Optimal Succinct Arguments via Hidden Order Groups

Russell W. F. Lai and Giulio Malavolta

Friedrich-Alexander-Universität Erlangen-Nürnberg

Abstract. An argument of knowledge allows a prover to convince a verifier of the validity of certain statements. We construct succinct arguments of knowledge with an optimal communication complexity of $O(\lambda)$ bits in the standard model, thereby resolving an open problem posed by Kilian (CRYPTO' 95), assuming that the strong root problem is hard over groups of unknown order. Our protocol can be generically transformed into a zero-knowledge succinct non-interactive argument of knowledge (zk-SNARK) with proofs of size $O(\lambda)$ bits, with efficient trusted setup in the random oracle model. These results can be instantiated under the strong RSA assumption. For groups that admit a public-coin setup, the transformation yields a zk-SNARK without setup. A plausible candidate family of such groups is class groups of imaginary quadratic orders. Existing zk-SNARKs with optimal proof size require inefficient trusted setup and use bilinear maps.
Our main technical tool is a generalization of vector commitments called subvector commitments. The latter allows one to open a commitment of a vector at a set of positions, where the opening size is independent of the size of the set. Apart from the new application in constructing succinct arguments, subvector commitments generally serve as a more efficient replacement of vector commitments in all applications where the prover needs to decommit at more than one position.

1 Introduction

An argument system allows a prover, with a witness w, to convince a verifier that a certain statement x is in an NP language $\mathcal{L}$. In contrast with proof systems, argument systems are only required to be computationally sound. Due to this relaxation, it is possible that the interaction between the prover and the verifier is succinct, *i.e.*, the communication complexity is bounded by some polynomial $\mathsf{poly}(\lambda)$ in the security parameter and is independent of the size of w. One can also require an argument system to be zero-knowledge, meaning that the communication transcript can be efficiently simulated without knowing the witness. An argument is of knowledge if for any successful prover there exists an extractor that can recover w. Finally, we say that an argument is non-interactive if it consists of a single message from the prover to the verifier.

1.1 The Quest of Constructing Ever Shorter Arguments

Recently, much progress has been made both in theory and practice to construct zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARK) for general NP languages. We distinguish between zk-SNARKs in the plain / setup-free model and the pre-processing model. In the plain model, the prover and the verifier do not

Scheme	Proof Size (bits)	CRS Size (bits)	Assumptions
[24]	$O(\lambda)$	$O(s)$	Knowledge of Exponent
[35]	$O(\lambda)$	$O(s)$	Generic Group Model
This work	$O(\lambda)$	$O(\lambda)$	Random Oracle, Strong RSA
[46]	$O(\lambda^2 \log \lvert w \rvert)$	–	Random Oracle
[19]	$O(\lambda \log n)$	–	Random Oracle, Discrete Logarithm
This work	$O(\lambda)$	–	Random Oracle, Strong Root w/o Trusted Setup

Table 1: Comparison of zk-SNARKs. λ: security parameter, w: witness, n: number of multiplication gates in NP verification circuit, s: size of NP verification circuit.

share any input other than the statement x to be proven. In the pre-processing model, they share a common reference string, generated by a trusted third party, which may depend on the language $\mathcal{L}$ and the statement x. In general, existing zk-SNARKs in the pre-processing model are asymptotically more efficient (at least in terms of the computational complexity of verification) than those in the plain model. This reflects the intuition that pushing the majority of the verifier workload to the offline pre-processing phase reduces its workload in the online phase. On the other hand, in some applications such as cryptocurrencies it is crucial to avoid a trusted setup. With the current landscape, one is forced to trade optimal-size proofs with the absence of a trusted setup. In Table 1, we highlight some schemes in each model with the shortest proof size.

In the pre-processing model, there exists plenty of zk-SNARKs constructed from linear interactive proofs (LIP) and pairings in the standard model [31]. The scheme with the shortest proofs is due to Danezis *et al.* [24], where a proof consists of 4 group elements. Groth [35] proposed a scheme in the generic group model [49] with only 3 group elements, and showed that proofs constructed from LIP must consist of at least 2 group elements. Assuming that each group element can be represented by $O(\lambda)$ bits, then all of these constructions have proof size $O(\lambda)$ in bits, which is optimal up to a constant multiplicative factor.

While it is known that setup-free non-interactive arguments for NP do not exists in the standard model [13], one can circumvent this impossibility by assuming the existence of a random oracle [6]. For a soundness parameter k, a proof in the scheme by Micali [46] consists of a λ-bit Merkle-tree commitment of a probabilistic checkable proof (PCP) string, $O(k)$ bits of the PCP string, and $O(k)$ openings of the commitment, each of size $O(\lambda \log |w|)$ bits. Assuming $k = O(\lambda)$, a proof consists of $O(\lambda^2 \log |w|)$ bits. Since the size of the witness can be bounded by $|w| \leq 2^\lambda$, the proof size is bounded by $O(\lambda^3)$. In Bulletproof [19], which improves upon Bootle *et al.* [14], a proof consists of $O(\log n)$ group elements and integers, where n is the number of multiplication gates in the arithmetic circuit representation of the verification algorithm of $\mathcal{L}$. Assuming that each of the group elements and integers can be represented by λ bits, the proof size is $O(\lambda \log n) \leq O(\lambda^2)$ bits, where the inequality is due to $n \leq 2^\lambda$.

1.2 Our Results

The state of the art raises the question of both theoretical and practical importance:

Can we construct setup-free zk-SNARKs with $O(\lambda)$-bit (optimal) proofs?

In this work we answer this question positively. Throughout the process, we also obtain similar results in other settings which might be of independent interest. We summarize our main theorems below, with detailed explanation deferred to Section 7.

The core of our results is a construction of 4-move argument systems for NP, assuming the existence of groups[1] where (a weaker form of) the strong root problem is hard. When instantiated with the strong RSA assumption, we obtain a system with communication complexity $O(\lambda)$. To the best of our knowledge, prior to our work the most succinct argument system was due to Kilian [43], with communication complexity $O(\lambda \log \lambda)$. In the same work, Kilian posed the problem of constructing a system with optimal $O(\lambda)$ communication, which was open for over 20 years.

Theorem 1. *Assuming the strong RSA problem is hard, there exist 4-move argument systems for NP with communication complexity $O(\lambda)$.*

Since the system obtained above is public-coin except for the first message, sent from the verifier to the prover, it immediately yields a pre-processing SNARK for NP in the random oracle model by applying the Fiat-Shamir [28] transformation. A zk-SNARK can then be obtained generically using non-interactive zero-knowledge proof of knowledge (NIZKPoK), which can be constructed from the (strong) RSA assumption in the common random string model [7].

Theorem 2. *Assuming the strong RSA problem is hard, there exist pre-processing zk-SNARKs for NP with CRS size $O(\lambda)$ and proof size $O(\lambda)$ in the random oracle model.*

Returning to the original question of constructing a setup-free zk-SNARK, we observe that the above zk-SNARK requires a setup due to the fact that the first message of the interactive argument system is not public-coin. To make this first message public-coin, we need to assume further that the strong root problem is still hard even if it is defined by a group element sampled with public coin. A candidate family of groups is that of class groups of imaginary quadratic orders, denoted $Cl(\Delta)$.

Theorem 3. *Assuming the strong root problem (without trusted setup) is hard over $Cl(\Delta)$, and NIZKPoK exists in the common random string model, there exists setup-free zk-SNARKs for NP with proof size $O(\lambda)$ in the random oracle model.*

1.3 Subvector Commitments

Our construction relies on a generalization of vector commitments (VC) called subvector commitments (SVC), which we introduce below. A VC scheme [20] is a commitment scheme which allows to commit a vector of messages, such that the committer can later open the commitment at any position i of the vector, *i.e.*, reveal a message and show that it equals to the i-th committed message. A VC scheme is required to be position binding, meaning that no efficient algorithm can open a commitment at some position i to two distinct messages $m_i \neq m_i'$. Note that the position binding property is only required

[1] Our formal results are presented over modules over Euclidean rings, which generalize groups.

to hold if the public parameters are generated honestly using *private randomness*. A VC scheme is also required to be dynamic, meaning that the committer can efficiently update a commitment and correspondingly its openings such that the new commitment contains a new vector of messages. Catalano and Fiore [20] constructed two VC schemes based on the computational Diffie-Hellman assumption over bilinear groups and the RSA assumption, respectively. In both schemes, a commitment and an opening both consist of a single group element. Furthermore, the scheme based on the RSA assumption has public parameters whose size is independent of the length of the vectors to be committed.

We generalize the notion of VC into that of subvector commitments (SVC). Given a vector $\boldsymbol{m}$ of length q and an *ordered* index set $I \subseteq [q]$, we define the I-subvector of $\boldsymbol{m}$ as the vector formed by collecting the i-th component of $\boldsymbol{m}$ for all $i \in I$. The main difference between VC and SVC is that, in the latter, the committer can open a commitment at all positions in I simultaneously. It is important that an opening has size sublinear in (or even independent of) $|I|$, otherwise it is not more efficient than opening a VC at $|I|$ positions separately. The committer can also update a commitment and its openings such that an I-subvector of the committed vector is replaced with a new one. In additional to the functional differences, we define position binding without trusted setup. This property says that, even if the public parameters are sampled using *public coins*, no efficient algorithm can open a commitment at some index sets I and J to some I-subvector $\boldsymbol{m}_I$ and $\boldsymbol{m}'_J$ respectively, such that there exists $i \in I \cap J$ with $m_i \neq m'_i$.

1.4 Construction Overview

With the background knowledge of (S)VC, we overview our techniques for obtaining the main result – the construction of a setup-free zk-SNARK with $O(\lambda)$-bit proofs. Our starting point is the succinct argument system of Micali [46], later proven to be an argument of knowledge by Valiant [50]. The scheme can be described as a Σ protocol between a prover and a verifier, where the latter uses public coins. First, the prover computes a PCP of the statement and sends a Merkle-tree commitment of the PCP string to the verifier. The latter then sends k independent randomness, each of which determines a set of positions queried by a PCP verifier. The prover responds by opening all queried positions of the PCP string. Since the verifier is public-coin, the protocol can then be turned non-interactive using the Fiat-Shamir [27] transformation.

