Lattice-Based zk-SNARKs from Square Span Programs

Rosario Gennaro[1], Michele Minelli[2,3], Anca Nitulescu[2,3], and Michele Orrù[2,3]

[1] City College of New York, USA
[2] DIENS, École normale supérieure, CNRS, PSL University, 75005 Paris, France
[3] Inria

rosario@cs.ccny.cuny.edu
{michele.minelli, anca.nitulescu, michele.orru}@ens.fr

Abstract. Zero-knowledge SNARKs (zk-SNARKs) are non-interactive proof systems with short 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 being used for delegating computation, electronic cryptocurrencies, and anonymous credentials. However, all current SNARKs implementations rely on *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 based on lattice assumptions, which are 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: post-quantum; SNARK; zero-knowledge.

1 Introduction

Zero-Knowledge Proof Systems. Proof systems [GMR89] are a fundamental tool in theoretical computer science and cryptography. Consider an NP relation $\mathcal{R}$ which defines the language of all statements x for which there exists a witness w so that $\mathcal{R}(x, w) = \mathbf{true}$. In a zero-knowledge proof for $\mathcal{R}$ a prover, knowing a witness, wants to convince a verifier that x is in the language, without revealing any additional information about the witness.

Since their introduction in [GMR89] zero-knowledge (ZK) proofs have been shown to be a very powerful instrument in the design of secure cryptographic protocols.

For practical applications, researchers immediately recognized two limiting factors in zero-knowledge proofs: the original protocols were interactive and the proof could be as long as (if not longer than) the witness. When considering statistically sound proof systems for NP, unless some complexity-theoretic collapse occurs, the prover P has to communicate, roughly, as much information as the size of the NP witness. Looking for ways to overcome this bound motivated the study of *computationally-sound* proof systems, also called *argument systems* [BCC88].

Non-interactive zero-knowledge (NIZK) proofs [BFM88] and succinct ZK arguments [Kil92, Mic94] were introduced shortly thereafter. Those results were considered mostly theoretical proofs of concept until more recently, when several theoretical and practical breakthroughs [PHGR13, DFGK14] have shown that such proofs (renamed zk-SNARGs for Succinct Non-interactive ARGuments, or zk-SNARKs if the proofs also guarantee that the prover knows the witness w) can indeed be used in practical applications.

Succinct Non-Interactive Arguments of Knowledge. Starting from Kilian's protocol [Kil92], Micali [Mic94] constructed a *one-message* succinct argument for NP whose soundness is set in the

random oracle model. In the plain model, 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 [Gro10, BCCT12, Lip12, BCC+14, GGPR13, BCI+13, PHGR13, BCG+13, BCTV14], 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 for general NP languages (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, computation delegation, confidential transactions, and 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 takes as input 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 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. Their work has led to fast progress towards practical verifiable computations. For instance, using Quadratic Arithmetic Programs (QAPs), a generalization of QSPs for arithmetic circuits, Pinocchio [PHGR13] provides evidence that verified remote computation can be faster than local computation. At the same time, their construction is zero-knowledge, enabling the server to keep intermediate and additional values used in the computation private. Optimized versions of SNARK protocols based on QSPs approach are used in various practical applications, including cryptocurrencies such as Zcash [BCG+14a], to guarantee anonymity while preventing double-spending.

The QSP approach was generalized in [BCI+13] under the concept of *Linear PCP* (LPCP), 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).

So far all known zk-SNARKs rely on "classical" pre-quantum assumptions[4]. Yet, there are widely deployed systems relying on zk-SNARKs (for instance, the Zcash cryptocurrency [BCG+14b]) which are expected not to withstand cryptanalitic efforts over the course of the next 10 years

[4] We note that the original protocol of Kilian [Kil92] is a zk-SNARK which can be instantiated with a post-quantum assumption since it requires only a collision-resistant hash function – however (even in the best optimized version recently proposed in [BSBHR18]) the protocol does not seem to scale well for even moderately complex computations.

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	✓

[ABL+17, Appendix C]. It is an interesting research question, as well our duty as cryptographers, to provide protocols that can guarantee people's privacy over the next decade. We attempt to make a step forward in this direction by building a designated-verifier zk-SNARK from lattice-based (knowledge) assumptions. Our scheme uses as a main building block encodings that rely on the Learning With Errors (LWE) assumption, initially proposed by Regev in 2005 [Reg05], and right now the most widespread post-quantum cryptosystem supported by a theoretical proof of security.

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*[5]. 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

[5] 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.

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 on encodings 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 lattices. We study the relations between our knowledge assumption and the (extractable) linear-only assumption in Appendix A.
– **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.

Technical challenges. Although conceptually similar to the original proof of security for QSP-based SNARKs, our construction must incorporate some additional modifications in order to overcome the noise growth of the LWE-based homomorphic operations. These challenges do not arise in the line of work of Boneh et al. [BISW17, BISW18] due to the more general (and stronger) assumption of linear-only encoding (see Appendix A for details). Additionally, our construction benefits from the optimizations of SSP-based SNARKs [DFGK14].

Instantiating our encoding scheme with a lattice-based scheme like Regev encryption, differs from [GGPR13] and introduces some technicalities, first in the verification step of the protocol, and secondly in the proof of security. Our encoding scheme is additively homomorphic and allows for linear operations; however, correctness of the encoding is guaranteed only for a limited number of homomorphic operations because of the error growth in lattice-based encoding schemes. More specifically, to compute a linear combination of N encodings, we need to scale some parameters for correctness to hold. Throughout this work we will consider only encodings where a bounded number of homomorphic "linear" operations is allowed, and make sure that this bound is sufficient to perform verification and to guarantee the existence of a security reduction.

2 Prerequisites

2.1 Notation

Let $\lambda \in \mathbf{N}$ be the computational security parameter, and $\kappa \in \mathbf{N}$ the statistical security parameter. We say that a function is *negligible* in λ, and we denote it by $\mathsf{negl}(\lambda)$, if it is a $f(\lambda) = o(\lambda^{-c})$ for any fixed constant c. We also say that a probability is *overwhelming* in λ if it is $1 - \mathsf{negl}(\lambda)$. We let $\mathsf{M.rl}(\lambda)$ be a *length function* (i.e., a function $\mathbf{N} \to \mathbf{N}$ polynomially bounded) in λ defining the length of the randomness for a probabilistic interactive Turing Machine M. When sampling uniformly at random the value a from the set S, we employ the notation $a \leftarrow\!\!\$\, S$. When sampling the value a from the probabilistic algorithm M, we employ the notation $a \leftarrow \mathsf{M}$. We use $:=$ to denote assignment. For an n-dimensional column vector $\vec{a}$, we denote its i-th entry by a_i. In the same way, given a polynomial f, we denote its i-th coefficient by f_i. Unless otherwise stated, the norm $\|\cdot\|$ considered in this work is the ℓ_2 norm. 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. u.

Unless otherwise specified, all the algorithms defined throughout this work are assumed to be probabilistic Turing machines that run in time $\mathsf{poly}(\lambda)$ - i.e., PPT. An adversary is denoted by $\mathcal{A}$; when it is interacting with an oracle $\mathcal{O}$, we write $\mathcal{A}^{\mathcal{O}}$. For two PPT machines A, B, with the writing $(\mathsf{A}\|\mathsf{B})(x)$ we denote the execution of A followed by the execution of B on the same input x and with the same random coins. The output of the two machines is concatenated and separated with a semicolon, e.g., $(out_{\mathsf{A}}; out_{\mathsf{B}}) \leftarrow (\mathsf{A}\|\mathsf{B})(x)$.

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 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 $\mathtt{C}(a_1, \ldots, a_\ell) = 1$.

