so that a sender S, holding a possibly very long input y, can reveal $f(x, y)$ to R by sending him a single message. This should be done while simultaneously protecting the secrecy of y against a malicious R and preventing S from any malicious influence on the output of R (in particular, computing the function f incorrectly). In *succinct NISC* (SNISC), we also require that the amount of work performed by R (and thus the communication complexity) is polynomial in the security parameter k and the input/output length of f. In particular, it is essentially independent of the complexity of f (more precisely, polylogarithmic in the running time of f).

When the parties are semi-honest, there are known solutions for the problem (e.g., based on fully homomorphic encryption with function privacy [63]). Naor and Nissim [109] observe that using the succinct zero-knowledge arguments of Kilian [89] one can enhance the GMW semi-honest-to-malicious compiler [74] to be communication preserving. However, the resulting protocol is not round preserving and hence cannot be used to achieve SNISC.

We observe that, relying on zkSNARKs, we can obtain SNISC against malicious parties in the CRS model. (Though the string published by the receiver R can only be used logarithmically many times; if the receiver wishes to receive more messages, he should publish a new string.) Here we only sketch the protocol and analysis behind this observation. For the full definitions of SNISC, we refer the reader to [88].

The protocol To jointly compute a function f:

1. The receiver R sends the verifier-generated reference string vgrs, an encryption c of its input x under a fully homomorphic encryption, and π_{ZK}^{R}, a NIZK proof of knowledge of input x (and randomness for the encryption) attesting that c is a valid encryption of x.
2. The sender S verifies that π_{ZK}^{R} is valid and aborts if not. The sender S then homomorphically evaluates $f(\cdot, y)$ on the cipher c (the evaluation is randomized to keep y private) and sends the resulting evaluated cipher $\hat{c}$, together with π_{ZK}^{S}, a zkSNARK proving knowledge of y (and randomness for the evaluation algorithm) attesting that $\hat{c}$ is a valid (homomorphic) evaluation of $f(\cdot, y)$ on c.
3. The receiver R verifies that π_{ZK}^{S} is a valid zkSNARK and, if so, outputs the decryption of $\hat{c}$ or else $\perp$.

We stress that the amount of work done by R (including his NIZK π_{ZK}^{R}) is independent of f's complexity. We next briefly describe how each party is simulated in order to establish non-concurrent security.

Proposition 10.2. *In the CRS model, assuming zkSNARKs and fully homomorphic encryption, there above protocol is a SNISC against static malicious corruptions. That is, either a malicious receiver or sender can be simulated given only access to a trusted party computing $f(\cdot, \cdot)$.*

Proof Sketch. We explain how each party is simulated.

Simulating a malicious R^*. To simulate R^*, we proceed as follows. First, generate the CRS for the NIZK of knowledge (together with a trapdoor if needed) and then provide R^* with the CRS to obtain $(\mathsf{vgrs}, c, \pi_{\mathsf{ZK}}^{R})$. In case the NIZK π_{ZK}^{R} does not verify, abort; otherwise, use the trapdoor to extract the input x, hand it to the trusted party, and receive $f(x, y)$. To simulate the message sent by S, simulate an FHE evaluation $\hat{c}$ with underlying plaintext $f(x, y)$; this can be done by the function privacy guarantee. Next, invoke the simulator for the zkSNARK with respect to the statement given by the simulated evaluation ciphertext $\hat{c}$. The validity of the simulation follows from the function privacy guarantee of the FHE and the validity zkSNARK simulator, as well as from the fact that the NIZK in use is a proof of knowledge.

Simulating a malicious S^*. To simulate S^*, we proceed as follows. Generate the CRS together with a trapdoor for the NIZK of knowledge. Simulate R's message by encrypting an arbitrary string (of the proper length) to create a simulated encryption c and running the NIZK simulator with respect to the statement given by c. Also, simulate the vgrs by employing the generator of the zkSNARK. Feed S^* with the generated message to obtain a proof π^R_{ZK}. Check the validity of π^R_{ZK} using the CRS and the private verification state generated with vgrs. If the proof is invalid abort; otherwise, use the zkSNARK extractor to obtain the input y and hand it to the trusted party. (Note that here the simulation makes non-black-box use of the adversary S^*, because the extractor for S^* guaranteed by the knowledge property of the zkSNARK depends on the code of S^*.) The validity of the simulation follows from the semantic security of the encryption and the validity of the NIZK simulator, as well as from the fact that the zkSNARK is a proof of knowledge.

More Generally We note that the FHE-based protocol above can be naturally generalized to yield a simple compiler that transforms any SNISC protocol in a slightly strengthened semi-honest model to a (still non-interactive) protocol that is secure against malicious parties in the CRS model. The strengthening of the semi-honest model is to require security with respect to parties that may choose arbitrary randomness.[18]

Non-concurrent Versus UC and the Length of the Sender's Input The above protocol achieves the standard "stand-alone" security guarantee. We note, however, that in the case where the input of the sender S is short, we can, in fact, obtain the stronger notion of universal composability (UC) [35]. Concretely, the gap is that, in order to achieve UC security the simulator for either party has to be straight-line (non-rewinding) and black-box in the code of malicious parties. In the above analysis this is violated when simulating a malicious sender S^*. Indeed, extraction of its input relies on the non-black-box zkSNARK extraction. When the sender's input is short (still the computation can be long), we can combine a standard NIZK of knowledge and a zkSNARG to avoid this. More accurately, we can devise a solution where the complexity of the receiver grows with the size of the sender's input, but not its computation.

Concretely, in the augmented protocol S^* acts as follows: He gives a NIZK of knowledge π' attesting that he knows his own input y and then uses a zkSNARG to prove that "there exists y (as well as randomness for the evaluation algorithm and randomness for the NIZK) for which $\hat{c}$ is a valid evaluation of $f(\cdot, y)$ on c, *and* y is the witness for the NIZK π'." Now the simulation of a malicious S^* can be performed in a black-box manner, by simply invoking the black-box extractor of the NIZK proof of knowledge; a non-black-box use of S^* is only made within the proof when the (computational) soundness of the zkSNARG is invoked.

11. Extractable One-Way Functions and Their Applications

In this section, we consider a notion that is closely related to ECRHs: *extractable one-way functions* (EOWFs). Specifically, we formalize two strong variants of EOWFs and show

[18] In the interactive setting, this assumption is not needed—the GMW compiler [74] deals with this issue using "coin tossing into the well," which cannot be done in the non-interactive setting.

that, assuming the existence of EOWFs and enhanced trapdoor permutations, there exist a *non-interactive* (two-message) selective-opening-attack-secure (SOA-secure) commitment scheme and a *three-round* concurrent zero-knowledge (ZK) argument for any language in NP. Previous works showed that it is impossible to construct the former primitive from standard assumptions using black-box security reductions [111] (provided one-to-one one-way functions exist) and that it is impossible to have sublogarithmic-round concurrent zero-knowledge protocols with black-box simulation [41]. Our constructions circumvent previous impossibility results by relying on the (non-black-box) extractability property of EOWFs.

The rest of this section is organized as follows. In Sect. 11.1, we formalize two variants of EOWFs: *strong extractable* defs (sEOWFs) and *strong concurrently extractable* defs (scEOWFs). Next, in Sect. 11.4, we use sEOWFs to construct a non-interactive SOA-secure commitment scheme; the technical core of our construction is obtaining a new three-round zero-knowledge argument of knowledge (ZKAOK) for NP having the special property that only the last message of the protocol depends on the statement to be proved; we believe this construction is of independent interest, and we present it first separately in Sect. 11.2. Next, in Sect. 11.3, we show that when the sEOWF used in our three-round ZKAOK protocol is replaced with a scEOWF, the protocol is also concurrent ZK, yielding a three-round concurrent ZK protocol. Finally, in Sect. 11.5 we give candidate constructions of sEOWFs and scEOWFs, based on the (original) Knowledge of Exponent assumption (KEA) of [48].

11.1. *Definitions of sEOWFs and scEOWFs*

A *strong extractable* def (sEOWF) is an ensemble of extractable functions that are one-to-one and everywhere one-way (namely, for every sufficiently large security parameter, *every* function in the family is hard to invert); furthermore, given a function, it is possible to efficiently verify whether the function belongs to the ensemble or not.