Our initial idea is to replace the Merkle-tree commitment used in Micali's scheme with a VC, so that a proof now consists of a commitment and k openings. This removes the $\log |w|$ factor in the proof size and yields a zk-SNARK with proof size $O(k\lambda)$ in the pre-processing model (as vector commitments assume a trusted setup). In order to remove the k factor (and to avoid pre-processing), we replace the VC with an SVC (without trusted setup), with the size of an opening to a set of k positions independent of k. This turns the multiplicative factor k into an additive one, *i.e.*, the proof size is now $O(\lambda + k) = O(\lambda)$ bits, assuming $k = O(\lambda)$. The resulting protocol is a succinct 4-move zero-knowledge argument of knowledge with communication complexity $O(\lambda)$ in bits. The Fiat-Shamir transformation is then applied to make the argument non-interactive.

It remains to construct the required SVC scheme. To begin, we upgrade the RSA-based VC scheme by Catalano and Fiore [20] into an SVC scheme *with trusted setup*. This intermediate SVC is already sufficient to instantiate our *interactive* argument and

our pre-processing zk-SNARK, by just letting the verifier sampling the public parameters. However a public-coin SVC scheme, needed to make the zk-SNARK setup-free, requires hidden order groups with a public-coin generation. To capture the minimal mathematical structure required, we generalize our construction to work over modules defined over Euclidean rings, following the framework of Lipmaa [45]. We then prove that the SVC is position binding (without trusted setup), assuming the existence of "strong distinct-prime-product root modules without trusted setup" (Definition 6). This assumption is qualitatively weaker than the assumption on the existence of strong root modules without trusted setup, which in turn is a natural generalization of the strong RSA assumption. Compared to the original proof of VC position binding, our proof requires a more elaborate manipulation of the exponents and assumes the hardness of (a variant of) the strong root problem.

In the setup-free setting, our construction can no longer be instantiated with the strong RSA assumption, since the problem is easy if the RSA modulus is sampled using public randomness. Instead, we suggest two candidate families of groups of unknown order, where the strong (distinct-prime-product) root problem is believed to be hard even if the group is sampled using public coin. The first candidate, which we recommend for efficiency (see below), is the family of class groups of imaginary quadratic orders, each defined by a single integer Δ, called the discriminant, with very little structure. The second candidate, which we include for completeness[2], is the family of the groups $\mathbb{Z}_N^*$ where N are the so called "generalized RSA-moduli with unknown complete factorization (RSA-UFOs)" [48], which are essentially randomly sampled integers. Both of these families were used to construct, among other primitives, accumulators without trusted setup [45,48]. For a more detailed discussion of these candidates, we refer to Section 6.

In terms of concrete efficiency, we emphasize that when instantiated with RSA groups or class groups of imaginary quadratic orders, the constant hidden in the $O(\lambda)$ proof size is reasonably small. With a class group defined by a d-bit discriminant Δ where $d = O(\lambda)$ or an RSA group with a d-bit modulus N, a soundness parameter k, and a PCP where the verifier tests $h = O(1)$ bits of the PCP string, a proof consists of at most $2d + hk$ bits. In terms of verifier efficiency, we note that our verifier performs $\log(hk)$ group operations, 1 exponentiation, and k PCP verifications. Finally, we remark that a class group defined by a 1000-bit discriminant Δ provides roughly the same level of security as RSA with a 1536-bit modulus [36], $k = 100$ is way more than what is typically considered necessary for soundness, and there exists PCPs where the verifier tests only $h = 3$ bits of the PCP string [37].

1.5 Other Applications

Catalano and Fiore [20] suggested a list of applications of VC, including verifiable databases with efficient updates, updatable zero-knowledge elementary databases, and universal dynamic accumulators. In all of these applications, one can gain efficiency by replacing the VC scheme with an SVC scheme which allows for batch opening and

[2] As humongous RSA-UFOs (in the order of 30000-bit) are required for security, cryptosystems based on these groups are not practical.

updating. When instantiated with our construction of SVC, one can further avoid the trusted setup, which is especially beneficial to database applications as trusted third parties are no longer required.

1.6 Related Work

Succinct arguments were introduced by Kilian [42,43] and later improved, in terms of round complexity, by Lipmaa and Di Crescenzo [26]. Non-interactive arguments, or computationally sound proofs, were first proposed by Micali [46]. These early approaches rely on PCP and have been recently extended [8] to handle interactive oracle proofs [11] (also known as probabilistic checkable interactive proofs [47]), in favor of a more efficient prover (but with the same asymptotics). A recent manuscript by Ben-Sasson *et al.* [9] improves the concrete efficiency of interactive oracle proofs. Ishai, Kushilevitz, and Ostrovsky [40] first observed that linear PCP can be combined with a linearly homomorphic encryption to construct more efficient arguments, with pre-processing. Afterwards, Groth [34] and Lipmaa [44] upgraded this approach to non-interactive proofs. Gennaro *et al.* [31] presented a very elegant linear PCP that gave rise to a large body of work to improve the practical efficiency of non-interactive arguments [4,5,10,12,21,22,29,25]. All of these constructions assume a highly structured and honestly generated common reference string (of size proportional to the circuit to be evaluated) and rely on some variant of the knowledge of exponent assumption. Recently, Ames *et al.* [1] proposed an argument based on the MPC-in-the-head [41] paradigm to prove satisfiability of a circuit C with proofs of size $O(\lambda\sqrt{|C|})$. Zhang *et al.* [52] showed how to combine interactive proofs and verifiable polynomial delegation schemes to construct succinct interactive arguments. The scheme requires a trusted pre-processing and the communication complexity is $O(\lambda \log |w|)$. A similar result by Whaby *et al.* [51] introduces a prover-efficient construction with proofs of size $O(\lambda\sqrt{|w|})$.

2 Preliminaries

Throughout this work we denote by $\lambda \in \mathbb{N}$ the security parameter, and by $\mathsf{poly}(\lambda)$ and $\mathsf{negl}(\lambda)$ the sets of polynomials and negligible functions in λ, respectively. We say that a Turing machine is probabilistic polynomial time (PPT) if its running time is bounded by some polynomial function $\mathsf{poly}(\lambda)$. An interactive protocol Π between two machines A and B is referred to as $(A, B)_\Pi$. Given a set S, we denote sampling a random element from S as $s \leftarrow\!\!\$\, S$ and the output of an algorithm A on input x is written as $z \leftarrow A(x)$. Let $\ell \in \mathbb{N}$, the set $[\ell]$ is defined as $[\ell] := \{1, \ldots, \ell\}$. Let $S_i = (s_{i,1}, \ldots, s_{i,\ell_i})$ be *ordered* sets for $i \in [k]$. Their concatenation is denoted as $S_1 \| \ldots \| S_k := (s_{1,1}, \ldots, s_{k,\ell_k})$. Let $S = (s_1, \ldots, s_\ell)$ be an *ordered* set. We denote by $\mathsf{Unique}(S)$ the set of all *unique* elements in S with their relative order preserved, *i.e.*, if $s_i = s_j$ and $i < j$, then s_j is dropped from S.

2.1 Hoeffding's Inequality

In the following we recall a useful inequality by Hoeffding. Let $X_1, \ldots, X_n$ be independent random variables bounded by the interval $[0, 1]$ and let $\overline{X} := \frac{X_1 + \ldots + X_n}{n}$, then it

holds that

$$\mathsf{Pr}\left[\overline{X} - E\left[\overline{X}\right] \geq d\right] \leq e^{-2nd^2}.$$

where $0 < d < \overline{X} - E\left[\overline{X}\right]$.

2.2 Pseudo-Random Generators

A pseudorandom generator [39] allows one to stretch random strings into new random looking ones.

Definition 1 (Pseudo-Random Generator). *A function* $\mathsf{PRG} : \{0,1\}^n \to \{0,1\}^m$ *is a pseudo-random generator if* $m > n$ *and for all PPT adversaries* $\mathcal{A}$ *the following ensembles are computationally indistinguishable*

$$\{\mathsf{PRG}(s)\}_{s \leftarrow\$ \{0,1\}^n} \approx \{r\}_{r \leftarrow\$ \{0,1\}^m}.$$

2.3 Arguments of Knowledge

Let $\mathcal{R} : \{0,1\} \times \{0,1\} \to \{0,1\}$ be an NP-relation with corresponding NP-language $\mathcal{L} := \{x : \exists w \; s.t. \; \mathcal{R}(x,w) = 1\}$. We define arguments of knowledge [15] for interactive Turing machines [32].

Definition 2 (Arguments of knowledge). *An interactive protocol* $\Pi = (\mathcal{P}, \mathcal{V})_\Pi$ *is a (succinct) argument of knowledge for* $\mathcal{R}$ *if the following conditions hold.*

(Completeness) If $\mathcal{R}(x,w) = 1$ *then*

$$\mathsf{Pr}\left[(\mathcal{P}(x,w), \mathcal{V}(x))_\Pi = 1\right] = 1.$$

(Argument of Knowledge) For all PPT adversary $\mathcal{A}$*, there exists a PPT machine* $\mathcal{E}_\mathcal{A}$*, such that for all* $x, z \in \{0,1\}^*$*, if there exists a constant* c *such that* $\mathsf{Pr}\left[(\mathcal{A}(x,z), \mathcal{V}(x))_\Pi = 1\right] \geq \frac{1}{\lambda^c}$*, then there exists a constant* d *such that*

$$\mathsf{Pr}\left[\mathcal{R}(x,w) = 1 \middle| w \leftarrow \mathcal{E}_{\mathcal{A}(x,z)}(x)\right] \geq \frac{1}{\lambda^d}.$$

(Succinctness) The communication between $\mathcal{P}$ *and* $\mathcal{V}$ *is at most* $\mathsf{poly}(\lambda)$.

An argument of knowledge satisfies the notion of zero-knowledge if the interaction between the prover and the verifier reveals nothing but the validity of the statement.