Universal circuit. In the definition, we may see $\mathtt{C}$ as a logical specification of a satisfiability problem. In our zk-SNARK we will split the ℓ inputs into ℓ_u public and ℓ_w private inputs to make it compatible with the universal circuit $C_{\mathcal{U}} : \{0,1\}^{\ell_u} \times \{0,1\}^{\ell_w} \to \{0,1\}$, that take as input an ℓ_u-bit description of a freely chosen circuit $\mathtt{C}$ and an ℓ_w-bit value w, and return 1 if and only if $\mathtt{C}(w) = 1$. Along the lines of [DFGK14], we consider the "public" inputs from the point of view of the prover. For an outsourced computation, they might include both the inputs sent by the clients and the outputs returned by the server performing the computation.

Theorem 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 $p \geq \max(d, 8)$, for $d = m + n$, there exist polynomials $v_0(x), \ldots, v_m(x) \in \mathbf{F}[x]$ and distinct roots $r_0, \ldots, r_{d-1} \in \mathbf{F}$ such that $\mathtt{C}$ is satisfiable if and only if:*

$$\prod_{i=0}^{d-1}(x - r_i) \text{ divides } \left(v_0(x) + \sum_{i=1}^{m} a_i v_i(x)\right)^2 - 1,$$

where $a_1, \ldots, a_m \in \{0,1\}$ correspond to the values on the wires in a satisfying assignment for the circuit.

Define $t(x) := \prod_{i=0}^{d-1}(x - r_i)$, then for any circuit $\mathtt{C} : \{0,1\}^\ell \to \{0,1\}$ of m wires and n gates, there exists a degree $d = m + n$ square span program $\mathsf{ssp} = (v_0(x), \ldots, v_m(x), t(x))$ over a field $\mathbf{F}$, of order $p \geq \max(d, 8)$ that verifies $\mathtt{C}$.

SSP generation. We consider the uniform probabilistic algorithm SSP that, on input a boolean circuit $\mathtt{C} : \{0,1\}^\ell \to \{0,1\}$ of m wires and n gates, chooses a field $\mathbf{F}$, with $|\mathbf{F}| \geq \max(d, 8)$ for $d = m + n$, and samples d random elements $r_0, \ldots, r_d \in \mathbf{F}$ to define the target polynomial $t(x) = \prod_{i=0}^{d-1}(x - r_i)$, together with the set of polynomials $\{v_0(x), \ldots, v_m(x)\}$ composing the SSP corresponding to $\mathtt{C}$.

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

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 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:*

$(\mathsf{crs}, \mathsf{vrs}, \mathsf{td}) \leftarrow \mathsf{G}(1^\lambda, \mathcal{R})$ *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*.*

Game $\mathrm{KSND}_{\Pi,\mathcal{R},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$

$(\mathsf{crs},\mathsf{vrs},\mathsf{td}) \leftarrow \Pi.\mathsf{G}(1^\lambda,\mathcal{R})$
$(u,\pi;w) \leftarrow (\mathcal{A}\|\mathsf{Ext}_{\mathcal{A}})^{\Pi.\mathsf{V}(\mathsf{vrs},\cdot)}(\mathsf{crs})$
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 $\mathrm{ZK}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda)$

$(\mathsf{crs},\mathsf{vrs},\mathsf{td}) \leftarrow \Pi.\mathsf{G}(1^\lambda,\mathcal{R})$
$b \leftarrow\!\!\$\ \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathrm{PROVE}}(\mathsf{vrs})$
return $(b = b')$

Oracle $\mathrm{PROVE}(u,w)$

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

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

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

$\textbf{\textit{bool}} \leftarrow \mathsf{V}(\mathsf{vrs},u,\pi)$ *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 4 (Completeness). *A non-interactive proof system* Π *for the relation* $\mathcal{R}$ *is (computationally)* complete *if for any* PPT *adversary* $\mathcal{A}$*:*

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

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 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:*

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

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 6. 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. While this distinction cannot be made in the publicly-verifiable setting, the same is not true for the designated-verifier setting. In the specific case of our construction, we formulate and prove our protocol allowing the adversary access to the verification algorithm (which has been named *strong soundness* in the past [BISW17]), and later discuss which optimizations can take place when using the weaker notion of soundness, where the adversary cannot access the verification oracle.

A proof system Π for $\mathcal{R}$ is zero-knowledge if no information about the witness is leaked by the proof. More precisely, Π specifies an additional $\mathsf{poly}(\lambda)$ algorithm $\Pi.\mathsf{Sim}$ that takes as input the trapdoor information td and a statement u, and outputs a valid proof π indistinguishable from those generated via $\Pi.\mathsf{P}$.

Definition 7 (Zero-Knowledge). *A non-interactive proof system* Π *is* zero-knowledge *if there for any* PPT *adversary* $\mathcal{A}$*:*

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

where $\mathrm{ZK}_{\Pi,\mathcal{R},\mathcal{A}}(\lambda)$ *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., $|\pi| = \tilde{O}(\lambda)$.

Definition 8 (SNARK). *A* succinct non-interactive argument of knowledge *(SNARK) is a non-interactive proof system that is complete, succinct, and knowledge-sound. A zk-SNARK 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.

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 9 (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 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*.*
  Depending on the encoding algorithm, E *will require either the public information* pk *generated from* K *or the secret state* sk*. For our application, it will be the case of* sk*. To ease notation, we will omit this additional argument.*

The above algorithms must satisfy the following properties:

d**-linearly homomorphic:** *there exists a* $\mathsf{poly}(\lambda)$ *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*.*

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

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

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

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

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

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

Fig. 2. Games for q-PKE, q-PKEQ, q-PDH assumptions.

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 δ, $\mathsf{ct}_0, \ldots, \mathsf{ct}_t$, 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.*

Our specific instantiation of the encoding scheme presents some slight differences with [GGPR13]. In fact, we can allow only for a limited number of homomorphic operations because of the error growth in lattice-based encoding schemes. We note that this modification does not invalidate previous constructions. Sometimes, in order to ease notation, we will employ the writing $\mathsf{ct} := \mathsf{Eval}(\mathsf{E}(a_i)_i, \vec{c}) = \mathsf{E}(t)$, actually meaning that ct is a valid encoding of $t = \sum a_i c_i$; that is, $\mathsf{ct} \in \{\mathsf{E}(t)\}$. It will be clear from the context (and the use of symbol for assignment instead of that for sampling) that the randomized encoding algorithm E is not actually invoked.

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

Throughout this work we rely on a number of computational assumptions. All of them are long-standing assumptions in the frame of dLog-hard groups, and have already been generalized in the scope of "encoding schemes" in [GGPR13]. We recall them here for completeness.

The q-power knowledge of exponent assumption (q-PKE) is a generalization of the knowledge of exponent assumption (KEA) introduced by Damgard [Dam92]. It says that given $\mathsf{E}(s), \ldots, \mathsf{E}(s^q)$ and $\mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^q)$ for some coefficient α, it is difficult to generate $\mathsf{ct}, \widehat{\mathsf{ct}}$ that encode $c, \alpha c$ without knowing the linear combination of the powers of s that produces ct.

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:*

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

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

The q-PDH assumption has been a long-standing, standard q-type assumption [Gro10, BBG05], It basically states that given $\left(\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}\left(s^{q+2}\right), \ldots, \mathsf{E}\left(s^{2q}\right)\right)$, it is hard to compute an encoding of the missing power $\mathsf{E}(s^{q+1})$.

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:*

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

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

Optionally, to achieve strong-soundness (see Remark 6), we need an assumption to be able to "compare" adversarially-generated messages. The q-PKEQ assumptions states that for any adversary $\mathcal{A}$ that outputs two ciphertexts, there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ that can tell whether they encode the same value.

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:*

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

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

Te q-PKEQ assumption is needed solely in the case where the attacker has access to a verification oracle. Since the encoding could be non-deterministic, the simulator in the security reduction of Section 5.2 needs to rely on q-PKEQ to simulate the verification oracle. Pragmatically, this assumption allows us to test for equality of two adversarially-produced encodings without having access to the secret key.

Finally, we recall here a well-known assumption for lattices, that we will use to instantiate our quantum-secure encoding scheme.

