Lattice-Based zk-SNARKs from Square Span Programs

Rosario Gennaro*
City College of New York, USA

Michele Minelli†
DIENS, École normale supérieure, CNRS

Anca Nitulescu‡
DIENS, École normale supérieure, CNRS

Michele Orrù§
DIENS, École normale supérieure, CNRS

ABSTRACT

Zero-knowledge SNARKs (zk-SNARKs) are non-interactive proof systems with short (i.e., independent of the size of the witness) and efficiently verifiable proofs. They elegantly resolve the juxtaposition of individual privacy and public trust, by providing an efficient way of demonstrating knowledge of secret information without actually revealing it. To this day, zk-SNARKs are widely deployed all over the planet and are used to keep alive a system worth billion of euros, namely the cryptocurrency Zcash. However, all current SNARKs implementations rely on so-called *pre-quantum* assumptions and, for this reason, are not expected to withstand cryptanalitic efforts over the next few decades.

In this work, we introduce the first designated-verifier zk-SNARK that can be instantiated from learning with errors (LWE), which is believed to be post-quantum secure. We provide a generalization in the spirit of Gennaro et al. (Eurocrypt'13) to the SNARK of Danezis et al. (Asiacrypt'14) that is based on Square Span Programs (SSPs) and relies on weaker computational assumptions. We focus on designated-verifier proofs and propose a protocol in which a proof consists of just 5 LWE encodings. We provide a concrete choice of parameters as well as extensive benchmarks on a C implementation, showing that our construction is practically instantiable.

KEYWORDS

SNARK, Square Span Programs, zero-knowledge, post-quantum.

1 INTRODUCTION

Succinct Non-Interactive Arguments of Knowledge. Proof systems [GMR89] are fundamental in theoretical computer science and cryptography. In a zero-knowledge proof system, a powerful prover P can prove to a weaker verifier V that a particular statement $x \in L$ is true, for some NP language L (with corresponding witness relation $\mathcal{R}$), without revealing any additional information about the witness.

A non-interactive argument requires the verifier V to generate a common reference string crs ahead of time and independently of the statement to be proved by the prover P. Such systems are called *succinct non-interactive arguments* (SNARGs) [GW11]. Several SNARGs constructions have been proposed, and the area of SNARGs has become popular in the last years with the proposal of constructions which introduced significant improvements in efficiency. An important remark is that all such constructions are based on non-falsifiable assumptions [Nao03], a class of assumptions that is likely to be inherent in proving the security of SNARGs (without random oracles), as shown by Gentry and Wichs [GW11].

Many SNARGs are also *arguments of knowledge* – so called SNARKs [BCCT12, BCC+14]. Intuitively speaking, the knowledge soundness property of SNARKs says that every prover producing a convincing proof must "know" a witness. Proofs of knowledge are useful in many applications, such as anonymous credentials, or delegation of computation where the untrusted worker contributes its own input to the computation, or recursive proof composition [Val08, BCCT13].

Public vs. Designated Verifiability. We distinguish two types of arguments of knowledge: *publicly verifiable* ones, where the verification algorithm takes as input only common reference string crs, and designated-verifier ones, where the verifier V generates together with the crs some additional private verification key vrs. In the first case, proofs are meant to be verified by anyone having access to the crs. In the case of designated-verifier proofs, the proof can be verified only by the verifier V knowing the secret information vrs. It is straightforward to note that, with the help of an encryption scheme, any publicly-verifiable proof system can be transformed into an analogous designated-verifier one (by just encrypting the proof under the verifier's key). It is nonetheless important to note that in the standard model, all NIZK constructions we are aware of so far somehow imply the existence of an encryption scheme.

Quadratic Span Programs. Gennaro, Gentry, Parno and Raykova [GGPR13] proposed a new, influential characterization of the complexity class NP using *Quadratic Span Programs* (QSPs), a natural extension of span programs defined by Karchmer and Wigderson [KW93]. They show there is a very efficient reduction from boolean circuit satisfiability problems to QSPs. The QSP approach was generalized in [BCI+13] under the concept of *Linear PCP* (LPCP) (there is a construction of an LPCP for a QSP satisfiability problem) – these are a form of interactive ZK proofs where security holds under the assumption that the prover is restricted to compute only linear combinations of its inputs. These proofs can then be turned into (designated-verifier) SNARKs by using an *extractable linear-only* encryption scheme, i.e., an encryption scheme where any adversary can output a valid new ciphertext only if this is an affine combination of some previous encryptions that the adversary had as input (intuitively this "limited malleability" of the encryption scheme, will force the prover into the above restriction).

*Rosario Gennaro was supported by NSF Award no. 1565403.
†Michele Minelli was supported by European Union's Horizon 2020 research and innovation programme under grant agreement no. H2020-MSCA-ITN-2014-643161 ECRYPT-NET.
‡Anca Nitulescu was supported by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 339563 – CryptoCloud).
§Michele Orrù was supported by ERC grant 639554 (project aSCEND).

SNARGs based on lattices. Recently, in two companion papers [BISW17, BISW18], Boneh et al. provided the first designated-verifier SNARGs construction based on lattice assumptions.

The first paper has two main results: an improvement on the LPCP construction in [BCI$^+$13] and a construction of linear-only encryption based on LWE. The second paper presents a different approach where the information-theoretic LPCP is replaced by a LPCP with multiple provers, which is then compiled into a SNARG again via linear-only encryption. The main advantage of this approach is that it reduces the overhead on the prover, achieving what they call *quasi-optimality*[1]. The stronger notion of knowledge soundness (which leads to SNARKs) can be achieved by replacing the linear-only property with a stronger (extractable) assumption [BCI$^+$13].

Our contributions. In this paper, we frame the construction of Danezis et al. [DFGK14] for Square Span Programs in the framework of "encodings" introduced by Gennaro et al. [GGPR13]. We slightly modify the definition of encoding to accommodate for the noisy nature of LWE schemes. This allows us to have a more fine-grained control over the error growth, while keeping previous example encodings still valid instantiations. Furthermore, SSPs are similar to but simpler than Quadratic Span Programs (QSPs) since they use a single series of polynomials, rather than 2 or 3. We use SSPs to build simpler and more efficient designated-verifier SNARKs and Non-Interactive Zero-Knowledge arguments (NIZKs) for circuit satisfiability (CIRC-SAT).

We think our work is complementary to [BISW17, BISW18]. However, there are several reasons why we believe that our approach is preferable:

- **Zero-Knowledge.** The LPCP-based protocols in [BISW17, BISW18] do not investigate the possibility of achieving zero-knowledge. This leaves open the question of whether zk-SNARKs can be effectively instantiated. Considering the LPCP constructed for a QSP satisfiability problem, there is a general transformation to obtain ZK property [BCI$^+$13]. However, in the case of "noisy" encodings, due to possible information leakages in the error term, this transformation cannot be directly applied. Our SNARK construction, being SSP-based, can be made ZK at essentially no cost for either the prover or the verifier. Our transformation is different, exploiting special features of SSPs, and yields a zk-SNARK with almost no overhead. Our construction constitutes the first (designated-verifier) zk-SNARK on lattices.
- **Weaker Assumptions.** The linear-only property introduced in [BCI$^+$13] implies all the security assumptions needed by a SSP-suitable encoding, but the reverse is not known to hold. Our proof of security therefore relies on weaker assumptions and, by doing so, "distills" the minimal known assumptions needed to prove security for SSP, and instantiates them with an LWE-based approach. We study the relations between our knowledge assumption and the (extractable) linear-only assumption in **??**.
- **Simplicity and Efficiency.** While the result in [BISW18] seems asymptotically more efficient than any SSP-based approach, we believe that, for many applications, the simplicity and efficiency of the SSP construction will still provide a concrete advantage in practice. We implemented and tested our scheme: we provide some possible concrete parameters for the instantiation of our zk-SNARKs in Table 1, whereas more details on the implementation, along with benchmark results, are presented in Section 6.