Definition 11.1. Let $\mathcal{F} = \{\mathcal{F}_k\}_k$ be a function ensemble where for every $k \in \mathbb{N}$, every function $f \in \mathcal{F}_k$ is described as a binary string of polynomial length $m(k)$ and maps $\{0,1\}^k$ to $\{0,1\}^{\ell(k)}$. $\mathcal{F}$ is an sEOWF if it is extractable (see Definition 1) and additionally satisfies the following properties:

1. **One-to-one:** every function in $\mathcal{F}$ is one-to-one (so that $\ell(k) \geq k$).
2. **Verifiability:** there is a polynomial-time algorithm that, given $k \in \mathbb{N}$ and f, decides whether f is in $\mathcal{F}_k$.
3. **Everywhere one-wayness:** For every polynomial-size adversary $\mathcal{A}$ and polynomial m, every sufficiently large security parameter $k \in \mathbb{N}$, every function $f \in \mathcal{F}_k$, and every auxiliary input $z \in \{0,1\}^{m(k)}$, the following holds:

$$\Pr_{x \leftarrow \{0,1\}^k}\left[f(x') = f(x) \middle| \begin{array}{r} y \leftarrow f(x) \\ x' \leftarrow \mathcal{A}(1^k, z, y) \end{array} \right] \leq \mathrm{negl}(k) \ .$$

Remark 11.2. (sEOWFvs. EPOW) Our notion of sEOWF is similar to the notion of extractable perfectly one-way (EPOW) functions defined by Canetti and Dakdouk [37].

While both notions seek to formalize the notion of extractability for a one-way function, the two notions differ in their concrete hardness requirements: A EPOW requires every function in the ensemble to be a perfectly one-way function [34] (which is a probabilistic function whose images hides all partial information of the preimage) whereas an sEOWF ensemble only requires every function in the ensemble to be hard to invert.

We now prove a useful lemma that says that if a function ensemble $\mathcal{F}$ is extractable, then it is also *parallel extractable*, in the sense that if the adversary outputs, in parallel, many values for different functions, then, for every value that is in the image of the corresponding function, the extractor succeeds in extracting a corresponding preimage.

Lemma 11.3. *If a function ensemble $\mathcal{F} = \{\mathcal{F}_k\}_k$ is extractable then it is also parallel extractable in the following sense: For any polynomial-size adversary $\mathcal{A}$ and polynomials m, t, there exists a polynomial-size extractor $\mathcal{E}$ such that for any $k \in \mathbb{N}$, and any auxiliary input $z \in \{0,1\}^{m(k)}$:*

$$\Pr_{f \leftarrow (\mathcal{F}_k)^{t(k)}} \left[\begin{array}{c} \exists\, i \in [t(k)] \text{ s.t.} \\ y_i \in \mathsf{Image}(f_i) \text{ and } y_i \neq f_i(x'_i) \end{array} \middle| \begin{array}{l} \boldsymbol{y} \leftarrow \mathcal{A}(\boldsymbol{f}, z) \\ \mathbf{x}' \leftarrow \mathcal{E}(\boldsymbol{f}, z) \end{array} \right] \leq \operatorname{negl}(k) \ .$$

Proof. Consider the adversary $\tilde{\mathcal{A}}$ that, on input $f_i, z'_i = (z, i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_{t(k)})$, runs $\mathcal{A}(\mathbf{f}, z)$, except that it only outputs the ith value x_i that A outputs. Let $\tilde{\mathcal{E}}$ be the extractor corresponding to $\tilde{\mathcal{A}}$. Then the parallel extractor $\mathcal{E}$ for $\mathcal{A}$ simply internally runs $\tilde{\mathcal{E}}$ in parallel with inputs f_i, z'_i for every i and outputs the values extracted by the parallel executions of $\tilde{\mathcal{E}}$. By definition of $\tilde{\mathcal{E}}$, except with negligible probability the ith invocation of $\tilde{\mathcal{E}}$ returns a valid preimage for the ith value the adversary $\mathcal{A}$ outputs. Thus, overall, $\mathcal{E}$ is a valid parallel extractor.

Next, a *strong concurrently extractable* def (scEOWF) is an sEOWF where the extractability property is strengthened to *concurrent extractability*:

Definition 11.4. An scEOWF is an sEOWF that is concurrently extractable (see below).

We now describe the notion of concurrent extractability. The adversary and extractor participate in an *interactive* game, in which the adversary adaptively outputs multiple values while receiving the preimages the extractor obtains for each output value. Concurrent extractability requires that for every adversary there is an extractor such that, in the interactive game, the extractor succeeds (with overwhelming probability) in extracting the correct preimage for every value the adversary outputs.

Definition 11.5. (*Concurrent extractability*) A function ensemble $\mathcal{F} = \{\mathcal{F}_k\}_k$ mapping $\{0,1\}^k$ to $\{0,1\}^{\ell(k)}$ is *concurrently extractable* if for any polynomial-size oracle machine $\mathcal{A}$ and polynomials m, t there exists a polynomial-size extractor $\mathcal{E}$ such that for every security parameter $k \in \mathbb{N}$, and any auxiliary input $z \in \{0,1\}^{m(k)}$, the advantage of $\mathcal{A}$ against $\mathcal{E}$ in the experiment $\mathsf{EXP}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k)$ (described below) is $\operatorname{negl}(k)$.

The definition of $\mathsf{EXP}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k)$ is as follows:

```
$\mathsf{EXP}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k) \equiv$
  1. $f_1 \cdots f_{t(k)} \leftarrow (\mathcal{F}_k)^{t(k)}$;
  2. $\mathsf{st}_0 \leftarrow (f_1, \cdots, f_{t(k)}, z)$;
  3. run $\mathcal{A}^{\mathcal{O}}(\mathsf{st}_0)$ until it halts, replying to the $i$th oracle query $(f_{j_i}, y_i)$ from $\mathcal{A}$ to
     $\mathcal{O}$ as follows:
     (a) $(x_i, \mathsf{st}_i) \leftarrow \mathcal{E}(f_{j_i}, y_i, \mathsf{st}_{i-1})$;
     (b) if $f_{j_i} \in \{f_1, \cdots, f_{t(k)}\}$, $y_i \in \mathsf{Image}(f_{j_i})$, and $y_i \neq f_{j_i}(x_i)$, abort and
         output $\mathsf{fail}$;
     (c) return $x_i$ to $\mathcal{A}$;
  4. output 1.
```

The advantage of $\mathcal{A}$ against $\mathcal{E}$ in the above game, denoted as $\mathsf{ADV}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k)$, is the probability that the game's output is fail; that is,

$$\mathsf{ADV}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k) := \Pr\left[\mathsf{EXP}^{\mathcal{F}}_{\mathcal{A},\mathcal{E}}(1^k) = \mathsf{fail}\right].$$

Remark 11.6. Concurrent extractability is very similar to the notion of extractability required of extractable hash functions in Definition 2.8 of [52], except that, in their security game, the adversary attacks only a single randomly chosen hash function, whereas in the above definition, the adversary is allowed to choose to attack any of an arbitrary polynomial number of functions adaptively.

Remark 11.7. (Auxiliary input) As noted after Definition 6.1, EOWF (and surely sEOWF or scEOWF) cannot be achieved with respect to arbitrary auxiliary input distributions of a priori unobunded polynomial size, assuming indistinguishability obfuscation. In the applications that follow, we use EOWFs within a larger protocol and cannot necessarily restrict the distribution on the auxiliary input. (For example, in a zero-knowledge protocol, the instance itself can been seen as auxiliary input that is adversarially chosen in a worst-case manner. Yet, using similar techniques to those used in [16], we can make sure that any auxiliary input is a priori bounded by a fixed polynomial, a setting in which no impossibility results for EOWFs are known. Thus, all of our results can be scaled down to consider only verifiers with bounded auxiliary information.

In Sect. 11.5, we discuss candidate constructions of sEOWFs and scEOWFs.