Definition 3 (Zero Knowledge). *An interactive protocol* $\Pi = (\mathcal{P}, \mathcal{V})_\Pi$ *is computationally (statistically, resp.) zero-knowledge if for all PPT* $\mathcal{V}^*$ *there exists a PPT algorithm* $\mathcal{S}$ *such that the following ensembles are computationally (statistically, resp.) indistinguishable*

$$\{(\mathcal{P}(x,w), \mathcal{V}^*(x))_\Pi\}_{\lambda \in \mathbb{N}, x \in \mathcal{L}, w \; s.t. \; \mathcal{R}(x,w)=1} \approx \{\mathcal{S}(x)\}_{\lambda \in \mathbb{N}, x \in \mathcal{L}}.$$

2.4 Witness-Extractable PCP

One of the principal tools in the construction of arguments of knowledge is probabilistic checkable proofs (PCP) [2]. The PCP theorem shows that any witness w (of lenght n) for an NP-statement can be encoded into a PCP of length $n(\log n)^{O(1)}$ such that it is sufficient to probabilistically test $O(1)$ bits of the encoded witness.

Definition 4 (Probabilistically Checkable Proofs). *A pair of machines $(\mathcal{P}_{PCP}, \mathcal{V}_{PCP})$ is a PCP for an NP-relation $\mathcal{R}$ if the following conditions hold.*

(Completeness) If $\mathcal{R}(x, w) = 1$, then

$$\mathsf{Pr}\left[\mathcal{V}_{PCP}(x)^{\boldsymbol{\pi}} = 1 | \boldsymbol{\pi} \leftarrow \mathcal{P}_{PCP}(x, w)\right] = 1.$$

(Soundness) For all $x \notin \mathcal{L}$ and $\boldsymbol{\pi} \in \{0,1\}^*$,

$$\mathsf{Pr}\left[\mathcal{V}_{PCP}(x)^{\boldsymbol{\pi}} = 1\right] < \frac{1}{3}.$$

It is well known that one can diminish the soundness error to a negligible function by parallel repetition. Let ℓ be a constant, for notational convenience we write $\mathcal{Q}(\mathcal{V}_{\text{PCP}}(x)^{\boldsymbol{\pi}})$ the *ordered* set of queries $(q_1, \ldots, q_\ell)$ made by $\mathcal{V}_{\text{PCP}}(x)^{\boldsymbol{\pi}}$, where $q_i \in [|\boldsymbol{\pi}|]$ for all $i \in [\ell]$. We additionally require that the witness can be efficiently recovered from the encoded string $\boldsymbol{\pi}$ [50].

Definition 5 (Witness-Extractability). *A PCP is witness-extractable if there exists a PPT algorithm $\mathcal{E}_{PCP}$ and a constant $\gamma \in (0,1)$ such that, given any strings x and $\boldsymbol{\pi}$ such that $\mathsf{Pr}\left[\mathcal{V}_{PCP}(x)^{\boldsymbol{\pi}} = 1\right] \geq 1 - \gamma$, extracts an NP witness w for x.*

3 Definitions

To capture the minimal mathematical structure required for our constructions, we follow the module-based cryptography framework of Lipmaa [45].

3.1 Algebraic Background

A (left) R-module R_D over the ring R (with identity) consists of an Abelian group $(D, +)$ and an operation $\circ : R \times D \to D$, denoted $r \circ A$ for $r \in R$ and $A \in D$, such that for all $r, s \in R$ and $A, B \in D$, we have i) $r \circ (A + B) = r \circ A + r \circ B$, ii) $(r + s) \circ A = r \circ A + s \circ A$, iii) $(r \cdot s)A = r \circ (s \circ A)$, and iv) $1_R \circ r = r$, where 1_R is the multiplicative identity of R. Let $S = (s_1, \ldots, s_q) \subseteq \mathbb{N}$ be an ordered set, and $\boldsymbol{r} = (r_{s_1}, \ldots, r_{s_q}) \in R^q$ and $\boldsymbol{A} = (A_{s_1}, \ldots, A_{s_q}) \in D^q$ be vectors of ring and group elements respectively. For notational convenience, we denote $\sum_{i \in S} r_i \circ A_i$ by $\langle \boldsymbol{r}, \boldsymbol{A} \rangle$.

A commutative ring R with identity is called an *integral domain* if for all $r, s \in R$, $rs = 0_R$ implies $r = 0_R$ or $s = 0_R$, where 0_R is the additive identity of R. A ring R is *Euclidean* if it is an integral domain and there exists a function $\deg : R \to \mathbb{Z}^+$, called the Euclidean degree, such that i) if $r, s \in R$, then there exists $q, k \in R$ such that $r = qs + k$ with either $k = 0_R$, $k \neq 0_R$ and $\deg(k) < \deg(q)$, and ii) if $r, s \in R$ with

$rs \neq 0_R$ and $r \neq 0_R$, then $\deg(r) < \deg(rs)$. An element $r \in R \setminus \{0_R, 1_R\}$ is said to be *irreducible* if there are no elements $s, t \in R \setminus \{1_R\}$ such that $r = st$. The set of all irreducible elements of R is denoted by $\mathrm{IRR}(R)$. An element $r \in R \setminus \{0_R, 1_R\}$ is said to be *prime* if for all $s, t \in R$, whenever r divides st, then r divides s or r divides t. If R is Euclidean, then an element is irreducible if and only if it is prime.

3.2 Intractability Assumption

We define the following variant of the "strong root assumption" [23] over modules over Euclidean rings, which is a generalization of the strong RSA assumption. Let R_D be a module over some Euclidean ring R, and A be an element of D. The strong distinct-prime-product root problem with respect to A asks to find a set of distinct prime elements $\{e_i\}_{i \in S}$ in R and an element X in D such that $\left(\prod_{i \in S} e_i\right) \circ X = A$. We define the assumption in two variants depending on whether R_D and A are sampled with public coins.

Definition 6 (Strong Distinct-Prime-Product Root Modules (w/o Trusted Setup)). *Let I be some ordered set. Let $\mathcal{R}_{\mathcal{D}} = ((R_i)_{D_i})_{i \in I}$ be a family of modules. Let $\mathsf{GGen}(1^\lambda; \omega)$ be a deterministic algorithm which picks some $i \in I$ (hence some $R_D = (R_i)_{D_i} \in \mathcal{R}_{\mathcal{D}}$) and some element $A \in D$. $\mathcal{R}_{\mathcal{D}}$ is a strong distinct-prime-product root modules family if for all* PPT *adversary $\mathcal{A}$ there exists $\epsilon(\lambda) \in \mathsf{negl}(\lambda)$ such that*

$$\mathsf{Pr}\left[\begin{array}{l|c} \left(\prod_{i \in S} e_i\right) \circ X = A & \omega \leftarrow\!\!\$\, \{0,1\}^\lambda \\ \forall i \in S, e_i \in \mathrm{IRR}(R) & (R_D, A) := \mathsf{GGen}(1^\lambda; \omega) \\ \forall i \neq j \in S, e_i \neq e_j & (\{e_i\}_{i \in S}, X) \leftarrow \mathcal{A}(1^\lambda, R_D, A, \boxed{\omega}) \end{array}\right] \leq \epsilon(\lambda),$$

where $\mathcal{A}$ does not *receive ω (highlighted by the dashed box) as an input. If the inequality holds even if $\mathcal{A}$ receives ω as an input, then we say that $\mathcal{R}_{\mathcal{D}}$ is a strong distinct-prime-product root modules family* without trusted setup.

Lipmaa defined several variants of the (strong) root assumption with respect to random elements in D sampled with *private coin*, given the description of the module R_D sampled with *public coin*. It is safe to assume that the description of R_D includes a generating set of D. Viewed in this way, the above can be seen as defining an assumption directly with respect to the generating set. Note that the assumption of the existence of strong distinct-prime-product root modules (without trusted setup) is weaker than that of strong root modules (without trusted setup), where the latter requires the adversary to simply output (e, X) such that $e \neq 1_R$ and $e \circ X = A$.

It is apparent that RSA groups are strong distinct-prime-product root modules under the strong RSA assumption. We shall elaborate more in Section 6.

3.3 Subvector Commitments

Subvector commitments are a generalization of vector commitments [20], where the opening and updating operations are performed with respect to subvectors.

Definition 7 (Subvectors). *Let $q \in \mathbb{N}$, $\mathcal{M}$ be a set, and $(m_1, \ldots, m_q) \in \mathcal{M}^q$ be a vector. Let $I \subseteq [q]$ be an ordered index set. The I-subvector of $\boldsymbol{m}$ is defined as $\boldsymbol{m}_I := (m_{i_1}, \ldots, m_{i_{|I|}})$ where $I = (i_1, \ldots, i_{|I|})$.*

Definition 8 (Subvector Commitments (SVC)). *A subvector commitment scheme* SVC *is a tuple of* PPT *algorithms* $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Update}, \mathsf{ProofUpdate})$*:*

$\underline{\mathsf{Setup}(1^\lambda, q; \omega)}$*: The* deterministic *setup algorithm inputs the security parameter 1^λ, the length q of the committed vector, and a random tape ω. It outputs a public parameter* pp *(which implicitly defines the "vector space"[3] $\mathcal{M}^q$). We assume that all other algorithms input* pp *which we omit.*

$\underline{\mathsf{Com}(\boldsymbol{m})}$*: The committing algorithm inputs a vector $\boldsymbol{m} \in \mathcal{M}^q$. It outputs a commitment string C and some auxiliary information* aux*.*

$\underline{\mathsf{Open}(I, \boldsymbol{m}'_I, \mathsf{aux})}$*: The opening algorithm inputs an index set I, an I-subvector $\boldsymbol{m}'_I$, and some auxiliary information* aux*. It outputs a proof Λ_I that $\boldsymbol{m}'_I$ is the I-subvector of the committed vector.*

$\underline{\mathsf{Verify}(C, I, \boldsymbol{m}'_I, \Lambda_I)}$*: The verification algorithm inputs a commitment string C, an index set I, an I-subvector $\boldsymbol{m}'_I$, and a proof Λ_I. It accepts (i.e., it outputs 1) if and only if C is a commitment to $\boldsymbol{m}$ and $\boldsymbol{m}'_I$ is the I-subvector of $\boldsymbol{m}$.*

$\underline{\mathsf{Update}(C, J, \boldsymbol{m}_J, \boldsymbol{m}'_J, \mathsf{aux})}$*: The updating algorithm inputs a commitment string C, an index set J, an old J-subvector $\boldsymbol{m}_J$, a new J-subvector $\boldsymbol{m}'_J$, and some old auxiliary information* aux*. It outputs a new commitment C', some update information U, and some new auxiliary information* aux'*.*

$\underline{\mathsf{ProofUpdate}(C, I, \Lambda_I, J, \boldsymbol{m}'_J, U)}$*: The proof updating algorithm inputs a commitment string C, an index set I, a proof Λ_I for some I-subvector of some vector $\boldsymbol{m}$ committed in C, an index set J, a new J-subvector $\boldsymbol{m}'_J$, and some update information U. It outputs a new commitment string C' and new proof Λ'_I for some I-subvector of some vector $\boldsymbol{m}'$ committed in C, where $\boldsymbol{m}'_J$ is the J-subvector of $\boldsymbol{m}'$.*

Definition 9 (Correctness). *A subvector commitment scheme* SVC *is said to be correct if, for all security parameter $\lambda \in \mathbb{N}$, length $q \in \mathsf{poly}(\lambda)$, random tape $\omega \in \{0,1\}^\lambda$, public parameters* $\mathsf{pp} \in \mathsf{Setup}(1^\lambda, q; \omega)$ *(which defines the vector space $\mathcal{M}^q$), $\boldsymbol{m} \in \mathcal{M}^q$, index set $I \subseteq [q]$, $(C, \mathsf{aux}) \in \mathsf{Com}(\boldsymbol{m})$, $\Lambda_I \in \mathsf{Open}(I, \boldsymbol{m}_I, \mathsf{aux})$, it holds that*

$$\Pr\left[\mathsf{Verify}(C, I, \boldsymbol{m}_I, \Lambda_I) = 1\right] \geq 1 - \mathsf{negl}(\lambda)$$

and the same holds for updated commitments and proofs.