[1] This is the first scheme where the prover does not have to compute a cryptographic group operation for each wire of the circuit, which is instead true e.g., in QSP-based protocols.

2 PREREQUISITES

2.1 Notation

We denote the real numbers by $\mathbf{R}$, the natural numbers by $\mathbf{N}$, the integers by $\mathbf{Z}$ and the integers modulo some q by $\mathbf{Z}_q$. We denote by $\vec{a} \cdot \vec{b}$ the dot product between vectors $\vec{a}$ and $\vec{b}$. Let $\mathcal{R}$ bet a relation between statements denoted by u and witnesses denoted by w. By $\mathcal{R}(u)$ we denote the set of possible witnesses for the statement u. We let $L(\mathcal{R}) := \{u : \mathcal{R}(u) \neq \emptyset\}$ denote the language associated to $\mathcal{R}$.

2.2 Square Span Programs

We characterize NP as Square Span Programs (SSPs) over some field $\mathbf{F}$ of order p. SSPs were introduced first by Danezis et al. [DFGK14].

Definition 2.1 (SSP). A Square Span Program (SSP) over the field $\mathbf{F}$ is a tuple consisting of $m+1$ polynomials $v_0(x), \ldots, v_m(x) \in \mathbf{F}[x]$ and a target polynomial $t(x)$ such that $\deg(v_i(x)) \leqslant \deg(t(x))$ for all $i = 0, \ldots, m$. We say that the square span program ssp has size m and degree $d = \deg(t(x))$. We say that ssp accepts an input $a_1, \ldots, a_\ell \in \{0,1\}$ if and only if there exist $a_{\ell+1}, \ldots, a_m \in \{0,1\}$ satisfying:

$$t(x) \text{ divides } \left(v_0(x) + \sum_{i=1}^{m} a_i v_i(x)\right)^2 - 1.$$

We say that ssp verifies a boolean circuit $\mathtt{C} : \{0,1\}^\ell \to \{0,1\}$ if it accepts exactly those inputs $(a_1, \ldots, a_\ell) \in \{0,1\}^\ell$ satisfying $\mathtt{C}(a_1, \ldots, a_\ell) = 1$.

Theorem 2.2 ([DFGK14, Theorem 2]). *For any boolean circuit $\mathtt{C} : \{0,1\}^\ell \to \{0,1\}$ of m wires and n fan-in 2 gates and for any prime $p \geq \max(n, 8)$, there exists a degree $d = m + n$ square span program $\mathrm{ssp} = (v_0(x), \ldots, v_m(x), t(x))$ over a field $\mathbf{F}$ of order p that verifies $\mathtt{C}$.*

2.3 Succinct Non-Interactive Arguments

In this section we provide formal definitions for the notion of succinct non-interactive arguments of knowledge (SNARKs).

Definition 2.3. A *non-interactive (NI) proof system* for a relation $\mathcal{R}$ is a triple of algorithms $\Pi = (\mathsf{G}, \mathsf{P}, \mathsf{V})$ as follows:

$(\mathrm{vrs}, \mathrm{crs}, \mathrm{td}) \leftarrow \mathsf{G}(1^\lambda, \mathcal{R})$ the CRS generation algorithm takes as input some security parameter in unary 1^λ and outputs a common reference string crs that will be given publicly, a verification key vrs, and trapdoor information td.

$\pi \leftarrow \mathsf{P}(\mathrm{crs}, u, w)$ the prover algorithm takes as input the CRS, a statement u, and a witness w. It outputs some proof π.

Table 1: Security estimates for different choices of LWE parameters (circuit size fixed to $d = 2^{15}$), together with the corresponding sizes of the proof π and of the CRS (when using a seeded PRG for its generation).

security level	λ	n	$\log \alpha$	$\log q$	$\lvert\pi\rvert$	$\lvert\mathsf{crs}\rvert$	ZK
medium	168	1270	−150	608	0.46 MB	7.13 MB	
	162	1470	−180	736	0.64 MB	8.63 MB	✓
high	244	1400	−150	672	0.56 MB	7.88 MB	
	247	1700	−180	800	0.81 MB	9.37 MB	✓
paranoid	357	1450	−150	800	0.69 MB	9.37 MB	
	347	1900	−180	864	0.98 MB	10.1 MB	✓

bool $\leftarrow \mathsf{V}(\mathsf{vrs}, u, \pi)$ the verifier algorithm takes as input a statement u together with a proof π, and vrs. It outputs **true** if the proof was accepted, **false** otherwise.

In the same line of past works [DFGK14, Fuc18], we will assume for simplicity that crs can be extracted from the verification key vrs, and that the unary security parameter 1^λ as well as the relation $\mathcal{R}$ can be inferred from the crs.

Non-interactive proof systems are generally asked to satisfy some security properties that simultaneously protect the prover from the disclosure of the witness, and the verifier from a forged proof. We now examine some of these notions.

A proof is complete if every correctly-generated proof verifies. More formally,

Definition 2.4 (Completeness). A non-interactive proof system Π for the relation $\mathcal{R}$ is (computationally) *complete* if for any PPT adversary $\mathcal{A}$:

$$\mathsf{Adv}^{\mathrm{compl}}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\mathrm{COMPL}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda) = \textbf{true}\right] = \mathsf{negl}(\lambda),$$

where $\mathrm{COMPL}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda)$ is the game depicted in Fig. 1.

The concept that the prover "must know" a witness is expressed by assuming that such knowledge can be efficiently extracted from the prover by means of a so-called *knowledge extractor*. For any prover able to produce a valid proof, there exists an efficient algorithm which, when given the same inputs as the prover (and the same random coins), is capable of extracting a witness for the given statement. Formally:

Definition 2.5 (Knowledge Soundness). A non-interactive proof system Π for the relation $\mathcal{R}$ is *knowledge-sound* if for any PPT adversary $\mathcal{A}$ there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ such that:

$$\mathsf{Adv}^{\mathrm{ksnd}}_{\Pi,\mathcal{R},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) \coloneqq \Pr\left[\mathrm{KSND}_{\Pi,\mathcal{R},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) = \textbf{true}\right] = \mathsf{negl}(\lambda),$$

where $\mathrm{KSND}_{\Pi,\mathcal{R},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$ is defined in Figure 1.

An *argument of knowledge* is a *knowledge-sound* proof system. If the adversary is computationally unbounded, we speak of *proofs* rather than arguments.