11.2. *A Special Three-Round ZKAOK Protocol*

In this section, we construct a three-round ZKAOK protocol for NP that has the special property that only the last message of the protocol depends on the statement to be proved, whereas the first two messages only depend on the size of the statement. Therefore, the statement only needs to be specified before the generation of the last message. Our construction relies on three building blocks:

1. An sEOWF family ensemble $\mathcal{F}$.
2. A ZAP protocol (P_Z, V_Z) for NP, that is, a two-round public-coin witness indistinguishable (WI) proof. Such protocols exist assuming the existence of trapdoor permutations [57].
3. A three-round WIAOK protocol (P_W, V_W) for NP that has two special properties. First, it satisfies that only the last message in the protocol depends on the statement to be proved (and the first two messages depends only on the size of the statement). Second, it has a strong argument of knowledge property. Namely, given two accepting transcripts (m_1, m_2^1, m_3^1) and (m_1, m_2^2, m_3^2) for two different statements x_1 and x_2 that have the same first message but different second messages $m_2^1 \neq m_2^2$, a witness of either x_1 or x_2 can be deterministically computed—we refer to this property as the *special soundness* property. It has been shown in [101] that such a protocol can be constructed from one-to-one one-way functions.

Given these building blocks, the three-round ZKAOK protocol (P, V) proceeds as follows: To prove a NP statement x, the prover P and the verifier V on common input 1^k and private input a witness w of x to P exchange the following three messages.

First message: The prover sends the following:

- A randomly sampled function $f \leftarrow \mathcal{F}_k$,
- the first message α of a (P_Z, V_Z) proof where the prover acts as the receiver, and
- the first message m_1 of a (P_W, V_W) proof where the prover acts as the prover.

Second message: The verifier sends the following:

- The images y_1, y_2 of two randomly sampled k-bit strings r_1, r_2 through the function f,
- the second message β of (P_Z, V_Z) in response to α, proving that y_1 or y_2 is in the range of f (the honest verifier uses the preimage of y_b chosen at random as the witness), and
- the second message m_2 of (P_W, V_W) in response to m_1.

Third Message: The committer sends the following:

- The third message m_3 of (P_W, V_W) in response to m_1, m_2, proving that either x is true, or either y_1 or y_2 is in the range of f. (The honest prover uses the witness w of x as the witness.)

By construction, it is easy to see that the first two messages of the protocol do not depend on the statement to be proved. Next we first show in Lemma 11.8 that indeed this protocol is a ZKAOK for NP. Then we observe in Lemma 11.9 that by slightly extending the proof of Lemma 11.8, we can in fact show that (P, V) satisfies two stronger properties, namely, parallel ZK and *adaptive soundness*, where the latter guarantees that no efficient prover can prove a false statement with non-negligible probability, even if it can adaptively choose the false statement to prove adaptively depending on the verifier's message (before the generation of the last message).

Lemma 11.8. *The protocol (P, V) is a ZKAOK for* NP.

Proof. We first show that (P, V) is ZK. Fix a malicious verifier V^*, a security parameter k and an auxiliary input z to V^*. We construct the simulator S for V^*. S internally runs V^* with input $(1^k, z)$ and a uniformly sampled random string r, and emulates the prover's messages as follows:

- S emulates the first message (f, α, m_1) to V^* honestly;
- Upon receiving the second message (y_1, y_2, β, m_2) from V^*, it tries to extract a preimage of y_1 or y_2 by relying on the extractability property of sEOWF. More precisely, consider a wrapper machine $\mathcal{A}$ that on input f and auxiliary input $z' = (1^k, z, r, \alpha, m_1)$ runs V^* internally with input $(1^k, z)$, random tape r and first message (f, α, m_1), and outputs the two images y_1, y_2 output by V^*. It follows from the (parallel) extractability of the sEOWF $\mathcal{F}$ that there is an extractor $\mathcal{E}$ that on the same input f, z' outputs x_1, x_2 such that x_i is a valid preimage of y_i as long as y_i is in the range of y with overwhelming probability. The simulator internally incorporates $\mathcal{E}$ and runs it with input f, z' to obtain x_1 and x_2. If neither x_1 nor x_2 is a valid preimage, it aborts and outputs fail. Otherwise, it records a valid preimage x_i.
- S simulates the third message m_3 by proving that one of y_1, y_2 is in the range of f using x_i as the witness.

We show that S emulates the view of V^* correctly. First, it follows from the soundness of the ZAP protocol (P_Z, V_Z) that at least one of the images y_1 and y_2 output by V^* is in the range of f. Then by the extractability of $\mathcal{F}$, $\mathcal{E}$ succeeds in extracting at least one valid preimage with overwhelming probability, and thus S outputs fail with only negligible probability. Whenever $\mathcal{E}$ succeeds in extracting a valid preimage, S uses it as a "trapdoor" to cheat in the last message. It then follows from the witness indistinguishability property of the protocol (P_W, V_W) that the simulated view is indistinguishable from the real view.

Next we show that (P, V) is an AOK for NP. Consider an efficient prover P^* (w.l.o.g. deterministic) that for infinitely many $k \in \mathbb{N}$ and auxiliary input z, proves a statement x with probability $1/p(k)$. We show that there is an efficient extractor $\mathcal{E}$, that on input $(1^k, x, z)$ and with black-box access to P^* extracts a valid witness w of x with probability at least $1/q(k) = 1/p(k)^2 - \text{negl}(k)$. The extractor $\mathcal{E}$ simply emulates two executions of P^* with an honest verifier V, obtaining two transcripts T_1, T_2, which share the same first message (f, α, m_1), and have different second and third messages $(y_1^1, y_2^1, \beta^1, m_2^1), m_3^1$ and $(y_1^2, y_2^2, \beta^2, m_2^2), m_3^2$; if both T_1 and T_2 are accepting, then it extracts a witness w from the two transcripts (m_1, m_2^1, m_3^1) and (m_1, m_2^2, m_3^2) of (P_W, V_W); it outputs w if it is indeed a valid witness of x; otherwise, it outputs fail.

We argue that $\mathcal{E}$ extracts a valid witness with probability at least $1/q(k) = 1/p(k)^2 - \text{negl}(k)$. To see this, first note that since $\mathcal{E}$ emulates two executions between P^* and the honest verifier V honestly, it happens with probability $1/p(k)^2$ that the two transcripts T_1 and T_2 collected are both accepting proofs for statement x. We show below that conditioned on this happening, except with negligible probability, the value w extracted from the two proofs of (P_W, V_W) must be a valid witness of x. If so, the probability that $\mathcal{E}$ extracts a valid witness successfully is at least $1/p(k)^2 - \text{negl}(k)$.

Assume for contradiction that conditioned on that T_1 and T_2 are both accepting proofs of x, the value w extracted from (m_1, m_2^1, m_3^1) and (m_1, m_2^2, m_3^2) in T_1 and T_2 is not a valid witness of x with non-negligible probability $1/p'(k)$. Then we can construct a machine B that violates the everywhere one-wayness of $\mathcal{F}$. The machine B on input $(1^k, x, z)$ internally proceeds as $\mathcal{E}$ does, except that, when emulating the two executions between P^* and V, it first forwards the function f from P^* externally; then upon receiving a challenge y which is the image of a random value $y = f(r)$, it assigns y to one of $y_1^1, y_2^1, y_1^2, y_2^2$ at random—let it be y_b^d—and generates the other three values honestly; furthermore, in the d^{th} execution, it emulates the second message of ZAP by proving that y_{1-b}^d is in the range of f. Finally, after extracting the value w, it checks if w is a preimage of y; if so, it outputs w; otherwise, it outputs fail. Since B emulates the view of P^* perfectly and extracts w as $\mathcal{E}$ does, by our hypothesis, it occurs with probability $1/p'(k)$ that T_1, T_2 are accepting but the extracted value w is not a valid witness of x. Then it follows from the special soundness of $\langle P_W, V_W\rangle$ that except with negligible probability w must be a valid witness of either the statement of (m_1, m_2^1, m_3^1) or that of (m_1, m_2^2, m_3^2); in other words, w must be a preimage of one of $y_1^1, y_2^1, y_1^2, y_2^2$. Furthermore, it follows from the fact that the distribution of the verifier's message is statistically close in the two executions and the WI property of ZAP that which of the four values y is assigned to is computationally hidden. Therefore, with probability $1/4 - \text{negl}(k)$, w is a preimage of y. Therefore B violates the everywhere one-wayness of $\mathcal{F}$ with probability at least $1/5p'(n)$, which gives a contradiction.

Lemma 11.9. *The protocol* (P, V) *is parallel ZK and has adaptive soundness.*

Proof Sketch. The proof of the parallel ZK property can be easily extended from that of the ZK property. Given a malicious verifier V^*, to simulate many parallel proofs of (P, V) to it, simply consider a simulator that simulates the prover's message in each parallel proof as how the simulator in the proof of Lemma 11.8 simulates a single proof, except that it extracts the "trapdoors" in all parallel proofs by relying on the parallel extractability property of the sEOWF as shown in Lemma 11.3.