Assumption 4 (dLWE)**.**

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

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

where $\text{dLWE}_{\mathsf{Pg},\mathcal{A}}(\lambda)$ *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.

3 An encoding scheme based on Learning With Errors

In this section we give a brief introduction to lattices and we describe a possible encoding scheme based on learning with errors (LWE).

Game $\mathrm{dLWE}_{\mathsf{Pg},\mathcal{A}}(\lambda)$	Oracle ENCODE
$\Gamma \coloneqq (p, q, n, \alpha) \coloneqq \mathsf{Pg}(1^\lambda)$	$\vec{a} \leftarrow\!\!\$\ \mathbf{Z}_q^n$
$\vec{s} \leftarrow\!\!\$\ \mathbf{Z}_q^n$	$e \leftarrow \chi_{q\alpha}$
$b \leftarrow\!\!\$\ \{0,1\}$	**if** $b = 1$ $\quad c \coloneqq \vec{s} \cdot \vec{a} + e$
$b' \leftarrow \mathcal{A}^{\text{ENCODE}}(\Gamma)$	**else** $c \leftarrow\!\!\$\ \mathbf{Z}_q$
return $(b = b')$	**return** $(\vec{a}, c)$

Fig. 3. The decisional LWE problem for parameters Γ.

Lattices. A m-dimensional lattice Λ is a discrete additive subgroup of $\mathbf{R}^m$. For an integer $k < m$ and a rank k matrix $B \in \mathbf{R}^{m \times k}$, $\Lambda(B) = \left\{ B\vec{x} \in \mathbf{R}^m \mid \vec{x} \in \mathbf{Z}^k \right\}$ is the lattice generated by the columns of B.

Gaussian distribution. For any $\sigma \in \mathbf{R}^+$, let $\rho_\sigma(\vec{x}) \coloneqq e^{-\pi \|\vec{x}\|^2/\sigma^2}$ be the Gaussian function over $\mathbf{R}^n$ with mean 0 and parameter σ. For any discrete subset $A \subseteq \mathbf{R}^n$ we define $\rho_\sigma(A) \coloneqq \sum_{\vec{x} \in A} \rho_\sigma(\vec{x})$, the discrete integral of ρ_σ over A. We then define χ_σ, the discrete Gaussian distribution over A with mean 0 and parameter σ as:

$$\chi_\sigma : A \to \mathbf{R}^+ : \vec{y} \mapsto \frac{\rho_\sigma(\vec{y})}{\rho_\sigma(A)}.$$

We denote by χ_σ^n the discrete Gaussian distribution over $\mathbf{R}^n$ where each entry is independently sampled from χ_σ.

3.1 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 $\Gamma \coloneqq (q, n, p, \alpha)$, 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]. The main advantage of this choice is that every entity can (re)compute the description for a given security parameter, and that no single party needs to be trusted with generating the encoding parameters. Moreover, it is often the case that real-world encodings have fixed parameters for some well-known values of λ. For the sake of simplicity, we define our encoding scheme with a LWE encoding description Γ and assume that the security parameter λ can be derived from Γ.

Roughly speaking, the public information is constituted by the LWE parameters Γ and an encoding of m is simply an LWE encryption of m. The LWE secret key constitutes the secret state of the encoding scheme.

3.2 Basic Properties

Correctness.. We say that the encoding scheme is (statistically) *correct* if all valid encodings are decoded successfully (with overwhelming probability).

Definition 10. *An encoding scheme* Enc *is* correct *if, for any* $\vec{s} \leftarrow \mathsf{K}(1^\lambda)$ *and* $m \in \mathbf{Z}_p$,

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

$\mathsf{K}(1^\lambda)$	$\mathsf{E}(\vec{s}, m)$	$\mathsf{D}(\vec{s}, (\vec{c}_0, c_1))$
$\Gamma \coloneqq (p, q, n, \alpha) \coloneqq \mathsf{Pg}(1^\lambda)$	$\Gamma \coloneqq (p, q, n, \alpha) \coloneqq \mathsf{Pg}(1^\lambda)$	$\Gamma \coloneqq (p, q, n, \alpha) \coloneqq \mathsf{Pg}(1^\lambda)$
$\vec{s} \leftarrow_\$ \mathbf{Z}_q^n$	$\vec{a} \leftarrow_\$ \mathbf{Z}_q^n$	**return** $(\vec{c}_0 \cdot \vec{s} + c_1) \pmod p$
return $(\Gamma,\ \vec{s}\,)$	$\sigma \coloneqq q\alpha;\quad e \leftarrow \chi_\sigma$	
	return $(-\vec{a},\ \vec{a} \cdot \vec{s} + pe + m)$	

Fig. 4. An encoding scheme based on LWE.

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 11 (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.. Using the decryption algorithm D, and provided with the secret key (i.e., $\delta \coloneqq \mathsf{sk}$), we can implement image verification. The algorithm $\in$ for image verification proceeds as follows: decrypts the encoded element and tests for equality between the two messages.

Quadratic root detection.. The algorithm Q for quadratic root detection is straightforward using D: decrypt the message and evaluate the polynomial, testing if it is equal to 0.

***d*-linearly homomorphicity..** Given a vector of d encodings $\vec{\mathsf{ct}} \in \mathbf{Z}_q^{d \times (n+1)}$ and a vector of coefficients $\vec{c} \in \mathbf{Z}_p^d$, the homomorphic evaluation algorithm is defined as follows: $\mathsf{Eval}\left(\vec{\mathsf{ct}}, \vec{c}\right) \coloneqq \vec{c} \cdot \vec{\mathsf{ct}}$.

3.3 Technical Challenges

Noise growth.. During the homomorphic evaluation the noise grows as a result of the operations which are performed on the encodings. Consequently, in order to ensure that the output of Eval is a valid encoding of the expected result, we need to start with a sufficiently small noise in each of the initial encodings.

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

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

At this point, this corollary follows:

Corollary 13. *Let* $\vec{s} \leftarrow_\$ \mathbf{Z}_q^n$ *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 12 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 11). 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].

Lemma 14 (Noise Smudging, [Gen09]). *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 $y \hookleftarrow_{\$} [-B_2, B_2]$. Then the distribution of y is statistically indistinguishable from that of $y + x$, as long as $B_1/B_2 = \mathsf{negl}(\kappa)$.*

Proof. Let Δ denote the statistical distance between the two distributions. By its definition:

$$\Delta = \frac{1}{2}\sum_{v=-(B_1+B_2)}^{B_1+B_2} |\Pr[y = v] - \Pr[y = v - x]| = \frac{1}{2}\left(\sum_{v=-(B_1+B_2)}^{-B_2}\frac{1}{B_2} + \sum_{v=B_2}^{B_1+B_2}\frac{1}{B_2}\right) = \frac{B_1}{B_2}.$$

The result follows immediately. □

In order to preserve the correctness of the encoding scheme while allowing linear evaluations, we need once again q to be large enough to accommodate for the flooding noise. In particular, q will have to be at least superpolynomial in the statistical security parameter κ.