Remark 1. *An important consideration that arises when defining knowledge soundness in the designated-verifier setting is whether the adversary should be granted access to a verification oracle. Pragmatically, allowing the adversary to query a verification oracle captures the fact that CRS can be reused* $\mathsf{poly}(\lambda)$ *times. In the specific case of our construction, we formulate and prove our protocol allowing the adversary access to the* $\Pi.\mathsf{V}(\mathsf{vrs},\cdot,\cdot)$ *oracle (which has been named* strong soundness *in the past [BISW17]).*

A proof system Π for $\mathcal{R}$ is zero-knowledge if no information about the witness is leaked by the proof. More precisely:

Definition 2.6 (Zero-Knowledge). A non-interactive proof system Π is *zero-knowledge* if there exists a simulator Sim such that for any PPT adversary $\mathcal{A}$:

$$\mathsf{Adv}^{\mathrm{zk}}_{\Pi,\mathcal{R},\mathsf{Sim},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\mathrm{ZK}_{\Pi,\mathcal{R},\mathsf{Sim},\mathcal{A}}(\lambda) = \textbf{true}\right] = \mathsf{negl}(\lambda),$$

where $\mathrm{ZK}_{\Pi,\mathcal{R},\mathsf{Sim},\mathcal{A}}(\lambda)$ is defined in Figure 1.

Succinctness. Finally, we say that a proof system Π is *succinct* if the proof has size (quasi-)linear in the security parameter, i.e., $|\pi| = \tilde{O}(\lambda)$. If an NI proof system satisfies all above properties, it is then called succinct non-interactive argument of knowledge (SNARK).

Definition 2.7 (SNARK). A *succinct non-interactive argument of knowledge* (SNARK) is a non-interactive proof system that is complete, succinct, and knowledge-sound. Π is a zk-SNARK if it is a SNARK with zero-knowledge.

Publicly verifiable vs. designated verifier. If security (knowledge soundness) holds against adversaries that have also access to the verification state vrs (i.e., $\mathcal{A}$ receives vrs) then the SNARK is called publicly verifiable, otherwise it is designated-verifier. For simplicity, in the remainder of this work all constructions and proofs are given for the designated-verifier setting.

2.4 Encoding Schemes

Encoding schemes for SNARKs were initially introduced in [GGPR13]. Here, we present a variant of this definition that accommodates for encodings with noise.

Definition 2.8 (Encoding Scheme). An encoding scheme Enc over a field $\mathbf{F}$ is composed of the following algorithms:

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$, a key generation algorithm that takes as input some security parameter in unary 1^λ and outputs some secret state sk together with some public information pk. To ease notation, we are going to assume the message space

Game $\mathrm{KSND}_{\Pi,\mathcal{R},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$
$(\mathsf{crs},\mathsf{vrs}) \leftarrow \Pi.\mathsf{G}(1^\lambda,\mathcal{R})$
$(u,\pi;w) \leftarrow (\mathcal{A}\|\mathsf{Ext}_{\mathcal{A}})^{\Pi.\mathsf{V}(\mathsf{vrs},\cdot)}(\mathsf{crs})$
return $(\mathcal{R}(u,w) = \textbf{false} \wedge \Pi.\mathsf{V}(\mathsf{vrs},u,\pi))$

Game $\mathrm{COMPL}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda)$
$(\mathsf{crs},\mathsf{vrs},\mathsf{td}) \leftarrow \Pi.\mathsf{G}(1^\lambda,\mathcal{R})$
$(u,w) \leftarrow \mathcal{A}(\mathsf{crs})$
$\pi \leftarrow \Pi.\mathsf{P}(\mathsf{crs},u,w)$
return $(\Pi.\mathsf{V}(\mathsf{vrs},u,\pi) = \textbf{false}$ **and** $\mathcal{R}(u,w))$

Game $\mathrm{ZK}_{\Pi,\mathcal{R},\mathrm{Sim},\mathcal{A}}(\lambda)$
$(\mathsf{crs},\mathsf{vrs},\mathsf{td}) \leftarrow \Pi.\mathsf{G}(1^\lambda,\mathcal{R})$
$b \leftarrow\$ \{0,1\}$
$b' \leftarrow \mathcal{A}^{\textsc{Prove}}(\mathsf{vrs})$
return $(b = b')$

Oracle $\textsc{Prove}(u,w)$
if $\mathcal{R}(u,w) =$ **false return** $\perp$
if $b = 1$ $\pi \leftarrow \Pi.\mathsf{P}(\mathsf{crs},u,w)$
else $\pi \leftarrow \mathrm{Sim}(\mathsf{td},u)$
return π

Figure 1: Games for completeness (COMPL), knowledge soundness (KSND), and zero-knowledge (ZK).

is always part of the public information and that pk can be derived from sk.

- $S \leftarrow \mathsf{E}(a)$, a non-deterministic encoding algorithm mapping a field element a to some encoding space S, such that $\{\{\mathsf{E}(a)\} : a \in \mathbf{F}\}$ partitions S, where $\{\mathsf{E}(a)\}$ denotes the set of the possible evaluations of the algorithm E on a, that is $\{\mathsf{E}(a;r) : r \in \mathsf{E.rl}(\lambda)\}$.

The above algorithms must satisfy the following properties:

d-linearly homomorphic: there exists a $\mathrm{poly}(\lambda)$ algorithm Eval that, given as input the public parameters pk, a vector of encodings $(\mathsf{E}(a_1),\ldots,\mathsf{E}(a_d))$, and coefficients $\vec{c} = (c_1,\ldots,c_d) \in \mathbf{F}^d$, outputs a valid encoding of $\vec{a}\cdot\vec{c}$ with probability overwhelming in λ.

quadratic root detection: there exists an efficient algorithm that, given some parameter δ (either pk or sk), $\mathsf{E}(a_0),\ldots,\mathsf{E}(a_t)$, and the quadratic polynomial $\mathsf{pp} \in \mathbf{F}[x_0,\ldots,x_t]$, can distinguish if $\mathsf{pp}(a_1,\ldots,a_t) = 0$. With a slight abuse of notation, we will adopt the writing $\mathsf{pp}(\mathsf{ct}_0,\ldots,\mathsf{ct}_t) = 0$ to denote the quadratic root detection algorithm with inputs $\delta,\mathsf{ct}_0,\ldots,\mathsf{ct}_t$, and pp.

image verification: there exists an efficiently computable algorithm $\in$ that, given as input some parameter δ (again, either pk or sk), can distinguish if an element c is a correct encoding of a field element.

Decoding algorithm. When using a homomorphic encryption scheme in order to instantiate an encoding scheme, we simply define the *decoding algorithm* D as the decryption procedure of the scheme. More specifically, since we study encoding schemes derived from encryption functions, quadratic root detection and image verification for designated-verifiers are trivially obtained by using the decryption procedure D.

2.5 Assumptions

Assumption 1 (q-PKE). *The q-Power Knowledge of Exponent (q-PKE) assumption holds relative to an encoding scheme* Enc *and for the class $\mathcal{Z}$ of auxiliary input generators if, for every non-uniform PPT auxiliary input generator $Z \in \mathcal{Z}$ and non-uniform* PPT *adversary $\mathcal{A}$, there exists a non-uniform extractor* Ext *such that:*

$$\mathrm{Adv}^{\mathrm{pke}}_{\mathsf{Enc},Z,\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) := \Pr\left[\text{q-PKE}_{\mathsf{Enc},Z,\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) = \textbf{\textit{true}}\right] = \mathrm{negl}(\lambda),$$

where $\text{q-PKE}_{\mathsf{Enc},Z,\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$ *is the game depicted in Figure 2.*

Assumption 2 (q-PDH). *The q-Power Diffie-Hellman (q-PDH) assumption holds for encoding* Enc *if for all* PPT *adversaries $\mathcal{A}$ we have:*

$$\mathrm{Adv}^{\text{q-pdh}}_{\mathsf{Enc},\mathcal{A}}(\lambda) := \Pr\left[\text{q-PDH}_{\mathsf{Enc},\mathcal{A}}(\lambda) = \textbf{\textit{true}}\right] = \mathrm{negl}(\lambda),$$

where $\text{q-PDH}_{\mathsf{Enc},\mathcal{A}}(\lambda)$ *is defined as in Figure 2.*

Assumption 3 (q-PKEQ). *The q-Power Knowledge of Equality (q-PKEQ) assumption holds for the encoding scheme* Enc *if, for every* PPT *adversary $\mathcal{A}$, there exists an extractor* $\mathsf{Ext}_{\mathcal{A}}$ *such that:*

$$\mathrm{Adv}^{\text{q-pkeq}}_{\mathsf{Enc},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) := \Pr\left[\text{q-PKEQ}_{\mathsf{Enc},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) = \textbf{\textit{true}}\right] = \mathrm{negl}(\lambda),$$

where $\text{q-PKEQ}_{\mathsf{Enc},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$ *is the game depicted in Figure 2.*

3 AN ENCODING SCHEME BASED ON LEARNING WITH ERRORS

In this section we describe a possible encoding scheme based on learning with errors (LWE).