We consider the notion of position binding for subvector commitments. Recall that position binding for vector commitments requires that it is infeasible to open a commitment with respect to some position i to two distinct messages m_i and m'_i. We extend this notion to subvector commitments, by requiring that it is infeasible to open a commitment with respect to some index sets I and J to subvectors $\boldsymbol{m}_I$ and $\boldsymbol{m}'_J$, respectively, such that there exists an index $i \in I \cap J$ where $m_i \neq m'_i$. Furthermore, we may also require

[3] Note that the "vector space" described here is not necessarily a vector space in the mathematical sense. In particular, we do not define any arithmetic operations over $\mathcal{M}^q$.

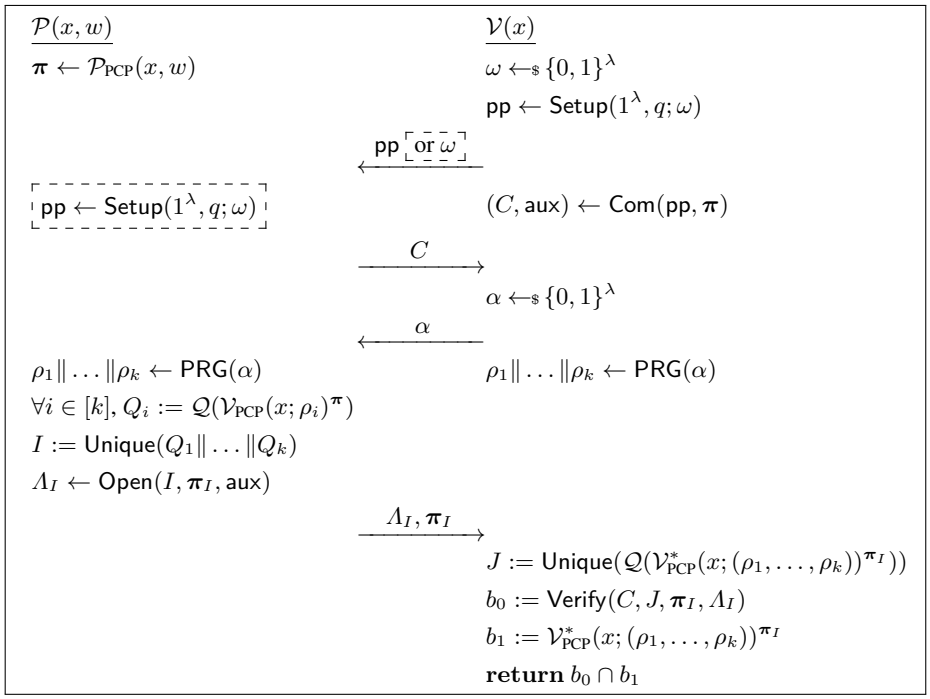

Fig. 1: Succinct Argument of Knowledge for NP

this property to hold without trusted setup. That is, the above should be infeasible even if the setup algorithm is public coin.

Definition 10 (Position Binding (without Trusted Setup)). *A subvector commitment* SVC *is position binding if for all* PPT *adversary $\mathcal{A}$, there exists a negligible function $\epsilon(\lambda) \in \mathsf{negl}(\lambda)$ such that*

$$\Pr\left[\begin{array}{l}\mathsf{Verify}(C, I, \boldsymbol{m}_I, \Lambda_I) = 1 \\ \mathsf{Verify}(C, J, \boldsymbol{m}'_J, \Lambda'_J) = 1 \\ \exists i \in I \cap J \ s.t.\ m_i \neq m'_i\end{array}\middle|\begin{array}{c}\omega \leftarrow_{\$} \{0,1\}^\lambda \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda; \omega) \\ (C, I, J, \boldsymbol{m}_I, \boldsymbol{m}'_J, \Lambda_I, \Lambda'_J) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}, \omega)\end{array}\right] \leq \epsilon(\lambda)$$

where $\mathcal{A}$ does not *receive ω (highlighted by the dashed box) as an input. If the inequality holds even if $\mathcal{A}$ receives ω as an input, then we say that* SVC *is position binding* without trusted setup.

As in [20], the hiding property can also be considered, and obtained generically from a (non-hiding) SVC scheme and a hiding (standard) commitment scheme. Since the hiding property is not necessary for our applications, we omit its definition.

4 Optimal Succinct Arguments of Knowledge for NP

Let $(\mathcal{P}_{\text{PCP}}, \mathcal{V}_{\text{PCP}})$ be a witness extractable PCP for NP with q being a bound on the size of the encoded proof, r being a bound on the length of the random coins of the (possibly adaptive) verifier, and k being a statistical parameter (such that γ^k is a negligible fraction). Let $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{k \cdot r}$ be a pseudo-random generator and let $\mathsf{SVC} := (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Update}, \mathsf{ProofUpdate})$ be a position binding subvector commitment (without trusted setup). We construct a 4-move interactive argument of knowledge in the standard model. Its formal description can be found in Figure 1.

4.1 Protocol Description

The prover first produces $\boldsymbol{\pi}$ as the PCP encoding of the witness w, while the verifier samples a random string ω. Suppose an SVC (with trusted setup) is used, the verifier computes the public parameter pp using ω, and sends pp to the prover. Otherwise, if an SVC without trusted setup is used, the verifier simply sends ω to the prover, so that both the prover and the verifier can compute the public parameter pp. After obtaining pp, the prover commits to $\boldsymbol{\pi}$ and sends its commitment C to the verifier. Once the verifier receives the commitment C, it responds with another random string α. The prover stretches α with a PRG into $\rho = \rho_1 \| \ldots \| \rho_k$ and executes the PCP verifier on ρ. The prover then records the sets of queries Q_i of $\mathcal{V}$ using randomness ρ_i to $\boldsymbol{\pi}$, and computes the opening of the commitment C at the unique positions $I = \mathsf{Unique}(Q_1 \| \ldots \| Q_k)$. The opening Λ_I, along with the corresponding bits $\boldsymbol{\pi}_I$ of $\boldsymbol{\pi}$ are sent to the verifier.

Let $\boldsymbol{\phi} = (\phi_1, \phi_2, \ldots) \in \{0,1\}^*$ and $\mathcal{V}^*_{\text{PCP}}(x; (\rho_1, \ldots, \rho_k))^{\boldsymbol{\phi}}$ be an algorithm which does the following: It maintains an initially empty ordered set Q^*. It iteratively runs $\mathcal{V}_{\text{PCP}}(x; \rho_i)$ from $i = 1$ to k, and maintain a global counter j for queries q^*_j made by $\mathcal{V}_{\text{PCP}}(x; \rho_i)$ for all $i \in [k]$. Upon receiving the j-th query q^*_j from $\mathcal{V}_{\text{PCP}}(x; \rho_i)$ (for some i), it first appends q^*_j to Q^*, and then checks if $q^*_j = q^*_{j'}$ for some $j' < j$. If so, it responds identically as for the j'-th query. Otherwise, it fetches the foremost entry of $\boldsymbol{\phi}$ which is not used as the response to a previous query, and use the fetched entry as the response to the current query. If such an entry cannot be fetched, it outputs 0. After running $\mathcal{V}_{\text{PCP}}(x; \rho_i)$ for all $i \in [k]$, it outputs 1 if and only if $\mathcal{V}_{\text{PCP}}(x; \rho_i)$ outputs 1 for all $i \in [k]$, and otherwise outputs 0. We slightly abuse the notation of $\mathcal{Q}$ and define $\mathcal{Q}(\mathcal{V}^*_{\text{PCP}}(x; (\rho_1, \ldots, \rho_k))^{\boldsymbol{\phi}}) := Q^*$. The verifier executes $\mathcal{V}^*_{\text{PCP}}$ over the same coins $\mathsf{PRG}(\alpha)$ and records the set of unique queries $J = \mathsf{Unique}(Q^*)$. It then checks whether Λ_I is a valid opening for $\boldsymbol{\pi}_I$ sent by the prover for the (ordered) set J. If both the PCP and the SVC verifications succeed, then the verifier returns 1, else it returns 0.

Completeness follows directly from the completeness of the PCP and of the SVC.

4.2 Communication Complexity

Suppose an SVC without trusted setup is used, the interaction between prover and verifier consists in the exchange of two random strings of length λ, a subvector commitment C, $O(k)$ bits of the proof encoding, and the corresponding opening Λ. Assuming a suitable instantiation of the subvector commitment where C and Λ are of size $O(\lambda)$ (see Section 6

for candidate instantiations), the amount of information exchanged accounts for $O(\lambda)$ bits. This is succinct and in fact optimal up to a constant factor.

Otherwise, if an SVC (with trusted setup) is used, then one of the random strings is replaced by a public parameter pp. Assuming again a suitable instantiation of the subvector commitment where pp is of size $O(\lambda)$ (see Section 6), the amount of communication is bounded by $O(\lambda)$ bits.