Next we show that (P, V) has adaptive soundness. Assume for contradiction that there is a malicious prover P^*, w.l.o.g. deterministic, such that for infinitely many $k \in \mathbb{N}$ and auxiliary input z, it can prove a false statement of its choice with a non-negligible probability $1/p(k)$. Then we show that we can construct a machine B' that can violate the everywhere one-wayness of $\mathcal{F}$. The machine B' proceeds identically to the machine B constructed in the proof of the AOK property of Lemma 11.8. As argued in the proof of Lemma 11.8, by our hypothesis, with probability $1/p(k)^2$, B' obtains two accepting transcripts T_1 and T_2 for two (potentially different) false statements x_1 and x_2. Then by the special soundness of (P_W, V_W), B' must extract a preimage w of one of the four values $y_1^1, y_2^1, y_1^2, y_2^2$ with probability $1/q(k) = 1/p(k)^2 - \text{negl}(k)$. Then by the same argument as in the proof of Lemma 11.8, w must be a preimage of the value y that B' receives externally with probability at least $1/5q(k)$. Therefore B' violates the everywhere one-wayness of $\mathcal{F}$ and this gives a contradiction.

Remark 11.10. Bitansky, Canetti, Paneth, and Rosen [16] show that when considering a restricted class of adversaries with *bounded* polynomial advice and unbounded polynomial running time, a weaker variant of extractable one-way functions can be constructed from standard assumptions (e.g., subexponential security of Decision Diffie–

Hellman or Quadratic Residuosity). They then show how to use this weaker variant of extractable one-way functions to construct two-message zero-knowledge arguments and three-message zero-knowledge arguments of knowledge against adversaries of the same class. Their protocols follow the same structure as our construction above, with modifications tailored to their weaker extractable one-way functions.

11.3. *A Three-Round Concurrent ZK Protocol*

In this section, we first recall the definition of concurrent ZK and then show that assuming that the underlying strong extractable def's are concurrently extractable, that is, an scEOWF, then the three-round ZKAOK protocol (P, V) described in Sect. 11.2 is a concurrent ZK protocol for NP.

Definition of Concurrent Zero-Knowledge Protocols Let (P, V) be an interactive argument for a language L. Consider a concurrent adversarial verifier V^* that on common input 1^k, x and auxiliary input z, interacts with *any polynomial number* of independent copies of P concurrently, without any restrictions over the scheduling of the messages in the different interactions with P. Let $\mathrm{View}_{V^*}^{P}(1^k, x, z)$ denote the random variable describing the view of the adversary V^* in an interaction with P.

Definition 11.11. Let (P, V) be an interactive argument system for a language L. We say that (P, V) is concurrent ZK if for every PPT concurrent adversary V^*, there exists a PPT simulator $\mathcal{S}$, such that, it holds that the ensembles $\{\mathrm{View}_{V^*}^{P}(1^k, x)\}_{k\in\mathbb{N}, x\in\{0,1\}^k\cap L}$ and $\{\mathcal{S}(1^k, x)\}_{k\in\mathbb{N}, x\in\{0,1\}^k\cap L}$ are computationally indistinguishable over $k \in \mathbb{N}$.

It was shown in [41] that it is impossible to construct a slightly sublogarithmic-round concurrent ZK protocol with black-box simulation, where the simulator $\mathcal{S}$ only uses black-box access to the malicious verifier V^*. Next, by relying on the existence of a scEOWF and enhanced trapdoor permutations, we show that there is a three-round concurrent ZK protocol, which circumvents the impossibility result by using non-black-box simulation. We note that in a recent work [82], Gupta and Sahai formulated a new knowledge assumption and showed that it implies constant-round concurrent ZK arguments. Although their knowledge assumption is formulated in a "stand-alone" fashion, it essentially implies concurrent extractability; moreover, their protocol has five rounds.

The Three-Round Protocol (P, V) *is Concurrent ZK* We show that assuming that the underlying family ensemble $\mathcal{F}$ used in the protocol (P, V) in Sect. 11.2 is an scEOWF, then the protocol (P, V) is concurrent ZK.

Theorem 11.12. *Let $\mathcal{F}$ be an scEOWF. Then (P, V) is concurrent ZK.*

Proof. Fix a PPT concurrent adversary V^*, a security parameter $k \in \mathbb{N}$, a statement x, and an auxiliary input z. We first construct a simulator $\mathcal{S}'$ that, with access to an oracle $\mathcal{O}$ that inverts the one-way functions in $\mathcal{F}$, simulates the view of V^*. More precisely, let $\mathcal{O}$ be an oracle satisfying that, if it is fed with a pair (f, y) with $f \in \mathcal{F}$ and y in the range of f, it returns a valid preimage through f; (otherwise, it can return any value).

The machine $\mathcal{S}'$ on auxiliary input $z' = (1^k, x, z)$ and a vector of randomly chosen functions $\mathbf{f} \leftarrow \mathcal{F}$, internally simulates a concurrent execution with $V^*(1^k, x, z)$ as follows: It emulates the first messages for V^* honestly by forwarding f_i as the randomly chosen function the ith interaction; it simulates the third messages using a "fake" witness—a preimage of one of the two values y_1 or y_2 from V^* sent in the second message—and cheating in the WI proof that either y_1 or y_2 is in the range of f (instead of proving that x is true). For every interaction of (P, V), $\mathcal{S}'$ obtains a "fake" witness by querying the oracle on the two values y_1 and y_2 from V^*, obtaining x_1 and x_2; if neither x_1 nor x_2 is a valid preimage of y_1 or y_2, $\mathcal{S}'$ aborts and outputs fail; otherwise, $\mathcal{S}'$ records a valid preimage and later simulates the ZAP proof of this interaction using it as a "fake" witness. Finally, $\mathcal{S}'$ outputs the simulated view of V^*. By the soundness of ZAP proof from the malicious verifier, except with negligible probability, for every interaction of (P, V), one of the two values y_1 and y_2 has a valid preimage; then, the oracle must return one valid preimage, and thus the probability that $\mathcal{S}'$ outputs fail is negligible. In this case, it follows directly from the witness indistinguishability of (P_W, V_W) that the simulated view of V^* by $\mathcal{S}'$ is indistinguishable from the real view of V^* when interacting with an honest prover.

Next, we construct the actual simulator $\mathcal{S}$ that emulates the oracle $\mathcal{O}$ for $\mathcal{S}'$ by relying on the concurrent extractability of $\mathcal{F}$. More precisely, by the concurrent extractability of $\mathcal{F}$, there is an extractor $\mathcal{E}$ such that when replacing the oracle answers to $\mathcal{S}'$ with the preimages extracted by $\mathcal{E}(\mathbf{f}, z')$, the probability that $\mathcal{E}$ fails to extract a valid preimage for a value output by $\mathcal{S}'$ that has a preimage is negligible. Furthermore, it follows from the one-to-one property of $\mathcal{F}$ that except with negligible probability, $\mathcal{E}$ emulates the oracle $\mathcal{O}$ perfectly. Therefore, the output view of $\mathcal{S}$ is indistinguishable from the real view of V^*, and we conclude the theorem.

Beyond Concurrent ZK By applying the transformation of [21] to our protocol, we obtain a three-round resettably sound concurrent ZK protocol. By additionally applying the transformation of [54] to the resulting resettably sound concurrent ZK protocol, we obtain a three-round *simultaneously resettable* ZK protocol.

Theorem 11.13. *Assuming the existence of an scEOWF and enhanced trapdoor permutations, there is a three-round simultaneously resettable ZK protocol.*

11.4. *Two-Message Selective-Opening-Attack Secure Commitments*

In this section, we first provide a formal definition of a SOA-secure commitment scheme and then provide a non-interactive construction of it using the three-round adaptively sound parallel ZK protocol constructed in Sect. 11.2.