Lattice-based encoding scheme. We propose an encoding scheme Enc that consists of three algorithms as depicted in Figure 4. This is a slight variation of the classical LWE cryptosystem initially presented by Regev [Reg05] and later extended in [BV11]. The encoding scheme Enc is described by parameters $\Gamma := (q,n,p,\alpha)$, with $q,n,p \in \mathbf{N}$ such that $(p,q) = 1$, and $0 < \alpha < 1$. Our construction is an extension of the one presented in [BV11].

We assume the existence of a deterministic algorithm Pg that, given as input the security parameter in unary 1^λ, outputs an LWE encoding description Γ. The choice of using a *deterministic* parameter generation Pg was already argued by Bellare et al. [BFS16].

Assumption 4 (dLWE). *The decisional Learning With Errors (dLWE) assumption holds for the parameter generation algorithm* Pg *if for any* PPT *adversary $\mathcal{A}$:*

$$\mathrm{Adv}^{\mathrm{dlwe}}_{\mathsf{Pg},\mathcal{A}}(\lambda) := \Pr\left[\mathrm{dLWE}_{\mathsf{Pg},\mathcal{A}}(\lambda) = \textbf{\textit{true}}\right] - 1/2 = \mathrm{negl}(\lambda),$$

where $\mathrm{dLWE}_{\mathsf{Pg},\mathcal{A}}(\lambda)$ *is defined as in Figure 3.*

In [Reg05], Regev showed that solving the decisional LWE problem is as hard as solving some lattice problems in the worst case.

Leftover hash lemma (LHL). We now recall the definition of min-entropy, and the famous "leftover hash lemma" introduced by Impagliazzo et al. [HILL99].

Definition 3.1 (Min-entropy). The min-entropy of a random variable X is defined as

$$H_\infty(X) = -\log\left(\max_x \Pr[X = x]\right)$$

Lemma 3.2 (Leftover hash lemma). *Assume a family of functions* $\left\{\mathcal{H}_x : \{0,1\}^n \to \{0,1\}^\ell\right\}_{x\in X}$ *is* universal, *i.e.,* $\forall a \neq b \in \{0,1\}^n$,

$$\Pr_{x\in X}[\mathcal{H}_x(a) = \mathcal{H}_x(b)] = 2^{-\ell}.$$

Game q-PKE$_{\mathsf{Enc},Z,\mathcal{A},\mathsf{Ext}_{\mathcal{A}},z}(\lambda)$

$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$
$\alpha, s \hookleftarrow \mathbf{F}^*$
$\sigma \leftarrow (\mathsf{pk}, \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^q))$
$z \leftarrow Z(\mathsf{pk}, \sigma)$
$(\mathsf{ct}, \widehat{\mathsf{ct}}; a_0, \ldots, a_q) \leftarrow (\mathcal{A} \| \mathsf{Ext}_{\mathcal{A}})(\sigma, z)$
return $(\widehat{\mathsf{ct}} - \alpha \mathsf{ct} = 0) \wedge \mathsf{ct} \notin \left\{\mathsf{E}(\sum_i^q a_i s^i)\right\}$

Game q-PKEQ$_{\mathsf{Enc},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$

$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$
$s \hookleftarrow \mathbf{F}$
$\sigma \leftarrow (\mathsf{pk}, \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}(s^{q+2}), \ldots, \mathsf{E}(s^{2q}))$
$(\mathsf{E}(c), e; b) \leftarrow (\mathcal{A} \| \mathsf{Ext}_{\mathcal{A}})(\sigma)$
if $b = 0$ **return** $e \in \{\mathsf{E}(c)\}$
else return $e \notin \{\mathsf{E}(c)\}$

Game q-PDH$_{\mathsf{Enc},\mathcal{A}}(\lambda)$

$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$
$s \hookleftarrow \mathbf{F}$
$\sigma \leftarrow (\mathsf{pk}, \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}(s^{q+2}), \ldots, \mathsf{E}(s^{2q}))$
$y \leftarrow \mathcal{A}(\sigma)$
return $y \in \left\{\mathsf{E}(s^{q+1})\right\}$

Figure 2: Games for q-PKE, q-PKEQ, q-PDH assumptions.

Game dLWE$_{\mathsf{Pg},\mathcal{A}}(\lambda)$

$\Gamma := (p, q, n, \alpha) := \mathsf{Pg}(1^\lambda)$
$\vec{s} \hookleftarrow \mathbf{Z}_q^n$
$b \hookleftarrow \{0,1\}$
$b' \leftarrow \mathcal{A}^{\textsc{Encode}}(\Gamma)$
return $(b = b')$

Oracle Encode

$\vec{a} \hookleftarrow \mathbf{Z}_q^n$
$e \leftarrow \chi_{q\alpha}$
if $b = 1$ $c := \vec{s} \cdot \vec{a} + e$
else $c \hookleftarrow \mathbf{Z}_q$
return $(\vec{a}, c)$

Figure 3: The decisional LWE problem for parameters Γ.

Then, for any random variable Y,

$$\Delta\left((X, \mathcal{H}_X(Y)), (X, U_\ell)\right) \leq \frac{1}{2}\sqrt{2^{-H_\infty(Y)} \cdot 2^\ell},$$

where $U_\ell \hookleftarrow \{0,1\}^\ell$.

We now present a version of the LHL that will be useful later in the paper, when proving the zero-knowledge property of our construction. In a nutshell, it says that, if parameters are set correctly, a random linear combination of the columns of a matrix is statistically close to a uniformly random vector.

Lemma 3.3 ("Specialized" leftover hash lemma). *Let n, p, q, d be non-negative integers. Let $\mathbf{A} \hookleftarrow \mathbf{Z}_q^{n \times d}$, and $\vec{r} \hookleftarrow \mathbf{Z}_p^d$. Then we have*

$$\Delta\left((\mathbf{A}, \mathbf{A}\vec{r}), (\mathbf{A}, \vec{u})\right) \leq \frac{1}{2}\sqrt{p^{-d} \cdot q^n},$$

where $\mathbf{A}\vec{r}$ is computed modulo q, and $\vec{u} \hookleftarrow \mathbf{Z}_q^n$.