4.3 Analysis

The argument of knowledge of the protocol is established by the following theorem.

Theorem 4. *Let $(\mathcal{P}_{PCP}, \mathcal{V}_{PCP})$ be a witness extractable PCP for NP,* PRG *be a pseudorandom generator, and* $\mathsf{SVC} := (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Update}, \mathsf{ProofUpdate})$ *be a position binding (without trusted setup) subvector commitment. Let $k = \lambda$. Then the protocol in Figure 1 is a (public-coin) argument of knowledge.*

Proof. Let $\omega \in \{0,1\}^\lambda$ and $\rho_i \in \{0,1\}^{rk}$ for all $i \in \mathbb{N}$. Consider the following extractor. $\mathcal{E}^{\mathcal{A}}(x)$: On input a statement x, the extractor initializes an empty string $\boldsymbol{\pi} := \{\bot\}^q$, samples $\omega \leftarrow_\$ \{0,1\}^\lambda$, computes $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, q; \omega)$ and sends pp (or ω if using SVC without trusted setup) to $\mathcal{A}$. The adversary $\mathcal{A}$ replies with a certain commitment C. The extractor enters into a loop. In the i-th iteration of the loop, it performs the following:

1. Run $w \leftarrow \mathcal{E}_{\text{PCP}}(\boldsymbol{\pi})$, if $\mathcal{R}(x,w) = 1$ then return w and terminate the execution.
2. Sample a random $\alpha^i \leftarrow_\$ \{0,1\}^\lambda$ and send it to $\mathcal{A}$. Set $\rho^i := \mathsf{PRG}(\alpha^i)$.
3. $\mathcal{A}$ responds with $(\Lambda^i, \boldsymbol{\phi}^i)$, let Q^* be the set of queries of $\mathcal{V}^*_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\phi}^i}$.
4. If $\mathsf{Verify}(C, \mathsf{Unique}(Q^*), \boldsymbol{\phi}^i, \Lambda^i) = 0$ or $\mathcal{V}^*_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\phi}^i} = 0$ then rewind the adversary $\mathcal{A}$ and go to step 1 of the $(i+1)$-th iteration.
5. For all $q_j \in Q^*$, if $\pi_{q_j} \notin \{\bot, \phi^i_j\}$ then abort. Otherwise set $\pi_{q_j} := \phi^i_j$.
6. Rewind the adversary $\mathcal{A}$ and go to step 1 of the $(i+1)$-th iteration.

It is clear that whenever the extractor does not abort and terminates, then the extraction is successful. We first argue that the extractor does not abort within polynomially-many steps except with negligible probability.

Lemma 1. *For all statements x, all auxiliary information z, all PPT adversary $\mathcal{A}$, and all polynomials $p \in \mathsf{poly}(\lambda)$ it holds that*

$$\mathsf{Pr}\left[\bot \leftarrow \mathcal{E}^{\mathcal{A}(x,z)}(x) \textit{ within } p \textit{ iterations}\right] \leq \mathsf{negl}(\lambda)\,.$$

Proof (Lemma 1). We observe that the extractor aborts if and only if the adversary successfully opens one bit of the proof encoding $\boldsymbol{\pi}$ to two different bits. That is, there exists two sets of queries Q, Q', two openings Λ, Λ', two strings $\boldsymbol{\phi}$, $\boldsymbol{\phi}'$ and an index i such that

1. $\mathsf{Verify}(C, \mathsf{Unique}(Q), \boldsymbol{\phi}, \Lambda) = 1$,
2. $\mathsf{Verify}(C, \mathsf{Unique}(Q'), \boldsymbol{\phi}', \Lambda') = 1$,
3. $i \in Q \cap Q'$, and

4. $\phi_i \neq \phi'_i$.

This set of conditions contradicts the position binding (without trusted setup) of the subvector commitment scheme. It follows that the event where the extractor aborts happens only with negligible probability. □

The rest of the analysis establishes the probability of the extractor to terminate. First we introduce the following helping lemma.

Lemma 2. *For all statements x, all $f(\lambda) \in \mathsf{poly}(\lambda)$, all $\boldsymbol{\pi}^* \in \{0,1\}^{\leq q}$, all constant $\gamma \in (0,1)$, all $k \geq \frac{-\log f(\lambda)}{\log(1-\gamma)}$, and all strings $\rho = \rho_1 \| \ldots \| \rho_k \in \{0,1\}^{rk}$ such that*

$$\mathsf{Pr}\left[\mathcal{V}^*_{PCP}(x;\rho)^{\boldsymbol{\pi}^*} = 1\right] \geq \frac{1}{f(\lambda)},$$

over the random choice of ρ, then $\mathsf{Pr}\left[\mathcal{R}(x,w) = 1 | w \leftarrow \mathcal{E}_{PCP}(\boldsymbol{\pi}^*)\right] = 1$.

Proof (Lemma 2). Let

$$p_k := \mathsf{Pr}\left[\mathcal{V}^*_{\mathrm{PCP}}(x;\rho)^{\boldsymbol{\pi}^*} = 1\right] \text{ and } p_1 := \mathsf{Pr}\left[\mathcal{V}_{\mathrm{PCP}}(x;\rho_1)^{\boldsymbol{\pi}^*} = 1\right].$$

Since the random tapes ρ_i of the verifier are chosen uniformly we have that $p_k = (p_1)^k$ and therefore

$$p_1 = (p_k)^{\frac{1}{k}} \geq f(\lambda)^{\frac{-1}{k}}.$$

The following shows that $\frac{1}{f(\lambda)^k} \geq (1-\gamma)$:

$$\begin{aligned} k &\geq \frac{-\log f(\lambda)}{\log(1-\gamma)} \\ \frac{1}{k}\log\left(\frac{1}{f(\lambda)}\right) &\geq \log(1-\gamma) \\ \frac{1}{f(\lambda)^{\frac{1}{k}}} &\geq (1-\gamma). \end{aligned}$$

By Definition 5 it follows that the extractor $\mathcal{E}_{\mathrm{PCP}}$ is successful with probability 1. □

Next we argue that for any given string ϕ, running $\mathcal{V}^*_{\mathrm{PCP}}$ with ϕ over truly random coins induces the a distribution of outputs which is computationally indistinguishable from the distribution induced by $\mathcal{V}^*_{\mathrm{PCP}}$ executed with ϕ over pseudo-random coins. This is proven in the following lemma.

Lemma 3. *Let* PRG *be a pseudorandom generator. For all statements x, and all proof encodings $\phi \in \{0,1\}^{\leq q}$, the ensembles*

$$\left\{\mathcal{V}^*_{PCP}(x;\mathsf{PRG}(\alpha))^{\phi}\right\}_{\alpha \leftarrow\$ \{0,1\}^{\lambda}} \text{ and } \left\{\mathcal{V}^*_{PCP}(x;\rho)^{\phi}\right\}_{\rho \leftarrow\$ \{0,1\}^{rk}}$$

are computationally indistinguishable.

Proof (Lemma 3). Assume the contrary, then we can construct the following distinguisher against PRG: On input a string ρ, it executes $b \leftarrow \mathcal{V}_{\text{PCP}}(x;\rho)^{\boldsymbol{\pi}}$ using ρ as the random tape. Then it outputs b. By initial assumption we have that the distributions of the output of the verifier are non-negligibly far depending on the random tape, consequently so are the distributions of the output of the distinguisher for the two cases. This contradicts the pseudo-randomness of PRG and shows the veracity of our proposition. □

Next we show that if the adversary is successful with non-negligible probability, then the extractor outputs w in polynomially many steps. This is shown in two steps. First, we show that if the adversary is successful with non-negligible probability, then the extractor produces a string $\boldsymbol{\pi}^*$ which is accepted with non-negligible probability. Given such a string $\boldsymbol{\pi}^*$, we show that the PCP extractor must succeed in extracting a witness w.

Concretely, for any statement x and for any auxiliary input z consider an adversary $\mathcal{A}$ such that

$$\epsilon_{\mathcal{A}} := \mathsf{Pr}\left[(\mathcal{A}(x,z),\mathcal{V}(x))_{\Pi} = 1\right] \geq \frac{1}{\lambda^c}$$

for some constant c, which is given by assumption. Set $t = \frac{\lambda}{\epsilon_{\mathcal{A}}} \leq \lambda^{c+1}$, and for $i \in [t]$, let Π^i be and independent execution of the protocol. Then we have that

$$\mathsf{Pr}\left[\forall i \in [t], (\mathcal{A}(x,z),\mathcal{V}(x))_{\Pi^i} = 0\right] = (1-\epsilon_{\mathcal{A}})^t \leq e^{-\epsilon_{\mathcal{A}} t} \leq e^{-\lambda}.$$

This means that with all but negligible probability the adversary is going to be successful in at least one of these t executions. Let $\boldsymbol{\pi}^*$ be the variable maintained by $\mathcal{E}$ after $t \cdot s$-many iterations, for $s := t\lambda = \frac{\lambda^2}{\epsilon_{\mathcal{A}}} \leq \lambda^{c+2} \in \mathsf{poly}(\lambda)$. In the following we are going to show that, with overwhelming probability, there exists a constant d such that

$$\epsilon_{\mathcal{V}} := \mathsf{Pr}_{\alpha \leftarrow\$ \{0,1\}^\lambda}\left[\mathcal{V}_{\text{PCP}}(x;\mathsf{PRG}(\alpha))^{\boldsymbol{\pi}^*} = 1\right] \geq \frac{1}{\lambda^d}.$$

Assume the contrary that, with non-negligible probability, there exists a constant c with $\epsilon \geq \frac{1}{\lambda^c}$ and $\epsilon_{\mathcal{V}} < \frac{1}{\lambda^d}$ for all constants d. Let $\boldsymbol{\phi}^i$ be the variable sent by $\mathcal{A}$ in the i-th iteration and let X_i be 1 if the extractor reaches step 5 in the i-th iteration by running $\mathcal{V}_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\phi}^i}$, and 0 otherwise. Let X_i^* be defined like X_i except that the verifier $\mathcal{V}_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\pi}^*} = 1$ is executed on $\boldsymbol{\pi}^*$. By Lemma 1, with overwhelming probability $\mathcal{E}$ does not abort, and we have that the string $\boldsymbol{\pi}^*$ is uniquely defined. In other words, for all $i \in [t \cdot s]$ such that $X_i = 1$ then $\boldsymbol{\phi}^i \subseteq \boldsymbol{\pi}^*$. This implies that whenever $\mathcal{V}_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\phi}^i} = 1$, then $\mathcal{V}_{\text{PCP}}(x;\rho^i)^{\boldsymbol{\pi}^*} = 1$. It follows that

$$\overline{X}^* := \frac{X_1^* + \ldots + X_{t\cdot s}^*}{t \cdot s} \geq \frac{X_1 + \ldots + X_{t\cdot s}}{t \cdot s} \geq \frac{s}{t \cdot s} = \frac{1}{t}.$$