Corollary 15. *Let $\vec{s} \in \mathbf{Z}_q^n$ be a secret key and $\vec{m} = (m_1, \ldots, m_d) \in \mathbf{Z}_p^d$ be a vector of messages. Let $\vec{\mathsf{ct}}$ be a vector of d encodings so that $\vec{\mathsf{ct}}_i$ is a valid encoding of m_i, and $\vec{c} \in \mathbf{Z}_p^d$ be a vector of coefficients. Let e_{Eval} be the noise in the encoding output by $\mathsf{Eval}(\vec{\mathsf{ct}}, \vec{c})$ and B_{Eval} a bound on its absolute value. Finally, let $B_{sm} = 2^\kappa B_{\mathsf{Eval}}$, and $e_{sm} \hookleftarrow_{\$} [-B_{sm}, B_{sm}]$. Then the statistical distance between the distribution of e_{sm} and that of $e_{sm} + e_{\mathsf{Eval}}$ is $2^{-\kappa}$. Moreover, if $q > 2p\,B_{\mathsf{Eval}}(2^\kappa + 1)$ then the result of $\mathsf{Eval}(\vec{\mathsf{ct}}, \vec{c}) + (\vec{0}, e_{sm})$ is a valid encoding of $\vec{m} \cdot \vec{c}$ under the secret key $\vec{s}$.*

Proof. The claim on the statistical distance follows immediately from Lemma 14 and the fact that the message part is $\vec{m} \cdot \vec{c}$ is true by simple homomorphic linear operations on the encodings. In order to ensure that the final result is a valid encoding, we need to make sure that the error in this output encoding remains smaller than $q/2p$. The final error is upper bounded by $B_{\mathsf{Eval}} + B_{sm}$, so we have

$$B_{\mathsf{Eval}} + B_{sm} < \frac{q}{2p} \implies B_{\mathsf{Eval}} + 2^\kappa B_{\mathsf{Eval}} < \frac{q}{2p} \implies q > 2p\,B_{\mathsf{Eval}}(2^\kappa + 1).$$

□

Error testing.. By making non-blackbox use of our LWE encoding scheme, it is possible to define an implementation of the function $\mathsf{test\text{-}error}$ 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 $\mathsf{test\text{-}error}$ is displayed in Figure 5.

Procedure $\mathsf{test\text{-}error}(\vec{s}, (\vec{c}_0, c_1))$

$\Gamma \coloneqq (p, q, n, \alpha) \coloneqq \mathsf{Pg}(1^\lambda)$

$e' \coloneqq (\vec{c}_0 \cdot \vec{s} + c_1) \ // \ p$

return (*Eq.* (3))

Fig. 5. The error testing procedure.

Other requirements for security reduction.. The following lemma will be needed later during the security proof. It essentially defines the conditions under which we can take an encoding, add a smudging term to its noise, sum it with the output of an execution of Eval and finally multiply the result by an element in $\mathbf{Z}_p$.

Lemma 16 (For reduction). *Let $\vec{s}$, $\vec{\mathsf{ct}}$, $\vec{c}$, e_{Eval}, B_{Eval} be as in Corollary 15, and let $\mathsf{ct}' = (-\vec{a}', \vec{s} \cdot \vec{a}' + pe' + m')$ be a valid encoding of a message $m' \in \mathbf{Z}_p$ with noise e' bounded by B_e. Let $B_{sm} = 2^\kappa B_e$ and $e_{sm} \leftarrow\!\!\$\ [-B_{sm}, B_{sm}]$ be a "smudging noise". Then, if $q > 2p^2\left(\left(2^\kappa + 1\right) B_e + B_{\mathsf{Eval}}\right)$, it is possible to add the smudging term e_{sm} to ct', sum the result with the output of $\mathsf{Eval}\left(\vec{\mathsf{ct}}, \vec{c}\right)$, multiply the outcome by a coefficient bounded by p, and obtain a valid encoding of $k\left(\vec{m} \cdot \vec{c} + m'\right)$.*

Proof. The correctness of the message part comes immediately from performing homomorphic linear operations on encodings, and the final output is valid if the noise remains below a certain threshold. After adding the smudging term and performing the sum, the noise term is at most $B_e + B_{sm} + B_{\mathsf{Eval}} = \left(2^\kappa + 1\right) B_e + B_{\mathsf{Eval}}$. After the multiplication by a coefficient bounded by p, it is at most $p\left(\left(2^\kappa + 1\right) B_e + B_{\mathsf{Eval}}\right)$. Thus, the encoding is valid if:

$$p\left(\left(2^\kappa + 1\right) B_e + B_{\mathsf{Eval}}\right) < \frac{q}{2p}, \tag{2}$$

which immediately gives the result. □

Conditions on the modulus q. Corollaries 13 and 15 and Lemma 16 give the conditions that the modulus q has to respect in order to allow for all the necessary computations. In particular, Corollary 13 gives the condition to be able to homomorphically evaluate a linear combination of fresh encodings through the algorithm Eval; Corollary 15 gives the condition to be able to add a smudging noise to the result of such an evaluation; Lemma 16 gives a condition that will have to be satisfied in the security reduction. They are ordered from the least stringent to the most stringent, so the condition that must be satisfied in the end is the one given by Lemma 16:

$$q > 2p^2\left(\left(2^\kappa + 1\right) B_e + B_{\mathsf{Eval}}\right) \tag{3}$$

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 17 (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 18 (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., $\forall a \neq b \in \{0,1\}^n$,*

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

Then, for any random variable Y,

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

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

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. (4)

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

return $(\mathsf{vrs}, \mathsf{crs}, \mathsf{td})$

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

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

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

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

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

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

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

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

return $\mathsf{test\text{-}error}(\mathsf{sk}, B_w)$

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

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

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

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

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

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

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

// Compute the proof terms as per Eq. (6)

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

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

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

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

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

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

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

Fig. 6. Our zk-SNARK protocol Π.

Zero-Knowledge.. We now present a version of the LHL that will be useful later in this work, when proving the zero-knowledge property of our construction. In a nutshell, it says that a random linear combination of the columns of a matrix is statistically close to a uniformly random vector, for some particular choice of coefficients.

Lemma 19 ("Specialized" leftover hash lemma). *Let n, p, q, d be non-negative integers. Let $\mathbf{A} \leftarrow_\$ \mathbf{Z}_q^{n \times d}$, and $\vec{r} \leftarrow_\$ \mathbf{Z}_p^d$. 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 $\vec{u} \leftarrow_\$ \mathbf{Z}_q^n$.

Proof. For the vector $\vec{r}$, we have that $H_\infty(\vec{r}) = d \log p$. Then the proof is immediate from Lemma 18:

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

□

4 Our designated-verifier zk-SNARK

Let Enc be an encoding scheme (Definition 9). Let $\mathtt{C}$ be some circuit taking as input an ℓ_u-bit string and outputting 0 or 1. Let $\ell \coloneqq \ell_u + \ell_w$, 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 $\mathtt{C}$ and n to the number of fan-in 2 gates. Let $d \coloneqq m + n$. We construct a zk-SNARK scheme for any relation $\mathcal{R}_\mathtt{C}$ on pairs $(u, w) \in \{0,1\}^{\ell_u} \times \{0,1\}^{\ell_w}$ that can be computed by a polynomial size circuit $\mathtt{C}$ 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 $\mathtt{C} : \{0,1\}^{\ell_u} \times \{0,1\}^{\ell_w} \to \{0,1\}$. It generates a square span program of degree $d = m + n$ over a field $\mathbf{F}$, of size $|\mathbf{F}| \geq d$ that verifies $\mathtt{C}$ by running:

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

Then, it runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda)$ using the encoding scheme Enc. Finally, it samples $\alpha, \beta, s \leftarrow \mathbf{F}$ such that $t(s) \neq 0$, and returns the CRS:

$$\begin{aligned}\mathsf{crs} \coloneqq \Big(\mathsf{ssp},\ \mathsf{pk},\ & \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^d), \\ & \mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^d), \\ & \mathsf{E}(\beta t(s)), (\mathsf{E}(\beta v_i(s)))_{i=\ell_u+1}^{m} \Big)\end{aligned} \tag{4}$$

The error for each of these encodings has to be chosen carefully. In a nutshell, we need to intentionally increase the magnitude of the noise in some encodings, in order to mimic its distribution in the simulated CRS provided to the adversary in the security reduction. Failing to do so results in limiting the adversary's ability to perform homomorphic operations on the CRS and, thus, in a flawed proof. We defer further analysis on this point to Section 6. The verification string vrs, as well as the trapdoor td, consists of the secret key sk of the encoding scheme, and the secrets s, α, β (we implicitly include the crs in vrs).