Proof. For the vector $\vec{r}$, we have that $H_\infty(\vec{r}) = d \log p$. Then the proof is immediate from Lemma 3.2:

$$\Delta\left((\mathbf{A}, \mathbf{A}\vec{r}), (\mathbf{A}, \vec{u})\right) \leq \frac{1}{2}\sqrt{2^{-d\log p} \cdot q^n} = \frac{1}{2}\sqrt{p^{-d} \cdot q^n}.$$

□

Definition 3.4. An encoding scheme Enc is *correct* if, for any $\vec{s} \leftarrow \mathsf{K}(1^\lambda)$ and $m \in \mathbf{Z}_p$,

$$\Pr\left[\mathsf{D}(\vec{s}, \mathsf{E}(\vec{s}, m)) \neq m\right] = \mathsf{negl}(\lambda).$$

We say that an encoding ct of a message m under secret key $\vec{s}$ is *valid* if $\mathsf{D}(\vec{s}, \mathsf{ct}) = m$. We say that an encoding is *fresh* if it is generated through the E algorithm. We say that an encoding is *stale* if it is not fresh.

Lemma 3.5 (Correctness). *Let $\mathsf{ct} = (-\vec{a}, \vec{a} \cdot \vec{s} + pe + m)$ be an encoding. Then* ct *is a valid encoding of a message $m \in \mathbf{Z}_p$ if $e < \frac{q}{2p}$.*

Image verification and quadratic root detection can be implemented using D, providing the secret key as input.

In order to bound the size of the noise, we first need a basic theorem on the tail bound of discrete Gaussian distributions due to Banaszczyk [Ban95]:

Lemma 3.6 ([Ban95, Lemma 2.4]). *For any $\sigma, T \in \mathbf{R}^+$ and $\vec{a} \in \mathbf{R}^n$:*

$$\Pr\left[\vec{x} \leftarrow \chi_\sigma^n : |\vec{x} \cdot \vec{a}| \geq T\sigma \|\vec{a}\|\right] < 2\exp(-\pi T^2). \tag{1}$$

At this point, this corollary follows:

Corollary 3.7. *Let $\vec{s} \hookleftarrow \mathbf{Z}_q^n$ be a secret key and $\vec{m} = (m_0, \ldots, m_{d-1}) \in \mathbf{Z}_p^d$ be a vector of messages. Let $\vec{\mathsf{ct}}$ be a vector of d fresh encodings so that $\vec{\mathsf{ct}}_i \leftarrow \mathsf{E}(\vec{s}, m_i)$, and $\vec{c} \in \mathbf{Z}_p^d$ be a vector of coefficients. If $q > 2p^2\sigma\sqrt{\frac{\kappa d}{\pi}}$, then $\mathsf{Eval}\left(\vec{c}, \vec{\mathsf{ct}}\right)$ outputs a valid encoding of $\vec{m} \cdot \vec{c}$ under the secret key $\vec{s}$ with probability overwhelming in κ.*

Proof. The fact that the message part is $\vec{m} \cdot \vec{c}$ is trivially true by simple homomorphic linear operations on the encodings. Then the final encoding is valid if the error does not grow too much during these operations. Let $\vec{e} \in \mathbf{Z}_p^d$ be the vector of all the error terms in the d encodings, and let $T = \sqrt{\kappa/\pi}$. Then by Lemma 3.6 we have:

$$\Pr\left[\vec{e} \leftarrow \chi_\sigma^d : |\vec{e} \cdot \vec{c}| \geq \sqrt{\frac{\kappa}{\pi}}\sigma \|\vec{c}\|\right] < 2\exp(-\kappa).$$

For correctness we need the absolute value of the final noise to be less than $q/2p$ (cf. Lemma 3.5). Since it holds that $\forall \vec{c} \in \mathbf{Z}_p^d$, $\|\vec{c}\| \leq p\sqrt{d}$, we can state that correctness holds if

$$\sqrt{\frac{\kappa}{\pi}}\sigma p\sqrt{d} < \frac{q}{2p}$$

which gives $q > 2p^2\sigma\sqrt{\frac{\kappa d}{\pi}}$. □

Smudging. When computing a linear combination of encodings, the distribution of the error term in the final encoding does not result in a correctly distributed fresh encoding. The resulting error distribution depends on the coefficients used for the linear combination, and despite correctness of the decryption still holds, the error could reveal more than just the plaintext. We combine homomorphic evaluation with a technique called *smudging* [AJL+12], which "smudges out" any difference in the distribution that is due to the coefficients of the linear combination, thus hiding any potential information leak. This technique has been also called "noise flooding" in the past [BPR12].

$\mathsf{K}(1^\lambda)$
$\Gamma := (p, q, n, \alpha) := \mathsf{Pg}(1^\lambda)$
$\vec{s} \leftarrow\!\!\$\ \mathbf{Z}_q^n$
return $(\Gamma,\ \vec{s}\,)$

$\mathsf{E}(\vec{s}, m)$
$\Gamma := (p, q, n, \alpha) := \mathsf{Pg}(1^\lambda)$
$\vec{a} \leftarrow\!\!\$\ \mathbf{Z}_q^n$
$\sigma := q\alpha;\ e \leftarrow \chi_\sigma$
return $(-\vec{a},\ \vec{a}\cdot\vec{s} + pe + m)$

$\mathsf{D}(\vec{s}, (\vec{c}_0, c_1))$
$\Gamma := (p, q, n, \alpha) := \mathsf{Pg}(1^\lambda)$
return $(\vec{c}_0 \cdot \vec{s} + c_1) \pmod p$

Figure 4: An encoding scheme based on LWE.

Procedure test-error$(\vec{s}, (\vec{c}_0, c_1))$
$\Gamma := (p, q, n, \alpha) := \mathsf{Pg}(1^\lambda)$
$e' := (\vec{c}_0 \cdot \vec{s} + c_1)\ //\ p$
return (??)

Figure 5: The error testing procedure.

Lemma 3.8 (Noise Smudging, [BGGK17]). *Let $B_1 = B_1(\kappa)$ and $B_2 = B_2(\kappa)$ be positive integers. Let $x \in [-B_1, B_1]$ be a fixed integer and $y \leftarrow\!\!\$\ [-B_2, B_2]$. Then the distribution of y is statistically indistinguishable from that of $y + x$, as long as $B_1/B_2 = \mathsf{negl}(\kappa)$.*

Error testing. By making non-blackbox use of our LWE encoding scheme, it is possible to define an implementation of the function test-error (cf. Section 2) in order to guarantee the existence of a security reduction from adversarially-generated proofs. In fact, it is not sufficient to show that a series of homomorphic operations over a forged proof can break one of the assumptions. We must also guarantee that these manipulations do not alter the correctness of the encoded value. In the specific case of LWE encodings, it is sufficient to use the secret key, recover the error, and enforce an upper bound on its norm. A possible implementation of test-error is displayed in Figure 5.

4 OUR DESIGNATED-VERIFIER ZK-SNARK

Let Enc be an encoding scheme (Definition 2.8). Let C be some circuit taking as input an ℓ_u-bit string and outputting 0 or 1. Let $\ell := \ell_u + \ell_w$, where ℓ_u is the length of the "public" input, and ℓ_w the length of the private input. The value m corresponds to the number of wires in C and n to the number of fan-in 2 gates. Let $d := m + n$. We construct a zk-SNARK scheme for any relation $\mathcal{R}_\mathsf{C}$ on pairs $(u, w) \in \{0,1\}^{\ell_u} \times \{0,1\}^{\ell_w}$ that can be computed by a polynomial size circuit C with m wires and n gates. Our protocol is formally depicted in Figure 6.