By assumption, for all constants d,

$$E[\overline{X}^*] = \mathsf{Pr}\left[\alpha \leftarrow\$ \{0,1\}^\lambda\right] \mathcal{V}_{\text{PCP}}(x;\mathsf{PRG}(\alpha))^{\boldsymbol{\pi}^*} = 1 \leq \frac{1}{\lambda^d}.$$

Let $d = c + 2$. By Hoeffding's inequality we have that

$$\begin{aligned}
&\mathsf{Pr}\left[\overline{X}^* - E[\overline{X}^*] \geq \frac{1}{t} - \frac{1}{\lambda^d}\right] \\
&\leq e^{-2ts\left(\frac{1}{t} - \frac{1}{\lambda^d}\right)^2} = e^{-2ts\left(\frac{1}{t^2} - \frac{2}{t\lambda^d} + \lambda^{-2d}\right)} = e^{-2\left(\lambda - \frac{2s}{\lambda^d} - \lambda^{-2d}\right)} \\
&\leq e^{-2\left(\lambda - \frac{2s}{\lambda^d}\right)} \leq e^{-2\left(\lambda - \frac{2\lambda^{c+2}}{\lambda^d}\right)} \leq e^{-2(\lambda-2)} = \mathsf{negl}(\lambda)\,.
\end{aligned}$$

contradicting the assumption that this event happens with non-negligible probability.

To recap, we have shown that

$$\epsilon_{\mathcal{V}} := \mathsf{Pr}_{\alpha \leftarrow\$ \{0,1\}^\lambda}\left[\mathcal{V}_{\mathrm{PCP}}(x; \mathsf{PRG}(\alpha))^{\boldsymbol{\pi}^*} = 1\right] \geq \frac{1}{\lambda^{c+2}}.$$

By Lemma 3, there exists a negligible function $\mathsf{negl}(\lambda)$ such that

$$\begin{aligned}
&\mathsf{Pr}_{\rho \leftarrow\$ \{0,1\}^{rk}}\left[\mathcal{V}_{\mathrm{PCP}}(x; \rho)^{\boldsymbol{\pi}^*} = 1\right] \\
&\geq \mathsf{Pr}_{\alpha \leftarrow\$ \{0,1\}^\lambda}\left[\mathcal{V}_{\mathrm{PCP}}(x; \mathsf{PRG}(\alpha))^{\boldsymbol{\pi}^*} = 1\right] - \mathsf{negl}(\lambda) \\
&\geq \frac{1}{\lambda^{c+2}} - \mathsf{negl}(\lambda)
\end{aligned}$$

Since $k = \lambda$, with sufficiently large λ, by Lemma 2, we have that

$$\mathsf{Pr}\left[\mathcal{R}(x, w) = 1 | w \leftarrow \mathcal{E}_{\mathrm{PCP}}(\boldsymbol{\pi}^*)\right] = 1.$$

This implies that the extractor terminates after $t \cdot s$ steps, except with negligible probability, and concludes the proof. □

5 Subvector Commitments from Modules over Euclidean Rings

Let GGen be an efficient algorithm as defined in Definition 6. Let R be an Euclidean ring sampled by GGen, and $e_1, \ldots, e_q$ be arbitrary distinct prime elements in R. Let $\mathcal{M}^q := \{0_R, 1_R\}^q$[4] where 0_R and 1_R are the additive and multiplicative identity elements of R respectively. We construct a subvector commitment scheme SVC in Figure 2.

Note that in the opening and proof updating algorithm, it is required to compute

$$\Lambda_I := \left(\prod_{i \in I} e_i\right)^{-1} \circ \langle \boldsymbol{m}_{[q]\setminus I}, \boldsymbol{S}_{[q]\setminus I} \rangle \quad \text{and}$$

$$\Lambda'_I := \Lambda_I + \left(\prod_{i \in I} e_i\right)^{-1} \circ \langle \boldsymbol{m}'_{([q]\setminus I)\cap J} - \boldsymbol{m}_{([q]\setminus I)\cap J}, \boldsymbol{S}_{([q]\setminus I)\cap J} \rangle$$

[4] In general, $\mathcal{M}$ can be set such that for all $m, m' \in \mathcal{M}$, $\gcd(m - m', e_i) = 1$ for all $i \in [q]$.

$\mathsf{Setup}(1^\lambda, q; \omega)$

$(R_D, A) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda, \omega)$

$\forall i \in [q], S_i := \left(\prod_{j \in [q]\setminus\{i\}} e_j\right) \circ A$

$\boldsymbol{S} := (S_1, \ldots, S_q)$

$\boldsymbol{e} := (e_1, \ldots, e_q)$

return $\mathsf{pp} := (R_D, A, \boldsymbol{S}, \boldsymbol{e})$

$\mathsf{Com}(\boldsymbol{m})$

$C := \langle \boldsymbol{m}, \boldsymbol{S} \rangle$

$\mathsf{aux} := \boldsymbol{m}$

return (C, aux)

$\mathsf{Open}(I, \boldsymbol{m}'_I, \mathsf{aux})$

parse aux **as** $\boldsymbol{m}$

$\Lambda_I := \left(\prod_{i \in I} e_i\right)^{-1} \circ \langle \boldsymbol{m}_{[q]\setminus I}, \boldsymbol{S}_{[q]\setminus I} \rangle$

return Λ_I

$\mathsf{Verify}(C, I, \boldsymbol{m}'_I, \Lambda_I)$

$b_0 := (\boldsymbol{m}'_I \in \mathcal{M}^{|I|})$

$b_1 := (C = \langle \boldsymbol{m}'_I, \boldsymbol{S}_I \rangle + \left(\prod_{i \in I} e_i\right) \circ \Lambda_I)$

return $b_0 \cap b_1$

$\mathsf{Update}(C, J, \boldsymbol{m}_J, \boldsymbol{m}'_J, \mathsf{aux})$

parse aux **as** $(m_1, \ldots, m_q)$

parse $\boldsymbol{m}'_J$ **as** $(m'_{j_1}, \ldots, m'_{j_{|J|}})$

$C' := C + \langle \boldsymbol{m}'_J - \boldsymbol{m}_J, \boldsymbol{S}_J \rangle$

$U := (J, \boldsymbol{m}_J, \boldsymbol{m}'_J)$

$\forall j \in [q], m''_j = \begin{cases} m_j & j \notin J \\ m'_j & j \in J \end{cases}$

$\mathsf{aux}' := (m''_1, \ldots, m''_q)$

return (C', U, aux')

$\mathsf{ProofUpdate}(C, I, \Lambda_I, J, \boldsymbol{m}'_J, U)$

parse U **as** $(J, \boldsymbol{m}_J, \boldsymbol{m}'_J)$

$C' := C + \langle \boldsymbol{m}'_J - \boldsymbol{m}_J, \boldsymbol{S} \rangle$

$\Lambda'_I := \Lambda_I + \left(\prod_{i \in I} e_i\right)^{-1}$
$\quad \circ \langle \boldsymbol{m}'_{([q]\setminus I)\cap J} - \boldsymbol{m}_{([q]\setminus I)\cap J}, \boldsymbol{S}_{([q]\setminus I)\cap J} \rangle$

return (C', Λ'_I)

Fig. 2: Construction of a Subvector Commitment Scheme SVC.

respectively. Although multiplicative inverses of ring elements do not exist in general, and if so, they may be hard to compute, the above are efficiently computable because, for all $i \in [q] \setminus I$ and hence for all $i \in ([q] \setminus I) \cap J$, we have

$$S_i := \left(\prod_{j \in [q]\setminus\{i\}} e_j\right) \circ A = \left(\prod_{j \in I} e_j \prod_{j \in [q]\setminus(I \cup \{i\})} e_j\right) \circ A.$$

The correctness of SVC follows straightforwardly by inspection.

Theorem 5. *If $\mathcal{R}_\mathcal{D}$ is a strong distinct-prime-product root modules family (without trusted setup), then* SVC *is position binding (without trusted setup).*

Proof. Suppose not, let $\mathcal{A}$ be a PPT adversary such that

$$\Pr\left[\begin{array}{l|c} \mathsf{Verify}(C, I, \boldsymbol{m}_I, \Lambda_I) = 1 & \omega \leftarrow\!\!\$\ \{0,1\}^\lambda \\ \mathsf{Verify}(C, J, \boldsymbol{m}'_J, \Lambda'_J) = 1 & \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda; \omega) \\ \exists i \in I \cap J \ s.t.\ m_i \neq m'_i & (C, I, J, \boldsymbol{m}_I, \boldsymbol{m}'_J, \Lambda_I, \Lambda'_J) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}, \omega) \end{array}\right] > \frac{1}{f(\lambda)}$$

for some polynomial $f(\lambda) \in \mathsf{poly}(\lambda)$, where $\mathcal{A}$ gets ω as input (highlighted by the dashed box) only in the variant without trusted setup. In any case, we have

$$\langle \boldsymbol{m}_I, \boldsymbol{S}_I \rangle + \left(\prod_{i \in I} e_i \right) \circ \Lambda_I = \langle \boldsymbol{m}'_J, \boldsymbol{S}_J \rangle + \left(\prod_{i \in J} e_i \right) \circ \Lambda'_J$$

which implies

$$\begin{aligned} &\langle \boldsymbol{m}_{I \setminus J}, \boldsymbol{S}_{I \setminus J} \rangle - \langle \boldsymbol{m}'_{J \setminus I}, \boldsymbol{S}_{J \setminus I} \rangle + \langle \boldsymbol{m}_{I \cap J} - \boldsymbol{m}'_{I \cap J}, \boldsymbol{S}_{I \cap J} \rangle \\ &= \left(\prod_{i \in I \cap J} e_i \right) \left(\left(\prod_{i \in J \setminus I} e_i \right) \circ \Lambda'_J - \left(\prod_{i \in I \setminus J} e_i \right) \circ \Lambda_I \right). \end{aligned}$$