Prover. The prover algorithm, on input some statement $u \coloneqq (a_1, \ldots, a_{\ell_u})$, computes a witness $w \coloneqq (a_{\ell_u+1}, \ldots, a_m)$ 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. Then, it samples $\gamma \leftarrow_{\$} \mathbf{F}$ and sets $\nu(x) \coloneqq v_0(x) + \sum_{i=1}^{m} a_i v_i(x) + \gamma t(x)$. Let:

$$h(x) \coloneqq \frac{(v_0(x) + \sum_i^m a_i v_i(x) + \gamma t(x))^2 - 1}{t(x)} = \frac{\nu(x)^2 - 1}{t(x)}, \tag{5}$$

whose coefficients can be computed from the polynomials provided in the ssp; them by linear evaluation it is possible to obtain:

$$\begin{aligned} H &\coloneqq \mathsf{E}(h(s)), \qquad \widehat{H} \coloneqq \mathsf{E}(\alpha h(s)), \qquad \widehat{V} \coloneqq \mathsf{E}\left(\alpha \nu(s)\right), \\ V_w &\coloneqq \mathsf{E}\left(\sum_{i=\ell_u+1}^{m} a_i v_i(s) + \gamma t(s)\right), \\ B_w &\coloneqq \mathsf{E}\left(\beta\left(\sum_{i=\ell_u+1}^{m} a_i v_i(s) + \gamma t(s)\right)\right). \end{aligned} \tag{6}$$

In fact, the encoding 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 (5). 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 $\pi \coloneqq (H, \widehat{H}, \widehat{V}, V_w, B_w)$.

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 (that implicitly contains the crs), proceeds with the following verifications. First, it uses the quadratic root detection algorithm of the encoding scheme Enc to verify that the proof satisfies:

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

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

$$b_s - \beta w_s = 0. \tag{eq-lin}$$

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

Then, the verifier checks whether it is still possible to perform some homomorphic operations, using the test-error procedure, implemented in Figure 5 for the specific case of lattice encodings. More precisely, the verifier tests whether it is still possible to add another encoding and multiply the result by an element bounded by p, without compromising the correctness of the encoded element. This will guarantee the existence of a reduction in the knowledge soundness proof of Section 5.2. If all above checks hold, the verifier returns **true**. Otherwise, return **false**.

Remark 20. Instantiating our encoding scheme on top of a "noisy" encryption scheme like Regev's introduces multiple technicalities that affect the protocol, the security proof, and the parameters' choice. For instance, in order to compute a linear combination of d encodings via Eval we need to scale down the error parameter and consequently increase the parameters q and n in order to maintain correctness and security. Similarly, the distributions of the error terms and the random vectors are affected by the homomorphic evaluation, and we must guarantee that the resulting terms are still simulatable. All these issues will be formally addressed in Section 5, and then analyzed more pragmatically in Section 6.

5 Proofs of security

In this section, we prove our main theorem:

Theorem 21. *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 13 states the conditions on Γ for which the homomorphically computed encodings are valid with probability at least $1 - \mathsf{negl}(\kappa)$. Lemma 16 affirms that correctly generated proofs satisfy Equation (2) with probability overwhelming in κ. Therefore test-error returns true and completeness follows trivially by Theorem 2. □

5.1 Zero-Knowledge

In order to obtain a zero-knowledge protocol, we perform smudging of the proofs terms, and we randomize the target polynomial $t(x)$. The first step hides the witness, the second makes the distribution of the final noise independent from the coefficient a_i. 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 19).

Proof (of zero-knowledge). The simulator for zero-knowledge is shown in Figure 7. The error are independently sampled from the same uniform distribution over the (integer) interval $[-2^\kappa T\sigma_{B_w},\ 2^\kappa T\sigma_{B_w}]$, where T is a small constant and $\sigma_{B_w} := p\sigma\sqrt{d+1}\sqrt{p^2+m-\ell_u}$. We will call this the *smudging distribution.*

Checking that the proof output by Π.Sim is indeed correct (i.e., that it verifies Eqs. (eq-pke) to (eq-lin)) is trivial. We are left with showing that the two proofs are statistically indistinguishable.

Note that once the value of V_w in the proof has been fixed, the verification equations uniquely determine $H, \widehat{H}, \widehat{V}$, and B_w. This means that for any (u, w) such that $\mathsf{C}(u, w) = 1$, both the real arguments and the simulated arguments are chosen uniformly at random such that the verification equations will be satisfied. One can prove that values for V_w are statistically indistinguishable when executing Π.P and Π.Sim: V_w is the encoding of a uniformly random variable γ_w in Π.Sim and the masking of a polynomial evaluation by adding $\gamma t(s)$, where γ is chosen uniformly at random (note that $t(s) \neq 0$) in Π.P. What is encoded in the remaining terms is simply dictated by the verification constraints.

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

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

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

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

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

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

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

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

Fig. 7. Simulator for Zero-Knowledge.

In both worlds, the proof is a tuple of 5 encodings $(H, \widehat{H}, \widehat{V}, V_w, B_w)$. Once the vrs is fixed, each encoding can be written as $(-\vec{a}, \vec{a} \cdot \vec{s} + pe + m)$, for some $\vec{a} \in \mathbf{Z}_q^n$ and some $m \in \mathbf{Z}_p$ satisfying the verification equations. Due to Lemma 18, the random vectors $\vec{a}$ are indistinguishable from uniformly random in both worlds. The error terms are statistically indistinguishable due to Lemma 14. (See Section 6 for a detailed explanation of these values.) □

Zero-knowledge comes at a cost: smudging the error terms requires us to scale the ciphertext modulus by κ bits. For those applications where zero-knowledge is not required, we can simplify the protocol by removing $\gamma t(x)$ from the computation of $h(x)$ and avoiding the smudging procedure on every proof term. In Table 1 we show some choices of parameters, both with and without zero-knowledge.

5.2 Knowledge Soundness

Before diving into the technical details of the proof of 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 for requiring 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, by soundness of the square span program (Theorem 2), 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. The 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 zero. Then, to generate the values in the crs it sets $\beta := \beta(s)$ (which can be computed from its input consisting of encodings of powers of s). Using the above, it runs the SNARK adversary and to obtain from its output B_w an encoding of some polynomial with coefficient s^{q+1} non-zero 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.

Proof (of computational knowledge soundness). Let $\mathcal{A}^{\Pi}$ be the PPT adversary in the game for knowledge soundness (Figure 1) able to produce a proof π for which $\Pi.\mathsf{V}$ returned **true**. We first claim that it is possible to extract the coefficients of the polynomial $v(x)$ corresponding to the values v_s encoded in V. The setup algorithm first generates the parameters $(\mathsf{pk}, \mathsf{sk})$ of an encoding scheme Enc and picks $\alpha, \beta, s \in \mathbf{F}$, which are used to compute $\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^d), \mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^d)$. Fix some circuit C, and let ssp be an SSP for C. Let $\mathcal{A}^{\mathrm{PKE}}$ be the d-PKE adversary, that takes as input a set of encodings:

$$\sigma := \left(\mathsf{pk}, \mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^d), \mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^d)\right).$$

The auxiliary input generator Z is the PPT machine that upon receiving as input σ, samples $\beta \leftarrow_{\$} \mathbf{Z}_p$, constructs the remaining terms of the CRS (as per Equation (4)), and outputs them in z using ssp. Thus, $\mathcal{A}^{\mathrm{PKE}}$ sets $\mathsf{crs} := (\mathsf{ssp} \| \sigma \| z)$ and invokes $\mathcal{A}^{\Pi}(\mathsf{crs})$. As a result, it obtains a proof $\pi = (H, \widehat{H}, \widehat{V}, V_w, B_w)$. On this proof, it computes:

$$V := \mathsf{E}\left(v_0 + \sum_{i=1}^{\ell_u} a_i v_i(s) + w_s\right) = V_w + v_0 + \sum_{i=1}^{\ell_u} a_i v_i(s). \tag{7}$$

where w_s is the element encoded in V_w. Finally, $\mathcal{A}^{\mathrm{PKE}}$ returns $(\widehat{V}, V)$. If the adversary $\mathcal{A}$ outputs a valid proof, then by verification equation Eq. (eq-pke) it holds that the two encodings $(V, \widehat{V})$ encode values $v_s, \widehat{v}_s$ such that $\widehat{v}_s - \alpha v_s = 0$. Therefore, by q-PKE assumption there exists an extractor $\mathsf{Ext}^{\mathrm{PKE}}$ that, using the same input (and random coins) of $\mathcal{A}^{\mathrm{PKE}}$, outputs a vector $(c_0, \ldots, c_d) \in \mathbf{F}^{d+1}$ such that V is an encoding of $\sum_{i=0}^{d} c_i s^i$ and $\widehat{V}$ is an encoding of $\sum_{i=0}^{d} \alpha c_i s^i$. In the same way, it is possible to recover the coefficients of the polynomial $h(x)$ used to construct $(H, \widehat{H})$, the first two elements of the proof of $\mathcal{A}^{\Pi}$ (again, by Eq. (eq-pke)).