Definition of SOA-Secure Commitments A commitment scheme secure against selective opening attack has a strong hiding property that holds even if the adversary gets to selectively ask for the decommitment of some of the commitments it receives. We consider a strong notion of SOA security, which is essentially the same as the simulation based definition of a SOA-secure commitment scheme in [58], but strengthens it to require indistinguishability of the simulation rather than a relation-based security guarantee. Formally, let (C, R) be a commitment scheme. We compare between a real and an ideal execution. In the real experiment, the adversary gets to interact with the honest committer C in the commit stages of $t = t(k)$ commitments to values $v_1, \ldots, v_t$ sampled from some distribution D and may adaptively ask for decommitments of any commitment c_i, where i is a part of a "legal" subset $I \subseteq \{1, \cdots t(k)\}$. In the ideal experiment, the adversary simply gets to ask for the values v_i in the legal subset $i \in I$. Let $\mathsf{real}((C, R), D, I, \mathcal{A}, 1^k)$ denote the view of an adversary $\mathcal{A}$ in the following experiment:

- Sample $(\mathbf{x}, z)$ from D, and for each $i \in |\mathbf{x}|$, engage with the adversary $\mathcal{A}(z)$ in the commit stage of (C, R) committing to value x_i, where $\mathbf{x}$ is a vector of length $t(k)$ and x_i is the ith component of $\mathbf{x}$.
- $\mathcal{A}$ chooses a subset $J \subseteq I$. For every $j \in J$, decommit the jth commitment to value x_j by sending the corresponding decommitment string d_j.

Let $\mathsf{ideal}(D, I, \mathcal{S}, 1^k)$ denote the output of the machine $\mathcal{S}$ in the following experiment:

- Sample $(\mathbf{x}, z)$ from D. Feed $(1^k, z)$ to $\mathcal{S}$.
- $\mathcal{S}$ chooses a subset $J \subseteq I$. For every $j \in J$, feed x_j to $\mathcal{S}$.

Definition 11.14. (*Hiding under selective decommitment*) Let (C, R) be a commitment scheme. We say that (C, R) is secure under selective decommitment w.r.t. the legal set $\mathcal{I} = \{I_k\}_{k\in\mathbb{N}}$, where $I_k \in [t(k)]$ and the ensemble of distributions $\mathcal{D} = \{D_k\}_{k\in\mathbb{N}}$, where D_k is a distribution over $(\{0,1\}^{\mathrm{poly}(k)})^{t(k)+1}$; if for every non-uniform PPT adversary $\mathcal{A}$, there exists a non-uniform PPT $\mathcal{S}$ such that the following two ensembles are indistinguishable.

- $\left\{\mathsf{real}((C, R), D_k, I_k, \mathcal{A}, 1^k)\right\}_{k\in\mathbb{N}}$
- $\left\{\mathsf{ideal}(D_k, I_k, \mathcal{S}, 1^k)\right\}_{k\in\mathbb{N}}$

It is implied by the impossibility result in [111] that assuming the existence of one-to-one one-way functions, then there exist legal set $\mathcal{I}$ and distribution $\mathcal{D}$, such that it is impossible to construct a non-interactive SOA-secure commitment scheme w.r.t. $\mathcal{I}$ and $\mathcal{D}$ based on any *bounded-round* assumptions[19] via a black-box security reduction.

In contrast to the impossibility result, next we show that assuming the existence of an sEOWF and enhanced trapdoor permutations, we can construct a non-interactive SOA-secure commitment scheme. Our construction circumvents the impossibility result of [111] at two aspects: First, the assumption of the existence of sEOWF is not a bounded-round assumption, and second, our security reduction is non-black-box.

A non-interactive SOA-secure commitment scheme $(\tilde{C}, \tilde{R})$. The construction makes use of the above three-round adaptively sound parallel ZK protocol (P, V) and a basic one-message statistically binding commitment scheme Com, which exists assuming one-to-one one-way functions. To commit to a value v, the committer $\tilde{C}$ and the receiver $\tilde{R}$ on common input a security parameter 1^k proceeds as follows:

First message: The committer sends a commitment c to v using Com, together with the first message m_1 of (P, V).
Second message: The receiver replies with the second message m_2 of (P, V).

[19] A bounded-round assumption is an intractability assumption formalized using a bounded-round interactive game between an adversary and a challenger. It is similar to the notion of falsifiable assumption proposed by [107] but does not require the challenger to be efficient [111].

Decommitment message: The committer sends v and the third message m_3 of (P, V) proving that v is indeed the value committed to in c. The receiver accepts if the proof (m_1, m_2, m_3) is accepting.

Theorem 11.15. *$(\tilde{C}, \tilde{R})$ is computationally binding and computationally hiding under selective decommitment w.r.t. any distribution ensemble $\mathcal{D}$ and legal set ensemble $\mathcal{I}$.*

Proof Sketch. It follows from the adaptive soundness of (P, V) and the statistically binding property of Com that the commitment scheme $(\tilde{C}, \tilde{R})$ is computationally binding. To show that it is also hiding under selective decommitment, consider a distribution ensemble $\mathcal{D} = \{D_k\}_k$, a legal set ensemble $\mathcal{I} = \{I_k\}_k$ and an adversary $\mathcal{A}$, and fix a security parameter $k \in \mathbb{N}$. We construct a simulator $\mathcal{S}$ that in an ideal experiment $\mathsf{ideal}(D_k, I_k, \mathcal{S}, 1^k)$ proceeds as follows:

- Upon receiving $1^k, z$, $\mathcal{S}$ internally runs $\mathcal{A}(1^k, z)$, and sends the first messages of $t = t(k)$ commitments of 0^n to $\mathcal{A}$.
- Upon receiving the request from $\mathcal{A}$ for the decommitment of commitments in subset $J \subseteq I_k$, $\mathcal{S}$ forwards J externally and obtains values x_j for every $j \in J$.
- $\mathcal{S}$ sends $\mathcal{A}$ x_j's and simulates the decommitment messages to x_j's, which are simply the third messages of (P, V), by relying on the parallel ZK property of (P, V).

It follows from the parallel ZK property of (P, V) and the hiding property of Com that the simulation is indistinguishable.

11.5. *Candidate Constructions of sEOWF and scEOWF*

A candidate construction of sEOWF. We show that the construction of ECRH in Sect. 8.1 when instantiated with the 1-KEA assumption (Assumption 8.1 in Sect. 8.1) is essentially already a sEOWF (up to a slight modification). For completeness, we provide the construction below.

Let $\mathcal{G} = \{\mathcal{G}_k\}$ be an efficiently samplable ensemble for which the 1-KEA assumption holds. That is, each $\mathcal{G}_k$ consists of a group $\mathbb{G}$ of prime order $p \in (2^{k-1}, 2^k)$ and a generator $g \in \mathbb{G}$, and it holds that for any polynomial-size adversary $\mathcal{A}$ there exists a polynomial-size extractor $\mathcal{E}$, such that, whenever $\mathcal{A}$, given $(g^r, g^{\alpha r}, z)$ for r and α chosen randomly from $\mathbb{Z}_p$, outputs a valid tuple (c, c') such that $c' = c^\alpha$, the extractor $\mathcal{E}(g^r, g^{\alpha r}, z)$ finds a discrete logarithm x such that $g^{xr} = c$. Now an sEOWF can be constructed from the KEA assumption as follows:

- To sample from $\mathcal{F}_k$: set $(\mathbb{G}, g) \leftarrow \mathcal{G}_k$ and sample $(r, \alpha) \overset{U}{\leftarrow} \mathbb{Z}_p^* \times \mathbb{Z}_p^*$ where $p = |\mathbb{G}|$, and output $f := (\mathbb{G}, g^r, g^{\alpha r})$. (Note that, different from the construction of ECRH in Sect. 8.1, α and r are sampled from $\mathbb{Z}_p^*$ instead of $\mathbb{Z}_p$.)
- To compute $f(x)$: output the pair $(g^{xr}, g^{x\alpha r})$.

Assuming the 1-KEA assumption and that the hardness of computing discrete logarithms in $\mathcal{G}$, the above construction is a sEOWF. First, it follows directly from the 1-KEA assumption and almost the same argument for the construction of ECRH from t-KEA in Sect. 8.1 that the above construction is extractable. Furthermore, by construction, g^{xr} for every $r \in \mathbb{Z}_p^*$ uniquely determines x and thus every function in the ensemble is one-to-one. Finally, it follows from the hardness of discrete logs that for sufficiently large security parameter k, every function in the family $\mathcal{F}_k$ is hard to invert.