Recall that $S_i = \left(\prod_{j \in [q] \setminus \{i\}} e_j \right) \circ A$. Define $\delta_i := \begin{cases} m_i & i \in I \setminus J \\ -m'_i & i \in J \setminus I \\ m_i - m'_i & i \in I \cap J \end{cases}$ and $\Lambda := \left(\left(\prod_{i \in J \setminus I} e_i \right) \circ \Lambda'_J - \left(\prod_{i \in I \setminus J} e_i \right) \circ \Lambda_I \right)$. We obtain

$$\left(\sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j \right) \circ A = \left(\prod_{i \in I \cap J} e_i \right) \circ \Lambda.$$

Let $K_0 := \{ i \in I \cap J : \delta_i = 0_R \}$ and $K_1 := \{ i \in I \cup J : \delta_i \neq 0_R \}$. Next, we show that $d := \gcd \left(\sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j, \prod_{i \in I \cap J} e_i \right) = \prod_{j \in K_0} e_j$. Furthermore, suppose that this is the case, we have $(I \cap J) \setminus K_0 \neq \emptyset$ since there exists $i \in I \cap J$ such that $\delta_i = m_i - m'_i \neq 0_R$. To prove the above, we first note that

$$\begin{aligned} \sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j &= \sum_{i \in K_1} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j \\ &= \prod_{j \in [q] \setminus (I \cup J)} e_j \left(\sum_{i \in K_1} \delta_i \prod_{j \in (I \cup J) \setminus \{i\}} e_j \right). \end{aligned}$$

Hence

$$\begin{aligned} d &= \gcd \left(\sum_{i \in K_1} \delta_i \prod_{j \in (I \cup J) \setminus \{i\}} e_j, \prod_{i \in I \cap J} e_i \right) \\ &= \prod_{j \in K_0} e_j \cdot \gcd \left(\sum_{i \in K_1} \delta_i \prod_{j \in (I \cup J) \setminus (K_0 \cup \{i\})} e_j, \prod_{i \in (I \cap J) \setminus K_0} e_i \right). \end{aligned}$$

It remains to show that $d' := \gcd \left(\sum_{i \in K_1} \delta_i \prod_{j \in (I \cup J) \setminus (K_0 \cup \{i\})} e_j, \prod_{i \in (I \cap J) \setminus K_0} e_i \right) = 1_R$. Suppose not, let $d' = \prod_{i \in L} e_i$ for some $L \subseteq (I \cap J) \setminus K_0$. Suppose $\ell \in L \neq \emptyset$.

This means $\delta_\ell \neq 0_R$ and hence $\ell \in K_1$. Then there exists $r \in R$ such that

$$\begin{aligned} e_\ell \cdot r = & \sum_{i \in K_1} \delta_i \prod_{j \in (I \cup J) \setminus (K_0 \cup \{i\})} e_j \\ = & \delta_\ell \prod_{j \in (I \cup J) \setminus (K_0 \cup \{\ell\})} e_j + e_\ell \sum_{i \in K_1 \setminus \{\ell\}} \delta_i \prod_{j \in (I \cup J) \setminus (K_0 \cup \{i\})} e_j. \end{aligned}$$

Let $r' := r - \sum_{i \in K_1 \setminus \{\ell\}} \delta_i \prod_{j \in (I \cup J) \setminus (K_0 \cup \{i\})} e_j$. We have

$$e_\ell \cdot r' = \delta_\ell \prod_{j \in (I \cup J) \setminus (K_0 \cup \{\ell\})} e_j.$$

Since $\delta_\ell \neq 0_R$, *i.e.*, $\delta_\ell \in \{-1_R, 1_R\}$, the above contradicts the fact that e_ℓ is a prime element. Thus we must have $L = \emptyset$ and hence $d' = 1_R$.

Now that we have concluded $d = \gcd\left(\sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j, \prod_{i \in I \cap J} e_i\right) = \prod_{j \in K_0} e_j$, we can use the extended Euclidean algorithm to find $a, b \in R$ such that

$$a \sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j + b \prod_{i \in I \cap J} e_i = \prod_{j \in K_0} e_j.$$

Multiplying this to A, we get

$$\begin{aligned} \left(\prod_{j \in K_0} e_j\right) \circ A = & \left(a \sum_{i \in I \cup J} \delta_i \prod_{j \in [q] \setminus \{i\}} e_j + b \prod_{i \in I \cap J} e_i = \prod_{j \in K_0} e_j\right) \circ A \\ = & \left(a \prod_{i \in I \cap J} e_i\right) \circ \Lambda + \left(b \prod_{i \in I \cap J} e_i\right) \circ A \\ = & \left(\prod_{i \in I \cap J} e_i\right) (a \circ \Lambda + b \circ A) \end{aligned}$$

Since $(I \cap J) \setminus K_0 \neq \emptyset$, we can set $S := (I \cap J) \setminus K_0$ and $X := (a \circ \Lambda + b \circ A)$, and output $(\{e_i\}_{i \in S}, X)$ as a solution to the strong distinct-prime-product root problem. □

6 Candidate Modules Families

In the following we suggest some candidate instantiations for groups of hidden order where the strong distinct-prime-root problem is conjectured to be hard.

6.1 Class Groups of Imaginary Quadratic Orders

The use of class groups in cryptography is first proposed by Buchmann and Williams [18]. We refer to, *e.g.*, [16,17], for more detailed discussions. We recall the preliminaries of

class groups necessary for our purpose. Let Δ be a negative integer such that $\Delta \equiv 0$ or $1 \pmod 4$. The ring $\mathcal{O}_\Delta := \mathbb{Z} + \frac{\Delta+\sqrt{\Delta}}{2}\mathbb{Z}$ is called an *imaginary quadratic order of discriminant* Δ. Its field of fractions is $\mathbb{Q}(\sqrt{\Delta})$. The discriminant is *fundamental* if $\Delta/4$ (resp. Δ) is square-free in the case of $\Delta \equiv 0 \pmod 4$ (resp. $\Delta \equiv 1 \pmod 4$). If Δ is fundamental, then $\mathcal{O}_\Delta$ is a *maximal* order. The *fractional ideals* of $\mathcal{O}_\Delta$ are of the form $q\left(a\mathbb{Z} + \frac{b+\sqrt{\Delta}}{2}\mathbb{Z}\right)$ with $q \in \mathbb{Q}$, $a \in \mathbb{Z}^+$, and $b \in \mathbb{Z}$, subject to the constraint that there exists $c \in \mathbb{Z}^+$ such that $\Delta = b^2 - 4ac$ and $\gcd(a, b, c) = 1$. A fractional ideal can therefore be represented by a tuple (q, a, b). If $q = 1$, then the ideal is called *integral* and can be represented by a tuple (a, b). An integral ideal (a, b) is *reduced* if it satisfies $-a < b \leq a \leq c$ and $b > 0$ if $a = c$. It is known that if an ideal (a, b) is reduced, then $a \leq \sqrt{|\Delta|/3}$. Two ideals $\mathfrak{a}, \mathfrak{b} \subseteq \mathcal{O}_\Delta$ are *equivalent* if there exists $0 \neq \alpha \in \mathbb{Q}(\sqrt{\Delta})$ such that $\mathfrak{b} = \alpha\mathfrak{a}$. It is known that, for each equivalence class of ideals, there exists exactly one reduced ideal which serves as the representative of the equivalence class. The set of equivalence classes of ideals equipped with ideal multiplication forms an Abelian group $Cl(\Delta)$ known as a *class group*.

Properties Useful in Cryptography. Since for all reduced ideals, $|b| \leq a \leq \sqrt{|\Delta|/3}$, $Cl(\Delta)$ is finite. For sufficiently large $|\Delta|$, no efficient algorithm is known for finding the cardinality of $Cl(\Delta)$, also known as the class number. Group operations can be performed efficiently, as there exists efficient algorithms for ideal multiplication and computing reduced ideals [16]. It remains to show how a random element can be sampled with public coin such that the corresponding strong root problem is believed to be hard. Below, we suggest a simple candidate sampling algorithm for concreteness, and leave the formal analysis and the design of other algorithms as an independent future work.

Assuming the extended Riemann hypothesis, $Cl(\Delta)$ is generated by the classes of all invertible prime ideals of norm smaller than $12(\log|\Delta|)^2$ [3], where the norm of a fractional ideal (q, a, b) is defined as q^2a ($= a$ for integral ideals). Since these ideals have norms logarithmic in $|\Delta|$, they can be found in polynomial time through exhaustive search. A random element can then be sampled by computing a power product of the elements in the generating set, with exponents randomly chosen from $[|\Delta|]$. Denote the generating set as $\{G_1, \ldots, G_k\}$ and the exponents as $\{c_1, \ldots, c_k\}$. Let GGen be an algorithm which generates a class group $Cl(\Delta)$ together with an element $A = \prod_{i\in[k]} G_i^{c_i}$ chosen by the above method, with an additional constraint that $d := \gcd(c_1, \ldots, c_k) = 1$.

The additional constraint is (almost) necessary for the strong distinct-prime-product root problem with respect to A to be hard: Suppose $d \neq 1$ and suppose further that d can be efficiently factorized into $\{e_i\}_{i\in S}$ such that $d = \prod_{i\in S} e_i$ for distinct primes $e_i \neq 1$. Define $X := \prod_{i\in[k]} G_i^{c_i/\prod_{i\in S} e_i}$. Then $(\{e_i\}_{i\in S}, X)$ is a solution to the strong distinct-prime-product root problem. Since it seems unreasonable to assume that d cannot be efficiently factorized into a product of distinct primes (see also the discussion of RSA-UFO below), we impose the much more reasonable restriction that $d = 1$. We conjecture that the additional constraint is sufficient for the strong distinct-prime-product root problem to be hard with respect to A sampled as above with public coin.

6.2 RSA Groups

RSA-based cryptosystems operate over $\mathbb{Z}_N^*$, the group of positive integers smaller and coprime with N, equipped with modular multiplication, where N is an integer with at least two distinct large prime factors. The security of these systems relies on the hardness of the (strong) root problem over $\mathbb{Z}_N^*$, known as the (strong) RSA assumption. Typically, the modulus N is chosen as a product of two distinct large primes p, q. However, the (strong) root problem over $\mathbb{Z}_N^*$ is easy if p and q are known. In other words, for N generated this way, $\mathbb{Z}_N^*$ is not a (strong) root module *without trusted setup*.