Our witness-extractor Ext^{Π}, given crs, emulates the extractor $\mathsf{Ext}^{\mathrm{PKE}}$ above on the same input σ, using as auxiliary information z the rest of the CRS given as input to Ext^{Π}. By the reasoning discussed above, Ext^{Π} can recover $(c_0, \ldots, c_d)$ coefficients extracted from the encodings $(V, \widehat{V})$. Consider now the polynomial $v(x) := \sum_{i=0}^{d} c_i x^i$. If it is possible to write the polynomial as $v(x) = v_0(x) + \sum_{i}^{m} a_i v_i(x) + \delta t(x)$ such that $(a_1, \ldots, a_m) \in \{0,1\}^m$ satisfies the assignment for the circuit C with $u = (a_1, \ldots, a_{\ell_u})$, then the extractor returns the witness $w = (a_{\ell_u+1}, \ldots, a_m)$.

With overwhelming probability, the extracted polynomial $v(x) := \sum_{i=0}^{d} c_i x^i$ does indeed provide a valid witness w. Otherwise, there exists a reduction to q-PDH that uses the SNARK adversary $\mathcal{A}^{\Pi}$. Define the polynomial

$$v_{\mathsf{mid}}(x) := v(x) - v_0(x) - \sum_{i=1}^{\ell_u} a_i v_i(x).$$

We know by definition of SSP (Definition 1) and by Theorem 2 that C is satisfiable if and only if

$$t(x) \mid v^2(x) - 1 \ \wedge \ v_{\mathsf{mid}}(x) = \sum_{i}^{d} c_i x^i - v_0(x) - \sum_{i}^{\ell_u} a_i v_i(x) \in \mathrm{Span}(v_{\ell_u+1}, \ldots, v_m, t)$$

Therefore, by contradiction, if the adversary $\mathcal{A}^{\Pi}$ does not know a witness $w \in \{0,1\}^{m-\ell_u}$ for u (such that $(u, w) \in \mathcal{R}_{\mathsf{C}}$), but still the two verification checks Eq. (eq-div) and Eq. (eq-lin) pass, we have that either one of the following two cases must hold:

i. $t(x)h(x) \neq v^2(x) - 1$, but $t(s)h(s) = v^2(s) - 1$; or

ii. $v_{\mathsf{mid}}(x) \notin \mathrm{Span}(v_{\ell_u+1}, \ldots, v_m, t)$, but B_w is an encoding of $\beta v_{\mathsf{mid}}(s)$.

Let $\mathcal{B}^{\mathrm{PDH}}$ be an adversary against the q-PDH assumption. Given a q-PDH challenge

$$\left(\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}\left(s^{q+2}\right), \ldots, \mathsf{E}\left(s^{2q}\right)\right), \quad \text{for } q \in \{2d-1, d\}$$

adversary $\mathcal{B}^{\mathrm{PDH}}$ samples uniformly at random $\alpha \leftarrow_{\$} \mathbf{F}$, and defines some $\beta \in \mathbf{F}$ (that we will formally construct later) and constructs a CRS as per Equation (4). There are some subtleties in how $\mathcal{B}^{\mathrm{PDH}}$ generates the value β. In fact, β can be generated without knowing its value explicitly, but rather knowing its representation over the power basis $\left\{s^i\right\}_{i=0, i\neq q+1}^{2q}$ – that is, knowing a polynomial $\beta(x)$ and its evaluation in s. Some particular choices of β will allow us to provide a solution for a q-PDH challenge. $\mathcal{B}^{\mathrm{PDH}}$ invokes the adversary $\mathcal{A}^{\Pi}$ as well as the extractor Ext^{Π} on the generated CRS, thus obtaining a proof π and the linear combination used by the prover for the polynomials $h(x), v(x)$ and also extracts a witness for the statement being proved.

For the *strong soundness* (see Remark 6), in order to simulate the verification oracle and to answer the verification queries of $\mathcal{A}^{\Pi}$, $\mathcal{B}^{\mathrm{PDH}}$ has to compare its encodings (obtained from the extracted coefficients and its input) with $\mathcal{A}$'s proof terms, accepts if the terms match, and rejects otherwise. Because the encoding scheme is not deterministic, adversary $\mathcal{B}^{\mathrm{PDH}}$ invokes the PKEQ extractor and simulates the verification oracle correctly with overwhelming probability.

The reduction in the two mentioned cases works as follows:

i. The extracted polynomials $h(x)$ and $v(x)$ satisfy $t(s)h(s) = v^2(s) - 1$, but $t(x)h(x) \neq v^2(x) - 1$. By q-PDH assumption this can happen only with negligible probability. We define $p(x) = v^2(x) - 1 - t(x)h(x)$, that in this case is a non-zero polynomial of degree $k \leq 2d$ having s as a root. Let p_k be the highest nonzero coefficient of $p(x)$. Write $\tilde{p}(x) = x^k - p_k^{-1} \cdot p(x)$. Since s is a root of $x^k - \tilde{p}(x)$, it is a root of $x^{q+1} - x^{q+1-k}\tilde{p}(x)$. $\mathcal{B}^{\mathrm{PDH}}$ solves q-PDH by computing $\mathsf{E}(s^{q+1}) = \mathsf{E}(s^{q+1-k}\tilde{p}(s))$ for $q = 2d-1$. Since $\deg(\tilde{p}) \leq k-1$, the latter is a known linear combination of encodings $\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q)$ which are available from the q-PDH challenge. More precisely, $\mathcal{B}^{\mathrm{PDH}}$ will compute $\mathsf{Eval}((\mathsf{E}(s^{i+q+1-k}))_i, (\tilde{p}_i)_i^{2d-1})$ on *fresh* encodings $\mathsf{E}(1), \mathsf{E}(s), \mathsf{E}(s^2), \ldots, \mathsf{E}(s^q)$ solving the q-PDH challenge for $q \geq 2d-1$.
ii. In the second case, suppose that the polynomial v_{mid} extracted as previously described cannot be expressed as a linear combination of $\{v_{\ell_u+1}, \ldots, v_m, t\}$. The proof still passes the verification, so we have a consistent value for $B_w \in \{\mathsf{E}(\beta v_{\mathsf{mid}}(s))\}$.
$\mathcal{B}^{\mathrm{PDH}}$ generates a uniformly random polynomial $a(x)$ of degree q subject to the constraint that all of the polynomials $a(x)t(x)$ and $\{a(x)v_i(x)\}_{i=\ell_u+1}^{m}$ have coefficient 0 for x^{q+1}. We note that for $q = d$, there are $q - (m - \ell_u) > 0$ degrees of freedom in choosing $a(x)$.
$\mathcal{B}^{\mathrm{PDH}}$ defines β to be the evaluation of $a(x)$ in s, i.e., $\beta \coloneqq a(s)$. Remark that $\mathcal{B}^{\mathrm{PDH}}$ does not know s explicitly, but having access to the encodings of $2q-1$ powers of s, it is able to generate valid encodings $(\mathsf{E}(\beta v_i(s)))_i$ and $\mathsf{E}(\beta t(s))$ using Eval. Note that, by construction of β, this evaluation is of $d+1$ elements in $\mathbf{F}$ and that the $(q+1)$-th power of s is never used. Now, since $v_{\mathsf{mid}}(x)$ is not in the proper span, the coefficient of degree $q+1$ of $xa(x)v_{\mathsf{mid}}(x)$ must be nonzero with overwhelming probability $1 - 1/|\mathbf{F}|$. The term B_w of the proof must encode a known polynomial in s: $\sum_{i=0}^{2q} b_i s^i \coloneqq \beta v_{\mathsf{mid}}(s) = a(s)v_{\mathsf{mid}}(s)$ where the coefficient b_{q+1} is non-trivial. $\mathcal{B}^{\mathrm{PDH}}$ can subtract off encodings of multiples of other powers of s to recover $\mathsf{E}(s^{q+1})$ and break q-PDH. This requires an evaluation on *fresh* encodings:

$$\mathsf{Eval}\left((\mathsf{E}(s^i))_{\substack{i=0 \\ i\neq q+1}}^{q+d}, (-b_i)_{\substack{i=0 \\ i\neq q+1}}^{q+d}\right). \tag{8}$$

Adding the above to B_w and multiplying by the inverse of the $(q+1)$-th coefficient (using once again Eval) will provide a solution to the q-PDH problem for $q = d$.

Since the two cases above are not possible by q-PDH assumption, Ext^{Π} extracts a valid witness if the proof of $\mathcal{A}^{\Pi}$ is valid. □

As previously mentioned in Remark 6, 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. 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. 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[6] of 32 bits for the size of the message space.

The CRS is composed of encodings of different nature: some of them are fresh $(\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^d))$, some happen to be stale in the construction of $\mathcal{A}^{\mathrm{PKE}}$ and the construction of $\mathcal{B}^{\mathrm{PDH}}$ Section 5.2 (Item i.) $(\mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^d))$, and some are stale from the construction of $\mathcal{B}^{\mathrm{PDH}}$ Section 5.2 (Item ii.) $(\mathsf{E}(\beta t(s)), (\mathsf{E}(\beta v_i(s)))_i)$. They are displayed in Figure 8. Since, as we have seen, $\mathcal{B}^{\mathrm{PDH}}$ manipulates the q-PDH challenge via homomorphic operations, we must guarantee that the protocol adversary can perform *at least* the same number of homomorphic operations as in the real-world protocol. Therefore, in the real protocol, we must intentionally increase the magnitude of the noise in the CRS: the terms $\mathsf{E}(\alpha s^i)$ (with $i = 0, \ldots, d$) are generated by multiplying the respective *fresh* encoding $\mathsf{E}(s^i)$ by a term bounded by p; the terms $\mathsf{E}(\beta t(s)), \{\mathsf{E}(\beta v_i(s))\}_i$ instead are generated via Eval of $d+1$ elements with coefficients bounded by p. Concretely, when encoding these elements using the encoding scheme of Section 3, the error for $\mathsf{E}(\alpha s^i)$ is sampled from $p \cdot \chi_\sigma$; the error for $\mathsf{E}(\beta t(s)), \mathsf{E}(\beta v_i(s)))$ is sampled from $(p\sqrt{d+1}) \cdot \chi_\sigma$.

The proof π consists of five elements $(H, \widehat{H}, \widehat{V}, V_w, B_w)$, as per Equation (6). H and V_w are computed using an affine function on d encodings with coefficients modulo p; $\widehat{H}, \widehat{V}$ are computed using a linear function on $d+1$ encodings with coefficients modulo p; finally, B_w is computed using a linear combination of $m - \ell_u$ encodings with coefficients in $\{0, 1\}$, except the last one which is modulo p. Overall, the term that carries the highest load of homomorphic computations is B_w. The generation of B_w is outlined in Figure 8, and to it (as well as to the other proof terms) we add a smudging term for constructing a zero-knowledge proof π.

In the construction of the adversary $\mathcal{B}^{\mathrm{PDH}}$ (Item ii.) we need to perform some further homomorphic operations on the proof element B_w in order to solve the q-PDH challenge, namely one addition (Equation (8)) and one multiplication by a known scalar b bounded by p. The result of the first operation is denoted by $\mathsf{E}(b \cdot s^{q+1})$ in Figure 8; the final result is the solution to the q-PDH challenge.

[6] In particular, we need p and q to be relatively prime for the correctness of the encoding scheme [BV11, footnote 18].

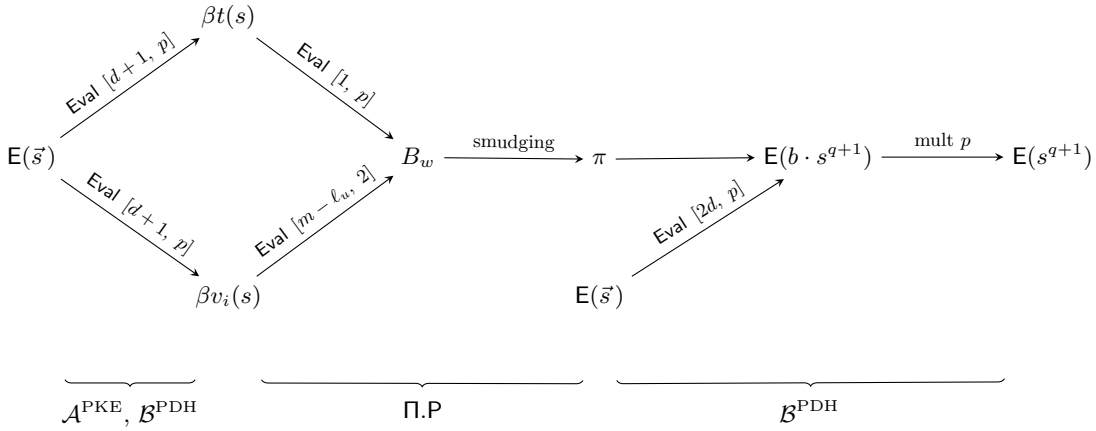

Fig. 8. Summary of evaluations in the security proof. The leftmost part of the figure refers to the construction of adversaries for q-PKE and q-PDH; the central part refers to the protocol itself (i.e., the construction of the proof π); the rightmost part refers to the construction of the adversary for q-PDH (Section 5.2 - Item ii.). The syntax $\mathsf{Eval}\,[d, p]$ denotes a homomorphic evaluation on d encodings with coefficients in $\mathbf{Z}_p$. $\mathsf{E}(\vec{s})$ denotes the PDH challenge.

We now outline the calculations that we use to choose the relevant parameters for our encoding scheme. In particular, we will focus on the term B_w since, as already stated, it is the one that is involved in the largest number of homomorphic operations. The chain of operations that need to be supported is depicted in Figure 8: we now analyze them one by one. The correctness of the other terms follows directly from Corollary 13.