A candidate construction of scEOWF. To obtain a scEOWF, we use the same construction as above, but assuming a stronger *concurrent* 1-KEA assumption, which simply extends the 1-KEA assumption to allow concurrent extraction in a similar way as in the definition of concurrent extractability property (Definition 11.5).

Assumption 11.16. (Concurrent 1-KEA) There exists a family $\mathcal{G} = \{\mathcal{G}_k\}$ where each $\mathcal{G}_k$ consists of a group $\mathbb{G}$ of prime order $p \in (2^{k-1}, 2^k)$ and a generator $g \in \mathbb{G}$, such that the following holds. For any polynomial-size adversary $\mathcal{A}$ and polynomials m, p there exists a polynomial-size extractor $\mathcal{E}$ such that for any $k \in \mathbb{N}$, any polynomial m, and any auxiliary input $z \in \{0,1\}^{m(k)}$, the advantage of the adversary in the following experiment is bounded by $\text{negl}(k)$.

$\mathsf{EXP}^{\mathcal{G}}_{\mathcal{A},\mathcal{E}}(1^k)$:

- Let $(\mathbb{G}, g) \leftarrow \mathcal{G}_k$. Sample $\alpha_1 \cdots \alpha_{t(k)} \overset{U}{\leftarrow} \mathbb{Z}_p$ and $r_1 \cdots r_{t(k)} \overset{U}{\leftarrow} \mathbb{Z}_p$, where $p = |\mathbb{G}|$. Set $St_0 = (g^{r_1}, \cdots, g^{r_{t(k)}}, g^{\alpha_1 r_1}, \cdots, g^{\alpha_{t(k)} r_{t(k)}}, z)$.
- Run $\mathcal{A}^{\mathcal{O}}(St_0)$ until it halts, replying to the ith oracle query (j_i, c_i, c'_i) as follows:

  $(x_i, St_i) = \mathcal{E}(j_i, c_i, c'_i, St_{i-1})$
  If $j_i \in [t(k)]$, $c'_i = c_i^{\alpha_{j_i}}$, and $c_i \neq g^{r_{j_i} x_i}$, abort and output fail.
  Return x_i to A.

- Output 1.

The advantage of the adversary in the above game, denoted as $\mathsf{ADV}^{\mathcal{G}}_{\mathcal{A},\mathcal{E}}(1^k)$, is the probability that the game outputs fail, that is,

$$\mathsf{ADV}^{\mathcal{G}}_{\mathcal{A},\mathcal{E}}(1^k) = \Pr[\mathsf{EXP}^{\mathcal{G}}_{\mathcal{A},\mathcal{E}}(1^k) = \mathsf{fail}]$$

It is easy to see that assuming the concurrent 1-KEA assumption and that the discrete logs problem is hard in $\mathcal{G}$, the above mentioned construction for sEOWF is in fact an scEOWF.

Acknowledgements

Nir and Ran wish to thank Ben Riva and Omer Paneth for enlightening discussions in the early stages of this research. Eran wishes to thank Shai Halevi for early discussions about using extractable collision resistance as a solution approach and Daniele Micciancio for a discussion of lattice-based Knowledge of Knapsacks assumptions. Huijia wishes to thank Rafael Pass, as well as Kai-Min Chung, for enlightening discussions about extractability assumptions and their applications in general.

References

[1] W. Aiello, S N. Bhatt, R. Ostrovsky, S. Rajagopalan, Fast verification of any remote procedure call: Short witness-indistinguishable one-round proofs for NP, in *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, 2000, pp. 463–474.

[2] J. Alwen, Y. Dodis, D. Wichs. Survey: Leakage resilience and the bounded retrieval model, in *Information Theoretic Security, 4th International Conference, ICITS 2009, Shizuoka, Japan, December 3–6, 2009. Revised Selected Papers*, 2009, pp. 1–18.

[3] M. Abe, S. Fehr. Perfect NIZK with adaptive soundness, in *Proceedings of the 4th theory of cryptography conference*, 2007, pp. 118–136.

[4] B. Applebaum, Y. Ishai, E. Kushilevitz. From secrecy to soundness: efficient verification via secure computation, in *Proceedings of the 37th international colloquium on automata, languages, and programming*, 2010, pp. 152–163.

[5] M. Ajtai. Generating hard instances of lattice problems, in *Proceedings of the 28th annual ACM symposium on the theory of computing*, 1996, pp. 99–108.

[6] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, Proof verification and the hardness of approximation problems, *J. ACM*, **45**(3), (1998) pp. 501–555.

[7] S. Arora, S. Safra, Probabilistic checking of proofs: a new characterization of NP. *J. ACM*, 45(1), (1998) pp. 70–122

[8] M. Backes, M. Barbosa, D. Fiore, R.M. Reischuk. ADSNARK: nearly practical and privacy-preserving proofs on authenticated data, in *Proceedings of the 36th IEEE Symposium on Security and Privacy, S&P '15*, 2015, pp. 271–286.

[9] N. Bitansky, A. Chiesa. Succinct arguments from multi-prover interactive proofs and their efficiency benefits, in *Proceedings of the 32nd annual international cryptology conference, CRYPTO '12*, 2012, pp. 255–272.

[10] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[11] N. Bitansky, R. Canetti, A. Chiesa, E. Tromer, From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. Cryptology ePrint Archive, Report 2011/443, 2011. http://eprint.iacr.org/.

[12] N. Bitansky, R. Canetti, A. Chiesa, E. Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again, in *Proceedings of the 3rd innovations in theoretical computer science conference, ITCS '12*, 2012, pp. 326–349.

[13] N. Bitansky, R. Canetti, A. Chiesa, E. Tromer, Recursive composition and bootstrapping for SNARKs and proof-carrying data, in *Proceedings of the 45th ACM symposium on the theory of computing, STOC '13*, 2013, pp. 111–120.

[14] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, M. Virza. SNARKs for C: verifying program executions succinctly and in zero knowledge, in *Proceedings of the 33rd annual international cryptology conference, CRYPTO '13*, 2013, pp. 90–108.

[15] N. Bitansky, A. Chiesa, Y. Ishai, R. Ostrovsky, O. Paneth, Succinct non-interactive arguments via linear interactive proofs, in *Proceedings of the 10th theory of cryptography conference, TCC '13*, 2013, pp. 315–333.

[16] N. Bitansky, R. Canetti, O. Paneth, A. Rosen, On the existence of extractable one-way functions, in *STOC*, 2014.

[17] E. Ben-Sasson, A. Chiesa, E. Tromer, M. Virza. Scalable zero knowledge via cycles of elliptic curves, in *Proceedings of the 34th annual international cryptology conference, CRYPTO '14*, pp. 276–294, 2014. Extended version at http://eprint.iacr.org/2014/595.

[18] E. Ben-Sasson, A. Chiesa, E. Tromer, M. Virza, Succinct non-interactive zero knowledge for a von Neumann architecture, in *Proceedings of the 23rd USENIX security symposium, security '14*, pp. 781–796, 2014. Extended version at http://eprint.iacr.org/2013/879.

[19] M. Blum, P. Feldman, S. Micali, Non-interactive zero-knowledge and its applications (extended abstract), in *Proceedings of the 20th annual ACM symposium on theory of computing, May 2–4, 1988, Chicago, IL, USA*, 1988, pp. 103–112.

[20] B. Barak, O. Goldreich. Universal arguments and their applications. *SIAM J. Comput.***38**(5), pp. 1661–1694, 2008. Preliminary version appeared in CCC '02.

[21] B. Barak, O. Goldreich, S. Goldwasser, Y. Lindell. Resettably-sound zero-knowledge and its applications, in *FOCS*, 2001, pp. 116–125.

[22] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S.P. Vadhan, K. Yang. On the (im)possibility of obfuscating programs, *SIAM J. Comput.*, **59**(2), 2012.

[23] S. Benabbas, R. Gennaro, Y. Vahlis. Verifiable delegation of computation over large datasets, in *Proceedings of the 31st annual international cryptology conference*, 2011, pp. 111–131.

[24] M. Braverman, A. Hassidim, Y.T. Kalai. Leaky pseudo-entropy functions. In *Proceedings of the 2nd symposium on innovations in computer science*, 2011, pp. 353–366.

[25] Ravi B. Boppana, Johan Håstad, and Stathis Zachos. Does co-NP have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.

[26] M. Blum. Coin flipping by telephone, in *Proceedings of the 18th annual international cryptology conference*, 1981, pp. 11–15.