CRS generation. The setup algorithm G takes as input some complexity 1^λ in unary form and the circuit $\mathsf{C} : \{0,1\}^{\ell_u} \times \{0,1\}^{\ell_w} \to \{0,1\}$. It generates a square span program that verifies C by running:

$$(v_0(x), \ldots, v_m(x), t(x)) \leftarrow \mathsf{SSP}(\mathsf{C})$$

Finally, it samples $\alpha, \beta, s \leftarrow \mathbf{F}$ such that $t(s) \neq 0$, and returns the CRS:

$$\begin{aligned}\mathsf{crs} := \Big(&\mathsf{ssp},\ \mathsf{pk}, \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^d),\\ &\mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^d),\\ &\mathsf{E}(\beta t(s)), (\mathsf{E}(\beta v_i(s)))_{i=\ell_u+1}^m\Big)\end{aligned} \tag{2}$$

Prover. The prover algorithm, on input some statement $u := (a_1, \ldots, a_{\ell_u})$, computes a witness $w := (a_{\ell_u+1}, \ldots, a_m)$ such that $(u \| w) = (a_1, \ldots, a_m)$ is a satisfying assignment for the circuit C. The $(a_i)_i$ are such that:

$$t(x) \text{ divides } \left(v_0(x) + \sum_{i=1}^m a_i v_i(x)\right)^2 - 1,$$

as per Theorem 2.2. Then, it samples $\gamma \leftarrow\!\!\$\ \mathbf{F}$ and sets $v(x) := v_0(x) + \sum_{i=1}^m a_i v_i(x) + \gamma t(x)$. Let:

$$h(x) := \frac{(v_0(x) + \sum_i^m a_i v_i(x) + \gamma t(x))^2 - 1}{t(x)} = \frac{v(x)^2 - 1}{t(x)}, \tag{3}$$

whose coefficients can be computed from the polynomials provided in the ssp. By linear evaluation it is possible to compute

$$\begin{aligned}&H := \mathsf{E}(h(s)), \qquad \widehat{H} := \mathsf{E}(\alpha h(s)), \qquad \widehat{V} := \mathsf{E}(\alpha v(s)),\\ &V_w := \mathsf{E}\left(\sum_{i=\ell_u+1}^m a_i v_i(s) + \gamma t(s)\right),\\ &B_w := \mathsf{E}\left(\beta\left(\sum_{i=\ell_u+1}^m a_i v_i(s) + \gamma t(s)\right)\right).\end{aligned} \tag{4}$$

In fact, H - respectively, $\widehat{H}$ - can be computed from the encodings of $1, s, \ldots, s^d$ - respectively, $\alpha, \alpha s, \ldots, \alpha s^d$ - and the coefficients of Equation (3). The element $\widehat{V}$ can be computed from the encodings of $\alpha s, \ldots, \alpha s^d$. Finally, V_w - respectively, B_w - can be computed from the encodings of $s, \ldots, s^d$ - respectively, $\beta t(s), \beta v_{\ell_u+1}(s), \ldots, \beta v_m(s)$. All these linear evaluations involve at most $d + 1$ terms and the coefficients are bounded by p. Using the above elements, the prover returns a proof $\pi := (H, \widehat{H}, \widehat{V}, V_w, B_w)$.

Verifier. Upon receiving a proof π and a statement $u = (a_1, \ldots, a_{\ell_u})$, the verifier, in possession of the verification key vrs, proceeds with the following verifications. First, it uses the quadratic root detection algorithm of the encoding scheme Enc to verify that

Setup $\Pi.\mathsf{G}(1^\lambda, \mathsf{C})$

$\alpha, \beta, s \leftarrow_\$ \mathbf{F}$; $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$

$(v_0, \ldots, v_m(x), t(x)) \leftarrow \mathsf{SSP}(\mathsf{C})$

Compute crs as per Eq. (2)

$\mathsf{vrs} \coloneqq \mathsf{td} \coloneqq (\mathsf{sk}, s, \alpha, \beta)$

return (vrs, crs, td)

Verifier $\Pi.\mathsf{V}(\mathsf{vrs}, u, \pi)$

$(H, \widehat{H}, \widehat{V}, V_w, B_w) \coloneqq \pi$

$(a_1, a_2, \ldots a_{\ell_u}) \coloneqq u$; $(\mathsf{sk}, s, \alpha, \beta) \coloneqq \mathsf{vrs}$

Read $(v_0, \ldots, v_m(x), t(x))$ from vrs

$w_s \coloneqq \mathsf{D}(V_w)$; $b_s \coloneqq \mathsf{D}(B_w)$

$h_s \coloneqq \mathsf{D}(H)$; $\widehat{h}_s \coloneqq \mathsf{D}(\widehat{H})$

$\widehat{v}_s \coloneqq \mathsf{D}(\widehat{V})$; $t_s \coloneqq t(s)$

$v_s \coloneqq v_0(s) + \sum_{i=1}^{\ell_u} a_i v_i(s) + w_s$

Check Eqs. (eq-pke) to (eq-lin)

return test-error(sk, B_w)

Prover $\Pi.\mathsf{P}(\mathsf{crs}, u, w)$

$(v_0, \ldots, v_m(x), t(x)) \leftarrow \mathsf{SSP}(\mathsf{C})$

$u \coloneqq (a_1, \ldots, a_{\ell_u}) \in \{0,1\}^{\ell_u}$;

$w \coloneqq (a_{\ell_u+1}, \ldots, a_m)$

$v(x) \coloneqq v_0(x) + \sum_{i=1}^{m} a_i v_i(x) + \gamma t(x)$

$v_{\mathsf{mid}}(x) \coloneqq \sum_{i>\ell_u}^{m} a_i v_i(x) + \gamma t(x)$

$h(x) = (v(x)^2 - 1)/t(x)$

/ Compute the proof terms as per Eq. (4)

$H \coloneqq \mathsf{Eval}((\mathsf{E}(s^i))_i^d,\ (h_i)_i^d) = \mathsf{E}(h(s))$

$\widehat{H} \coloneqq \mathsf{Eval}((\mathsf{E}(\alpha s^i))_i^d,\ (h_i)_i^d) = \mathsf{E}(\alpha h(s))$

$\widehat{V} \coloneqq \mathsf{Eval}((\mathsf{E}(\alpha s^i)_i^d,\ (v_i)_i^d) = \mathsf{E}(\alpha v(s))$

$B_w \coloneqq \mathsf{Eval}((\mathsf{E}(\beta v_i(s)))_i^m \| (\mathsf{E}(\beta t(s))),\ (a_i)_i^m \| (\gamma))$

$V_w \coloneqq \mathsf{Eval}((\mathsf{E}(s^i))_i^d,\ (v_{\mathsf{mid}\,i})_i^d) = \mathsf{E}(v_{\mathsf{mid}}(s))$

Apply smudging on $H, \widehat{H}, \widehat{V}, B_w, V_w$

return $(H, \widehat{H}, \widehat{V}, V_w, B_w)$

Figure 6: Our zk-SNARK protocol Π.

the proof satisfies:

$$\widehat{h}_s - \alpha h_s = 0 \text{ and } \widehat{v}_s - \alpha v_s = 0, \qquad \text{(eq-pke)}$$

$$(v_s^2 - 1) - h_s t_s = 0, \qquad \text{(eq-div)}$$

$$b_s - \beta w_s = 0. \qquad \text{(eq-lin)}$$

where $(h_s, \widehat{h}_s, \widehat{v}_s, w_s, b_s)$ are the values encoded in $(H, \widehat{H}, \widehat{V}, V_w, B_w) \coloneqq \pi$ and t_s, v_s are computed as $t_s \coloneqq t(s)$ and $v_s \coloneqq v_0 + \sum_{i=1}^{\ell_u} a_i v_i(s) + w_s$.