RSA-UFOs. The problem of constructing RSA-based accumulators without trapdoors was considered by Sander [48], who proposed a way to generate (k, ϵ)-"generalized RSA moduli of unknown complete factionization (USA-UFOs)" N which has at least two distinct k-bit prime factors with probability $1-\epsilon$, summarized as follows. Let $N_1, \ldots, N_r$ be random $3k$-bit integers with $r = O(\log 1/\epsilon)$. It is known that with constant probability N_i has at least two distinct k-bit prime factors [48]. It then follows that $N := \prod_{i\in[r]} N_i$ has at least two distinct k-bit prime factors. An important observation is that N can be generated with public coin (*e.g.*, using a random oracle). However, since N is a $3kr$-bit integer, any cryptosystem based on $\mathbb{Z}_N^*$ seems impractical.

7 Summary of Our Results

We conclude the paper with a summary of our main results.

7.1 Arguments of Knowledge with $O(\lambda)$ Communication (Theorem 1)

By Theorem 5, we obtain position binding SVC assuming the existence of strong distinct-prime-product root modules, which is implied by the strong RSA assumption. When instantiated with the strong RSA assumption, the public parameter $\mathsf{pp} = (R_D, A, \boldsymbol{S}, \boldsymbol{e})$, where $R_D = \mathbb{Z}_N^*$ for some $N \in \mathbb{N}$, can be succinctly represented by (N, A) which is of size $O(\lambda)$. This is because $\boldsymbol{e}$ is a vector of arbitrary distinct primes which can be derived publicly, and $\boldsymbol{S}$ is uniquely determined by A and $\boldsymbol{e}$. Furthermore, an opening consists of a single element in $\mathbb{Z}_N^*$, which is again of size $O(\lambda)$.

Next, by Theorem 4, we obtain a 4-move argument of knowledge system for NP assuming the existence of witness extractable PCPs for NP, pseudorandom generators, and a position binding SVC. Note that witness extractable PCPs for NP exist unconditionally, and both pseudorandom generators and position binding SVC can be obtained from the strong RSA assumption. By the complexity analysis in Section 4.2, the resulting system has communication $O(\lambda)$.

7.2 zk-SNARK with $O(\lambda)$ CRS and Proof Size in ROM (Theorem 2)

In the argument of knowledge protocol obtained above, the verifier is public-coin except for the first message, where it sends an SVC public parameter to the prover. However, since the public parameter is independent of the statement to be proven and can be

reused for proving different statements, the verifier can publish the public parameter as a common reference string, so that the remaining steps of the protocol form a 3-move protocol in which the verifier is public-coin. We can then apply the Fiat-Shamir [28] transformation to obtain a SNARK in the random oracle model with a CRS of size $O(\lambda)$ and $O(\lambda)$ proof size.

To turn this SNARK into a zk-SNARK, we can apply a generic transformation similar to the following.

Theorem 6 ([13]). *If adaptively-sound non-interactive zero-knowledge (NIZK) arguments and SNARKs exist, then zk-SNARGs exist. If furthermore the NIZK argument has proof of knowledge (PoK) then zk-SNARKs exist.*

Strictly speaking, the difference between such a transformation and the one we need, is that our SNARK is in the random oracle model. This might be problematic because we do not know how to prove statements involving random oracles. Below, we argue that we can use the same transformation without any issues.

In a nutshell, the transformation consists in using SNARK to prove the existence of a NIZKPoK proof of the existence of the witness w, instead of the existence of w directly. Adaptive NIZKPoK (*e.g.*, [7] and [33]) are known from standard assumptions in the common random string model, which can be made setup-free by sampling the string with a random oracle. In particular, the construction based on (enhanced) trapdoor permutation [7] can be instantiated with the (strong) RSA assumption. Note that the random oracle here is only used for generating the common random string, but is not used for generating and verifying the NIZKPoK proof. Therefore the transformation does *not* need to use SNARK to prove statements involving the random oracle.

7.3 Setup-Free zk-SNARK with $O(\lambda)$ Proof Size in ROM (Theorem 3)

In order to obtain a setup-free zk-SNARK, we use SVC without trusted setup instead in Theorem 4 and obtain a public-coin protocol. We can then apply the transformation as described above to obtain a setup-free zk-SNARK in the random oracle model. By Theorem 5, position binding SVC without trusted setup can be obtained assuming the existence of strong distinct-prime-product root modules without trusted setup. As discussed in Section 6, class groups seem to be a reasonable instantiation of such modules.

References

1. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2087–2104. ACM, 2017.
2. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. *Journal of the ACM (JACM)*, 45(1):70–122, 1998.
3. Eric Bach. Explicit bounds for primality testing and related problems. In *Mathematics of Computation*, volume 55 (191), pages 355–380, 1990.

4. Michael Backes, Manuel Barbosa, Dario Fiore, and Raphael M. Reischuk. ADSNARK: Nearly practical and privacy-preserving proofs on authenticated data. In IEEE S&P 2015 [38], pages 271–286.
5. Michael Backes, Dario Fiore, and Raphael M. Reischuk. Verifiable delegation of computation on outsourced data. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 863–874, Berlin, Germany, November 4–8, 2013. ACM Press.
6. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.
7. Mihir Bellare and Moti Yung. Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation. *Journal of Cryptology*, 9(3):149–166, 1996.
8. Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. Computational integrity with a public random string from quasi-linear PCPs. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 551–579, Paris, France, May 8–12, 2017. Springer, Heidelberg, Germany.
9. Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 24, page 134, 2017.
10. Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 90–108, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.
11. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 31–60, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany.
12. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 276–294, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.
13. Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. The hunting of the SNARK. *Journal of Cryptology*, 30(4):989–1066, October 2017.
14. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Fischlin and Coron [30], pages 327–357.
15. Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.
16. Johannes Buchmann and Safuat Hamdy. A survey on iq-cryptography. In *Tech. Report TI-4/01, Technische Universitäat Darmstadt, Fachbereich Informatik*, 2000.
17. Johannes Buchmann, Tsuyoshi Takagi, and Ulrich Vollmer. Number field cryptography. In *High Primes and Misdemeanours: Lectures in Honour of the 60th Birthday of Hugh Cowie Williams*, volume 41, pages 111–125, 2004.
18. Johannes Buchmann and Hugh C. Williams. A key-exchange system based on imaginary quadratic fields. *Journal of Cryptology*, 1(2):107–118, 1988.
19. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more.
20. Dario Catalano and Dario Fiore. Vector commitments and their applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 55–72, Nara, Japan, February 26 – March 1, 2013. Springer, Heidelberg, Germany.

21. Alessandro Chiesa, Eran Tromer, and Madars Virza. Cluster computing in zero knowledge. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 371–403, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.
22. Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur. Geppetto: Versatile verifiable computation. In IEEE S&P 2015 [38], pages 253–270.
23. Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 125–142, Queenstown, New Zealand, December 1–5, 2002. Springer, Heidelberg, Germany.
24. George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct NIZK arguments. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 532–550, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Heidelberg, Germany.
25. George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct nizk arguments. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 532–550. Springer, 2014.
26. Giovanni Di Crescenzo and Helger Lipmaa. Succinct np proofs from an extractability assumption. In *Conference on Computability in Europe*, pages 175–185. Springer, 2008.
27. Amos Fiat and Adi Shamir. Polymorphic arrays: A novel VLSI layout for systolic computers. In *25th FOCS*, pages 37–45, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press.
28. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany.
29. Dario Fiore, Rosario Gennaro, and Valerio Pastro. Efficiently verifiable computation on encrypted data. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14*, pages 844–855, Scottsdale, AZ, USA, November 3–7, 2014. ACM Press.
30. Marc Fischlin and Jean-Sébastien Coron, editors. *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.
31. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 626–645, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.
32. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.
33. Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459, Shanghai, China, December 3–7, 2006. Springer, Heidelberg, Germany.
34. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany.
35. Jens Groth. On the size of pairing-based non-interactive arguments. In Fischlin and Coron [30], pages 305–326.
36. Safuat Hamdy and Bodo Möller. Security of cryptosystems based on class groups of imaginary quadratic orders. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 234–247, Kyoto, Japan, December 3–7, 2000. Springer, Heidelberg, Germany.
37. Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.
38. *2015 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press.

39. Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press.
40. Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Efficient arguments without short pcps. In *Computational Complexity, 2007. CCC'07. Twenty-Second Annual IEEE Conference on*, pages 278–291. IEEE, 2007.
41. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30, San Diego, CA, USA, June 11–13, 2007. ACM Press.
42. Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732, Victoria, British Columbia, Canada, May 4–6, 1992. ACM Press.
43. Joe Kilian. Improved efficient arguments (preliminary version). In Don Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 311–324, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Heidelberg, Germany.
44. Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 169–189, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Heidelberg, Germany.
45. Helger Lipmaa. Secure accumulators from euclidean rings without trusted setup. In Feng Bao, Pierangela Samarati, and Jianying Zhou, editors, *ACNS 12*, volume 7341 of *LNCS*, pages 224–240, Singapore, June 26–29, 2012. Springer, Heidelberg, Germany.
46. Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453, Santa Fe, New Mexico, November 20–22, 1994. IEEE Computer Society Press.
47. Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 49–62, Cambridge, MA, USA, June 18–21, 2016. ACM Press.
48. Tomas Sander. Efficient accumulators without trapdoor extended abstracts. In Vijay Varadharajan and Yi Mu, editors, *ICICS 99*, volume 1726 of *LNCS*, pages 252–262, Sydney, Australia, November 9–11, 1999. Springer, Heidelberg, Germany.
49. Victor Shoup. OAEP reconsidered. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 239–259, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.
50. Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 1–18, San Francisco, CA, USA, March 19–21, 2008. Springer, Heidelberg, Germany.
51. Riad S Wahby, Ioanna Tzialla, Justin Thaler, and Michael Walfish. Doubly-efficient zksnarks without trusted setup.
52. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. In *2017 IEEE Symposium on Security and Privacy*, pages 863–880, San Jose, CA, USA, May 22–26, 2017. IEEE Computer Society Press.