[27] M. Bellare, A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols, in *Proceedings of the 24th annual international cryptology conference*, 2004, pp. 273–289.

[28] E. Boyle, R. Pass, Limits of extractability assumptions with distributional auxiliary input. Cryptology ePrint Archive, Report 2013/703, 2013. http://eprint.iacr.org/.

[29] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer. On the concrete efficiency of probabilistically-checkable proofs, in *Proceedings of the 45th ACM symposium on the theory of computing, STOC '13*, 2013.

[30] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, S. Vadhan, Short PCPs verifiable in polylogarithmic time, in *Proceedings of the 20th annual IEEE conference on computational complexity*, 2005, pp. 120–134.

[31] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, 38(2):551–607, 2008.

[32] D. Boneh, G. Segev, B. Waters. Targeted malleability: Homomorphic encryption for restricted computations. Cryptology ePrint Archive, Report 2011/311, 2011.

[33] Z. Brakerski, V. Vaikuntanathan, Efficient fully homomorphic encryption from (standard) LWE, in *Proceedings of the 51th annual IEEE symposium on foundations of computer science*, 2011.

[34] R. Canetti, Towards realizing random oracles: hash functions that hide all partial information, in *Proceedings of the 17th annual international cryptology conference*, 1997, pp. 455–469.

[35] R. Canetti, Universally composable security: a new paradigm for cryptographic protocols, in *Proceedings of the 42nd annual IEEE symposium on foundations of computer science*, 2001, pp. 136–145.

[36] R. Canetti, R.R. Dakdouk, Extractable perfectly one-way functions, in *Proceedings of the 35th international colloquium on automata, languages and programming*, 2008, pp. 449–460.

[37] R. Canetti, R.R. Dakdouk, Towards a theory of extractable functions, in *Proceedings of the 6th theory of cryptography conference*, 2009, pp. 595–613.

[38] C. Costello, C. Fournet, J. Howell, M. Kohlweiss, B. Kreuter, M. Naehrig, B. Parno, S. Zahur. Geppetto: versatile verifiable computation, in *Proceedings of the 36th IEEE symposium on security and privacy, S&P '15*, 2015, pp. 253–270.

[39] M. Chase, M. Kohlweiss, A. Lysyanskaya, S. Meiklejohn, Succinct malleable nizks and an application to compact shuffles, in *TCC*, 2013, pp. 100–119.

[40] K.-M. Chung, Y. Kalai, F.-H. Liu, R. Raz. Memory delegation, in *Proceeding of the 31st annual cryptology conference*, 2011, pp. 151–168.

[41] R. Canetti, J. Kilian, E. Petrank, A. Rosen, Black-box concurrent zero-knowledge requires $\tilde{\omega}(\log n)$ rounds, in *STOC '01*, 2001, pp. 570–579.

[42] K.-M. Chung, Y. Kalai, S. Vadhan, Improved delegation of computation using fully homomorphic encryption, in *Proceedings of the 30th annual international cryptology conference*, 2010, pp. 483–501.

[43] C. Cachin, S. Micali, M. Stadler, Computationally private information retrieval with polylogarithmic communication, in *Proceedings of the international conference on the theory and application of cryptographic techniques*, 1999, pp. 402–414.

[44] R. Canetti, B. Riva, G.N. Rothblum, Two 1-round protocols for delegation of computation. Cryptology ePrint Archive, Report 2011/518, 2011.

[45] A. Chiesa, E. Tromer, Proof-carrying data and hearsay arguments from signature cards, in *Proceedings of the 1st symposium on innovations in computer science*, 2010, pp. 310–331.

[46] G. Cormode, J. Thaler, K. Yi, Verifying computations with streaming interactive proofs. Technical report, 2010. ECCC TR10-159.

[47] R.R. Dakdouk, *Theory and application of extractable functions*. Ph.D. thesis, Yale University, Computer Science Department, December 2009.

[48] I. Damgård, Towards practical public key systems secure against chosen ciphertext attacks, in *Proceedings of the 11th annual international cryptology conference*, 1992, pp. 445–456.

[49] G. Di Crescenzo, H. Lipmaa, Succinct NP proofs from an extractability assumption, in *Proceedings of the 4th conference on computability in Europe*, 2008, pp. 175–185.

[50] A.W. Dent, The hardness of the DHK problem in the generic group model. Cryptology ePrint Archive, Report 2006/156, 2006.

[51] G. Danezis, C. Fournet, J. Groth, M. Kohlweiss, Square span programs with applications to succinct NIZK arguments, in *Proceedings of the 20th international conference on the theory and application of cryptology and information security, ASIACRYPT '14*, 2014, pp. 532–550.

[52] I. Damgård, S. Faust, C. Hazay, Secure two-party computation with low communication. Cryptology ePrint Archive, Report 2011/508, 2011.

[53] A. Dent, S. Galbraith, Hidden pairings and trapdoor DDH groups, in F. Hess, S. Pauli, M. Pohst, editors, *Algorithmic number theory, vol. 4076 of lecture notes in computer science*, 2006, pp. 436–451.

[54] Y. Deng, V. Goyal, A. Sahai, Resolving the simultaneous resettability conjecture and a new non-black-box simulation strategy, in *FOCS*, 2009, pp. 251–260.

[55] S. Dziembowski, P. Krzysztof, Leakage-resilient cryptography, in *Proceedings of the 49th annual IEEE symposium on foundations of computer science*, 2008, pp. 293–302.

[56] C. Dwork, M. Langberg, M. Naor, K. Nissim, O. Reingold. Succinct NP proofs and spooky interactions, December 2004. Available at www.openu.ac.il/home/mikel/papers/spooky.ps.

[57] C. Dwork, M. Naor, Zaps and their applications, in *FOCS*, 2000, pp. 283–293

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

[59] Y. Dodis, T. Ristenpart, T. Shrimpton, Salvaging merkle-damgård for practical applications, in *Proceedings of the 28th annual international conference on the theory and applications of cryptographic techniques*, 2009, pp. 371–388.

[60] B. Ederov, *Merkle tree traversal techniques*. Ph.D. thesis, Darmstadt University of Technology, Department of Computer Science, April 2007

[61] P. Fauzi, H. Lipmaa, B. Zhang, Efficient modular NIZK arguments from shift and product, in *Proceedings of the 12th international conference on cryptology and network security, CANS '13*, 2013, pp. 92–121.

[62] A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, in *Proceedings of the 6th annual international cryptology conference*, 1987, pp. 186–194

[63] C. Gentry, Fully homomorphic encryption using ideal lattices, in *Proceedings of the 41st annual ACM symposium on theory of computing*, 2009, pp. 169–178

[64] O. Goldreich, S. Goldwasser, S. Halevi, Collision-free hashing from lattice problems. Technical report, 1996. ECCC TR95-042

[65] R. Gennaro, C. Gentry, B. Parno. Non-interactive verifiable computing: outsourcing computation to untrusted workers, in *Proceedings of the 30th annual international cryptology conference*, 2010, pp. 465–482.

[66] R. Gennaro, C. Gentry, B. Parno, M. Raykova, Quadratic span programs and succinct NIZKs without PCPs, in *Proceedings of the 32nd annual international conference on theory and application of cryptographic techniques, EUROCRYPT '13*, 2013, pp. 626–645

[67] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Information Processing Letters*, 67(4):205–214, 1998.

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

[69] S. Goldwasser, Y.T. Kalai, G.N. Rothblum, Delegating computation: interactive proofs for Muggles, in *Proceedings of the 40th annual ACM symposium on theory of computing*, 2008, pp. 113–122.

[70] R. Gennaro, H. Krawczyk, T. Rabin, Okamoto–Tanaka revisited: fully authenticated Diffie–Hellman with minimal overhead, in *Proceedings of the 8th international conference on applied cryptography and network security*, 2010, pp. 309–328

[71] S. Goldwasser, H. Lin, A. Rubinstein, Delegation of computation without rejection problem from designated verifier CS-proofs. Cryptology ePrint Archive, Report 2011/456, 2011.

[72] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[73] S. Goldwasser, S. Micali, C. Rackoff, The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, **18**(1), pp. 186–208, 1989. Preliminary version appeared in STOC '85

[74] 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.