If all above checks hold, return **true**. Otherwise, return **false**.

Simulator $\mathsf{Sim}(\mathsf{td}, u)$

$(\mathsf{sk}, s, \alpha, \beta) \coloneqq \mathsf{td}$; $(a_1, \ldots, a_{\ell_u}) \coloneqq u$

$\gamma_w \leftarrow_\$ \mathbf{F}$

$h \coloneqq \left((v_0(s) + \sum_i^{\ell_u} a_i v_i(s) + \gamma_w)^2 - 1\right) \ / \ t(s)$

$H \leftarrow \mathsf{E}(h)$; $\widehat{H} \leftarrow \mathsf{E}(\alpha h)$; $\widehat{V} \leftarrow \mathsf{E}(\alpha v_0(s) + \sum_i^{\ell_u} a_i \alpha v_i(s) + \alpha \gamma_w)$

$V_w \leftarrow \mathsf{E}(\gamma_w)$; $B_w \leftarrow \mathsf{E}(\beta \gamma_w)$

Apply smudging on $H, \widehat{H}, \widehat{V}, B_w, V_w$

return $(H, \widehat{H}, \widehat{V}, V_w, B_w)$

Figure 7: Simulator for Zero-Knowledge.

5 PROOFS OF SECURITY

In this section, we prove our main theorem:

Theorem 5.1. *If the q-PKE, q-PKEQ and q-PDH assumptions hold for the encoding scheme* Enc*, the protocol* Π *on* Enc *is a zk-SNARK with statistical completeness, statistical zero-knowledge and computational knowledge soundness.*

Proof of statistical completeness. Corollary 3.7 states the conditions on Γ for which the homomorphically computed encodings are valid with probability at least $1 - \mathsf{negl}(\kappa)$. **??** affirms that correctly generated proofs satisfy **??** with probability overwhelming in κ. Therefore test-error returns true and completeness follows trivially by Theorem 2.2. □

5.1 Zero-Knowledge

To obtain a zero-knowledge protocol, we do two things: we add a smudging term to the noise of the encoding, in order to make the distribution of the final noise independent of the coefficients a_i, and we randomize the target polynomial $t(x)$ to hide the witness. The random vectors constituting the first element of the ciphertext are guaranteed to be statistically indistinguishable from uniformly random vectors by leftover hash lemma (cf. Lemma 3.3).

5.2 Knowledge Soundness

We provide some intuition in an informal sketch of the security reductions: the CRS for the scheme contains encodings of $\mathsf{E}(s), \ldots, \mathsf{E}(s^d)$, as well as encodings of these terms multiplied by some field elements $\alpha, \beta \in \mathbf{F}$. The scheme requires the prover P to exhibit encodings computed homomorphically from such CRS.

The reason why we require the prover to duplicate its effort w.r.t. α is so that the simulator in the security proof can extract representations of $\widehat{V}$, $\widehat{H}$ as degree-d polynomials $v(x)$, $h(x)$ such that $v(s) = v_s, h(s) = h_s$, by the q-PKE assumption (for $q = d$). The assumption also guarantees that this extraction is efficient. This explains the first quadratic root detection check Equation (eq-pke) in the verification algorithm.

Suppose an adversary manages to forge a SNARK of a false statement and pass the verification test. Then, the soundness of

the square span program (Theorem 2.2) implies that, for the extracted polynomials $v(x), h(x)$ and for the new defined polynomial $v_{\mathsf{mid}}(x) := v(x) - v_0(x) - \sum_i^{\ell_u} a_i v_i(x)$, one of the following must be true:

i. $h(x)t(x) \neq v^2(x) - 1$, but $h(s)t(s) = v^2(s) - 1$, from Equation (eq-div);
ii. $v_{\mathsf{mid}}(x) \notin \mathrm{Span}(v_{\ell_u+1}, \ldots, v_m)$, but B_w is a valid encoding of $\mathsf{E}(\beta v_{\mathsf{mid}}(s))$, from Equation (eq-lin).

If the first case holds, then $p(x) := (v^2(x) - 1) - h(x)t(x)$ is a nonzero polynomial of degree some $k \leq 2d$ that has s as a root, since the verification test implies $(v^2(s) - 1) - h(s)t(s) = 0$. The simulator can use $p(x)$ to solve q-PDH for $q \geq 2d - 1$ using the fact that $\mathsf{E}\left(s^{q+1-k}p(s)\right) \in \{\mathsf{E}(0)\}$ and subtracting off encodings of lower powers of s to get $\mathsf{E}(s^{q+1})$.

To handle the second case, i.e., to ensure that $v_{\mathsf{mid}}(x)$ is in the linear span of the $v_i(x)$'s with $\ell_u < i \leq m$ we use an extra scalar β, supplement the CRS with the terms $\{\mathsf{E}(\beta v_i(s))\}_{i>\ell_u}, \mathsf{E}(\beta t(s))$, and require the prover to present (encoded) $\beta v_{\mathsf{mid}}(s)$ in its proof. An adversary against q-PDH will choose a polynomial $\beta(x)$ convenient to solve the given instance. More specifically, it sets $\beta(x)$ with respect to the set of polynomials $\{v_i(x)\}_{i>\ell_u}$ such that the coefficient for x^{q+1} in $\beta(x)v_{\mathsf{mid}}(x)$ is non-zero. Then, for the values in the crs it uses $\beta := \beta(s)$. All these allow it to run the SNARK adversary and to obtain from its output B_w an encoding of some polynomial with coefficient s^{q+1} nonzero and thus solve q-PDH. Also here, the verification algorithm guarantees that even with all the above homomorphic operations, the challenger still decrypts the correct value with $1 - \mathsf{negl}(\kappa)$ probability. As previously mentioned in Remark 1, the proof of knowledge soundness allows oracle access to the verification procedure. In the context of a weaker notion of soundness where the adversary does not have access to the $\Pi.\mathsf{V}(\mathsf{vrs}, \cdot, \cdot)$ oracle, the proof is almost identical, except that there is no need for the $\mathcal{B}^{\mathrm{PDH}}$ adversary to answer queries and to simulate the verification, and therefore no need for the q-PKEQ assumption anymore. This greatly simplifies our construction: the protocol does not need to rely on the q-PKEQ assumption, and the prime modulus can be of κ bits.

6 EFFICIENCY AND CONCRETE PARAMETERS

The prover's computations are bounded by the security parameter and the size of the circuit, i.e., $\mathsf{P} \in \widetilde{O}(\lambda d)$. As in [GGPR13, DFGK14], the verifier's computations depend solely on the security parameter, i.e., $\mathsf{V} \in O(\lambda)$. The proof consists of a constant number (precisely, 5) of LWE encodings, i.e., $|\pi| = 5 \cdot \widetilde{O}(\lambda)$. Finally, the complexity for the setup procedure is $\widetilde{O}(\lambda d)$.

Using the propositions from Section 3 and knowing the exact number of homomorphic operations that need to be performed in order to produce a proof, we can now attempt at providing some concrete parameters for our encoding scheme.

We fix the statistical security parameter $\kappa := 32$, as already done in past works on fully homomorphic encryption (e.g., [DM15, CGGI16]). We fix the circuit size $d := 2^{15}$, which is sufficient for some practical applications such as the computation of SHA-256.