[75] N. Gama, P.Q. Nguyen. Predicting lattice reduction, in *Proceedings of the 27th annual international conference on the theory and applications of cryptographic techniques*, 2008, pp. 31–51

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

[77] O. Goldreich, *The foundations of cryptography—volume 1, basic techniques*. Cambridge University Press, 2001

[78] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, New York, NY, USA, 2001.

[79] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, New York, NY, USA, 2004.

[80] C. Gentry, Z. Ramzan. Single-database private information retrieval with constant communication rate, in *Proceedings of the 32nd international colloquium on automata, languages and programming*, 2005, pp. 803–815

[81] J. Groth. Short pairing-based non-interactive zero-knowledge arguments, in *Proceedings of the 16th international conference on the theory and application of cryptology and information security*, 2010, pp. 321–340

[82] D. Gupta, A. Sahai, On constant-round concurrent zero-knowledge from a knowledge assumption. Cryptology ePrint Archive, Report 2012/572, 2012. http://eprint.iacr.org/

[83] Oded Goldreich, Salil Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1/2):1–53, 2002.

[84] C. Gentry, D. Wichs, Separating succinct non-interactive arguments from all falsifiable assumptions, in *Proceedings of the 43rd annual ACM symposium on theory of computing*, 2011, pp. 99–108.

[85] I. Haitner, D. Harnik, O. Reingold. Efficient pseudorandom generators from exponentially hard one-way functions, in *Proceedings of the 33rd international colloquium on automata, languages and programming*, 2006, pp. 228–239

[86] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[87] S. Hada, T. Tanaka, On the existence of 3-round zero-knowledge protocols, in *Proceedings of the 18th annual international cryptology conference*, 1998, pp. 408–423

[88] Y. Ishai, E. Kushilevitz, R. Ostrovsky, M. Prabhakaran, A. Sahai, Efficient non-interactive secure computation, in *Proceedings of the 30th annual international conference on the theory and applications of cryptographic techniques*, 2011, pp. 406–425

[89] J. Kilian. A note on efficient zero-knowledge proofs and arguments, in *Proceedings of the 24th annual ACM symposium on theory of computing*, 1992, pp. 723–732

[90] E. Kushilevitz, R. Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval, in *38th annual symposium on foundations of computer science, FOCS '97, Miami Beach, Florida, USA, October 19–22, 1997*, 1997, pp. 364–373

[91] Y.T. Kalai, O. Paneth. Delegating RAM computations. Cryptology ePrint Archive, Report 2015/957, 2015

[92] A.E. Kosba, D. Papadopoulos, C. Papamanthou, M.F. Sayed, E. Shi, N. Triandopoulos, TRUESET: Faster verifiable set computations, in *Proceedings of the 23rd USENIX security symposium*, USENIX Security '14, 2014, pp 765–780

[93] Y.T. Kalai, R. Raz, Succinct non-interactive zero-knowledge proofs with preprocessing for LOGSNP, in *Proceedings of the 47th annual IEEE symposium on foundations of computer science*, 2006, pp. 355–366

[94] Y.T. Kalai, R. Raz, Probabilistically checkable arguments, in *Proceedings of the 29th annual international cryptology conference*, 2009, pp. 143–159

[95] Y. Kalai, R. Raz, R. Rothblum, Delegation for bounded space, in *Proceedings of the 45th ACM symposium on the theory of computing*, STOC '13, 2013, pp. 565–574

[96] Y.T. Kalai, R. Raz, R.D. Rothblum, How to delegate computations: the power of no-signaling proofs, in *Proceedings of the 46th annual ACM symposium on theory of computing, STOC '14*, 2014, pp. 485–494

[97] H. Lipmaa, Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments, in *Proceedings of the 9th theory of cryptography conference on theory of cryptography, TCC '12*, 2012, pp. 169–189

[98] H. Lipmaa, Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes, in *Proceedings of the 19th international conference on the theory and application of cryptology and information security, ASIACRYPT '13*, 2013, pp. 41–60

[99] H. Lipmaa, Efficient NIZK arguments via parallel verification of Beneš networks, in *Proceedings of the 9th international conference on security and cryptography for networks, SCN '14*, 2014, pp. 416–434

[100] V. Lyubashevsky, D. Micciancio, On bounded distance decoding, unique shortest vectors, and the minimum distance problem, in *Proceedings of the 29th annual international cryptology conference*, 2009, pp. 577–594

[101] D. Lapidot, A. Shamir, Publicly verifiable non-interactive zero-knowledge proofs, in *CRYPTO*, 1990, pp. 353–365

[102] R. Merkle, *Secrecy, authentication and public key systems*. Ph.D. thesis, Stanford University, Department of Electrical Engineering, 1979

[103] R.C. Merkle, A certified digital signature, in *Proceedings of the 9th annual international cryptology conference*, 1989, pp. 218–238

[104] S. Micali, Computationally sound proofs. *SIAM J. Comput.*, **30**(4), pp. 1253–1298, 2000. Preliminary version appeared in FOCS '94.

[105] Thilo Mie. Polylogarithmic two-round argument systems. *Journal of Mathematical Cryptology*, 2(4):343–363, 2008.

[106] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37:267–302, April 2007.

[107] M. Naor, On cryptographic assumptions and challenges, in *Proceedings of the 23rd annual international cryptology conference*, 2003, pp. 96–109

[108] V.I. Nechaev, Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, **55**, (1994), pp. 165–172

[109] M. Naor, K. Nissim, Communication preserving protocols for secure function evaluation, in *Proceedings of the 33rd annual ACM symposium on theory of computing*, 2001, pp. 590–599

[110] Eiji Okamoto and Kazue Tanaka. Key distribution system based on identification information. *Selected Areas in Communications, IEEE Journal on*, 7(4):481–485, May 1989.

[111] R. Pass, Limits of provable security from standard assumptions, in *STOC*, 2011, pp. 109–118

[112] B. Parno, C. Gentry, J. Howell, M. Raykova, Pinocchio: nearly practical verifiable computation, in *Proceedings of the 34th IEEE symposium on security and privacy, Oakland '13*, 2013, pp. 238–252

[113] O. Paneth, G.N. Rothblum. Publicly verifiable non-interactive arguments for delegating computation. Cryptology ePrint Archive, Report 2014/981, 2014

[114] C. Papamanthou, E. Shi, R. Tamassia, K. Yi, Streaming authenticated data structures, in *Proceedings of the international conference on the theory and application of cryptographic techniques*, 2013, pp. 353–370

[115] C. Papamanthou, R. Tamassia, N. Triandopoulos, Optimal verification of operations on dynamic sets, in *Proceeding of the 31st annual cryptology conference*, 2011, pp. 91–110

[116] M. Prabhakaran, R. Xue. Statistically hiding sets, in *Proceedings of the cryptographers' track at the RSA conference 2009*, 2009, pp. 100–116

[117] O. Regev, New lattice based cryptographic constructions, in *Proceedings of the 35th annual ACM symposium on theory of computing*, 2003, pp. 407–416

[118] Oded Regev. New lattice-based cryptographic constructions. *Journal of the ACM*, 51(6):899–942, 2004.

[119] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, in *Proceedings of the 37th annual ACM symposium on theory of computing*, 2005, pp. 84–93

[120] Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:204–213, 1994.

[121] O. Reingold, L. Trevisan, M. Tulsiani, S.P. Vadhan, Dense subsets of pseudorandom sets, in *Proceedings of the 49th annual IEEE symposium on foundations of computer science*, 2008, pp. 76–85

[122] Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[123] V. Shoup, Lower bounds for discrete logarithms and related problems, in *Proceedings of the international conference on the theory and application of cryptographic techniques*, 1997, pp. 256–266

[124] P. Valiant, Incrementally verifiable computation or proofs of knowledge imply time/space efficiency, in *Proceedings of the 5th theory of cryptography conference*, 2008, pp. 1–18

[125] H. Wee, On round-efficient argument systems, in *Proceedings of the 32nd international colloquium on automata, languages and programming*, 2005, pp. 140–152

[126] R.S. Wahby, S. Setty, Z. Ren, A.J. Blumberg, M. Walfish, Efficient RAM and control flow in verifiable outsourced computation, in *Proceedings of the 22nd network and distributed system security symposium, NDSS '15*, 2015

[127] Y. Zhang, C. Papamanthou, J. Katz, Alitheia: towards practical verifiable graph processing, in *Proceedings of the 21st ACM conference on computer and communications security, CCS '14*, 2014, pp. 856–867