Table 2: Detailed comparison with previous work. PQ stands for post-quantum.

	λ	PQ	ZK	$\lvert\pi\rvert$	$\lvert\mathrm{crs}\rvert$	multi gates
[PHGR13]	256	✗	✓	288 B	6.50 MB	23,785
[BISW17]	100	✗	✗	0.02 MB	1.23 GB	10,000
this work	162	✓	✓	0.64 MB	8.63 MB	10,922

For some practical examples of circuits, we direct the reader towards [BCG+14a, PHGR13].

For a first attempt at implementing our solution, we assume a weaker notion of soundness, i.e., that in the KSND game the adversary does not have access to a verification oracle (cf. Figure 1). Concretely, this means that the only bound in the size of p is given by the guessing probability of the witness, and the guessing of a field element. We thus fix p to be a prime[2] of 32 bits for the size of the message space.

In Table 2 we show a comparison between our implementation, the zk-SNARK of [PHGR13] (informally called "Pinocchio"), and the recent implementation of [BISW17] by Samir Menon, Brennan Shacklett, and David Wu[3]. Despite the fact that the construction of Parno et al. [PHGR13] is fundamentally different as it targets encoding over elliptic curves, we believe that they provide a good term of comparison (when used with circuits of the same size) for the loss incurred when using lattice-based encodings instead. Note therefore that the security parameter of [PHGR13] is not comparable with the two other results.

Moreover, it is worth noting that the implementation of [BISW17] targets 80 bits of security, which is justified using the estimate provided in [LP11]. We report $\lambda = 100$ as given by Albrecht's tool [APS15], which we believe to be more accurate. Nonetheless, the estimated post-quantum security level is 50, thus insufficient for modern applications. Additionally, we note that, despite targeting the construction of SNARGs, it seems the construction of [BISW17] can be turned into a SNARK by using the stronger extractable linear-only assumption. In order to achieve this, they can use a technique called *double encryption*, which doubles the size of each ciphertext. More details about this are given in **??**. Finally, we remark that perhaps our "trick" of using a PRG to generate the random part of the encoding, might be applied to the construction of [BISW17] as well.

[2] In particular, we need p and q to be relatively prime for the correctness of the encoding scheme [BV11, footnote 18].
[3] Results are extracted from the source code at https://github.com/dwu4/lattice-snarg.

REFERENCES

[ABL+17] Divesh Aggarwal, Gavin K Brennen, Troy Lee, Miklos Santha, and Marco Tomamichel. Quantum attacks on bitcoin, and how to protect against them. *arXiv preprint arXiv:1710.10377*, 2017.

[ACFP14] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, and Ludovic Perret. Algebraic algorithms for LWE. Cryptology ePrint Archive, Report 2014/1018, 2014. http://eprint.iacr.org/2014/1018.

[AFG14] Martin R. Albrecht, Robert Fitzpatrick, and Florian Göpfert. On the efficacy of solving LWE by reduction to unique-SVP. In Hyang-Sook Lee and Dong-Guk Han, editors, *ICISC 13*, volume 8565 of *LNCS*, pages 293–310. Springer, Heidelberg, November 2014.

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP 2011, Part I*, volume 6755 of *LNCS*, pages 403–415. Springer, Heidelberg, July 2011.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 483–501. Springer, Heidelberg, April 2012.

[Alb17] Martin R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 103–129. Springer, Heidelberg, May 2017.

[APS15] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

[Ban95] Wojciech Banaszczyk. Inequalities for convex bodies and polar reciprocal lattices inr n. *Discrete & Computational Geometry*, 13(2):217–231, 1995.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 440–456. Springer, Heidelberg, May 2005.

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, October 1988.

[BCC+14] Nir Bitansky, Ran Canetti, Alessandro Chiesa, Shafi Goldwasser, Huijia Lin, Aviad Rubinstein, and Eran Tromer. The hunting of the SNARK. Cryptology ePrint Archive, Report 2014/580, 2014. http://eprint.iacr.org/2014/580.

[BCCT12] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In Shafi Goldwasser, editor, *ITCS 2012*, pages 326–349. ACM, January 2012.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 111–120. ACM Press, June 2013.

[BCG+13] 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. Springer, Heidelberg, August 2013.

[BCG+14a] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

[BCG+14b] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. Cryptology ePrint Archive, Report 2014/349, 2014. http://eprint.iacr.org/2014/349.

[BCI+13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 315–333. Springer, Heidelberg, March 2013.

[BCTV14] 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. Springer, Heidelberg, August 2014.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988.

[BFS16] Mihir Bellare, Georg Fuchsbauer, and Alessandra Scafuro. NIZKs with an untrusted CRS: Security in the face of parameter subversion. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 777–804. Springer, Heidelberg, December 2016.

[BG14] Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary LWE. In Willy Susilo and Yi Mu, editors, *ACISP 14*, volume 8544 of *LNCS*, pages 322–337. Springer, Heidelberg, July 2014.

[BGGK17] Dan Boneh, Rosario Gennaro, Steven Goldfeder, and Sam Kim. A lattice-based universal thresholdizer for cryptographic systems. Cryptology ePrint Archive, Report 2017/251, 2017. http://eprint.iacr.org/2017/251.

[BISW17] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. Lattice-based SNARGs and their application to more efficient obfuscation. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 247–277. Springer, Heidelberg, May 2017.

[BISW18] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. Quasi-optimal snargs via linear multi-prover interactive proofs. Cryptology ePrint Archive, Report 2018/133, 2018. https://eprint.iacr.org/2018/133.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012.

[BSBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. https://eprint.iacr.org/2018/046.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 97–106. IEEE Computer Society Press, October 2011.

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2016.

[Dam92] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, August 1992.

[DFGK14] 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. Springer, Heidelberg, December 2014.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 617–640. Springer, Heidelberg, April 2015.

[Fuc18] Georg Fuchsbauer. Subversion-zero-knowledge snarks. In *IACR International Workshop on Public Key Cryptography*, pages 315–347. Springer, 2018.

[Gal13] Steven D. Galbraith. Space-efficient variants of cryptosystems based on learning with errors. preprint, 2013. https://www.math.auckland.ac.nz/~sgal018/compact-LWE.pdf.

[GGPR13] 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. Springer, Heidelberg, May 2013.

[GJS15] Qian Guo, Thomas Johansson, and Paul Stankovski. Coded-BKW: Solving LWE using lattice codes. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 23–42. Springer, Heidelberg, August 2015.

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

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 321–340. Springer, Heidelberg, December 2010.

[Gt12] Torbjörn Granlund and the GMP development team. *GNU MP: The GNU Multiple Precision Arithmetic Library*, 5.0.5 edition, 2012. http://gmplib.org/.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.

[HILL99] 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.

[HJP13] W. Hart, F. Johansson, and S. Pancratz. FLINT: Fast Library for Number Theory, 2013. Version 2.4.0, http://flintlib.org.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992.

[KW93] M. Karchmer and A. Wigderson. On span programs. In IEEE Computer Society Press, editor, *In Proc. of the 8th IEEE Structure in Complexity Theory*, pages 102–111, Gaithersburg, MD, USA, 1993. IEEE Computer Society Press.

[Lip12] 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. Springer, Heidelberg, March 2012.

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, Heidelberg, February 2011.

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th FOCS*, pages 436–453. IEEE Computer Society Press, November 1994.

[Nao03] Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Heidelberg, August 2003.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[Val08] 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. Springer, Heidelberg, March 2008.