First of all, the terms $(\beta v_i(s))_i$ and $\beta t(s)$ are produced through the algorithm Eval executed on $d+1$ *fresh* encodings with coefficients modulo p. Let σ be the discrete Gaussian parameter of the noise terms in fresh encodings; then, by Pythagorean additivity, the Gaussian parameter of encodings output by this homomorphic evaluation is $\sigma_{\mathsf{Eval}} \coloneqq p\sigma\sqrt{d+1}$. Then the term $\beta t(s)$ is multiplied by a coefficient in $\mathbf{Z}_p$, and the result is added to a subset sum of the terms $(\beta v_i(s))_i$, i.e., a weighted sum with coefficients in $\{0, 1\}$. It is trivial to see that, for the first term, the resulting Gaussian parameter is bounded by $p\sigma_{\mathsf{Eval}}$, whereas for the second term it is bounded by $\sigma_{\mathsf{Eval}}\sqrt{m-\ell_u}$. The parameter of the sum of these two terms is then bounded by $\sigma_{B_w} \coloneqq \sigma_{\mathsf{Eval}}\sqrt{p^2+m-\ell_u}$. Let us then consider a constant factor T for "cutting the Gaussian tails", i.e., such that the probability of sampling from the distribution and obtaining a value with magnitude larger than T times the standard deviation is as small as desired. We can then write that the absolute value of the error in B_w is bounded by $T\sigma_{B_w}$. At this point we add a *smudging* term, which amounts to multiplying the norm of the noise by $(2^\kappa + 1)$ (cf. Corollary 15). Finally, the so-obtained encoding has to be summed with the output of an Eval invoked on $2d$ fresh encodings with coefficients modulo p and multiplied by a constant in $\mathbf{Z}_p$. The final noise is then bounded by $Tp\sigma_{B_w}(2^\kappa+1) + Tp\sigma_{\mathsf{Eval}}$ (cf. Lemma 16). By substituting the values of σ_{Eval}, σ_{B_w}, remembering that $\sigma \coloneqq \alpha q$ and imposing the condition for having a valid encoding, we obtain

$$Tp^2\alpha q\sqrt{d+1}\left(\sqrt{p^2+m-\ell_u}\,(2^\kappa+1)+1\right) < \frac{q}{2p}.$$

The above corresponds to Equation (3) with bounds $B_e \coloneqq T\sigma_{B_w}$ and $B_{\mathsf{Eval}} \coloneqq T\sigma_{\mathsf{Eval}}$. By simplifying q and isolating α, we get:

$$\alpha < \frac{1}{2Tp^3\sqrt{d+1}\left(\sqrt{p^2+m-\ell_u}\,(2^\kappa+1)+1\right)}.$$

With our choice of parameters and by taking $T = 8$, we can select for instance $\alpha = 2^{-180}$.

Table 2. Comparison with previous works. PQ stands for post-quantum. We note that the construction of [PHGR13] is very different (namely, based on elliptic curves), and comparing security levels is therefore difficult.

	PQ	λ	ZK	$\lvert\pi\rvert$	$\lvert\mathsf{crs}\rvert$	d
[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	32,767

Once α and p are chosen, we select the remaining parameters q and n in order to achieve the desired level of security for the LWE encoding scheme. To do so, we take advantage of Albrecht's estimator[7] [APS15] which, as of now, covers the following attacks: meet-in-the-middle exhaustive search, coded-BKW [GJS15], dual-lattice attack and small/sparse secret variant [Alb17], lattice reduction with enumeration [LP11], primal attack via uSVP [AFG14, BG14], Arora-Ge algorithm [AG11] using Gröbner bases [ACFP14]. Some possible choices of parameters are reported in Table 1.

Finally, based on these parameters, we can concretely compute the size of the CRS[8] and that of the proof π. The CRS is composed of $d + (d+1) + (m+1)$ encodings, corresponding to the encodings of the d powers of s, the encodings of α multiplied by the $d+1$ powers of s, the m encodings of $(\beta v_i)_i$, and the encoding of $\beta t(s)$. This amounts to $(2d+m+2)$ LWE encodings, each of which has size $(n+1)\log q$ bits[9]. For the calculations, we bound m by d and state that the size of the CRS is that of $(3d+2)$ LWE encodings. From an implementation point of view, we can consider LWE encodings $(\vec{a}, b) \in \mathbf{Z}_q^{n+1}$ where the vector $\vec{a}$ is the output of a seeded PRG. This has been proven secure in the random oracle model [Gal13]. Therefore, the communication complexity is greatly reduced, as sending an LWE encoding just amounts to sending the seed for the PRG and the value $b \in \mathbf{Z}_q$. For security to hold, we can take the size of the seed to be λ bits, thus obtaining the final size of the CRS: $(3d+2)\log q + \lambda$ bits. The proof π is composed of 5 LWE encodings, therefore it has size $|\pi| = 5(n+1)\log q$ bits. Note that in this case we cannot trivially use the same trick with the PRG, since the encodings are produced through homomorphic evaluations.

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[10]. 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 Appendix A.

Table 3. Benchmarks of our proof system (zk) for different circuit sizes (i.e., d).

Circ. size	Setup (s)	Prover (s)	Verifier (ms)
2^{10}	1.46 s ± 18.7 ms	1.61 s ± 27.8 ms	1.26 ms ± 16 µs
2^{13}	12.3 s ± 37.9 ms	13 s ± 224 ms	1.50 ms ± 16 µs
2^{15}	57.8 s ± 134 ms	53.6 s ± 247 ms	2.28 ms ± 17 µs
2^{16}	167 s ± 269 ms	235 s ± 451 ms	3.46 ms ± 17 µs

6.1 Implementation

We implemented our construction in standard C11, using the library GMP [Gt12] for handling arbitrary precision integers and the library FLINT [HJP13] for handling polynomials. We chose the pseudo-Mersenne prime $p := 2^{32} - 5$, and a the modulus $q := 2^{736}$. This allows for fast arithmetic operations: reduction modulo q simply consists in a bitmask, modular operations by p can fit a `uint64_t` type, and multiplication of a scalar modulo p to a vector in $\mathbf{Z}_q^{n+1}$ does not require any memory allocation for the carry. The dimension of the lattice was chosen $n = 1470$, corresponding to the "medium" security level displayed in Table 1.

We performed extensive benchmarks of our protocol on a single thread of an Intel Core i7-4720HQ CPU @ 2.60GHz, running Arch Linux (kernel version 4.14.39). Our implementation is publicly available [11]. Time is measured using `gettimeofday(2)`. Encoding a uniformly random element of $\mathbf{Z}_p$ using $\mathsf{Enc.E}$ takes on average 310 µs (std. dev. 34 µs); decoding it using $\mathsf{Enc.D}$ is about the same order of magnitude, 197 µs (std. dev. 24 µs). Measurements were done over 100,000 samples. The algorithm for homomorphic evaluations Eval is able to compute a linear combination of 2^{15} ciphertexts with coefficients modulo p in roughly 13 s, and of 2^{18} in about 94 s.

For proving satisfiability of a boolean circuit with roughly 2^{14} gates (i.e., $d = 2^{15}$), we measured roughly one minute for the CRS generation algorithm; 53 s for the prover; 2.28 ms for the verifier (on average, over 100 repeated executions varying SSPs). This is about one order of magnitude slower w.r.t. Pinocchio's benchmarks [PHGR13, Fig. 8]; verification instead is one order of magnitude faster. More detailed benchmarks for different circuit sizes can be found in Table 3.

Acknowledgments. We would like to thank Balthazar Bauer and Florian Bourse for insightful discussions. 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).

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.

[7] https://bitbucket.org/malb/lwe-estimator

[8] We take into account only the encodings that are contained in the CRS. The other terms have considerably smaller impact on its size or can be agreed upon offline (e.g., the SSP).

[9] Note that the magnitude of the noise term, i.e., whether the encoding is fresh or stale, has no impact on the size of an encoding. This size is a function only of n (the number of elements in the vector) and the modulus q.

[10] Results are extracted from the source code at https://github.com/dwu4/lattice-snarg.

[11] See https://www.di.ens.fr/~orru/pq-zk-snarks.

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.

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.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

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.

$\underline{\text{Game EXT-LO}_{\mathsf{Enc},\mathsf{M},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)}$

$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{K}(1^\lambda, 1^d)$

$(m_1,\ldots,m_d) \leftarrow \mathsf{M}(1^\lambda, 1^d)$

$\sigma \leftarrow (\mathsf{E}(m_1),\ldots,\mathsf{E}(m_d))$

$(\mathsf{ct}; a_0,\ldots,a_d) \leftarrow (\mathcal{A}\|\mathsf{Ext}_{\mathcal{A}})(\sigma)$

$\textbf{return } \mathsf{ct} \notin \left\{\mathsf{E}(a_0 + \sum_{i=1}^{d} a_i m_i)\right\}$

Fig. 9. Game for Extractable Linear-Only target malleability.

Appendix A Lattices and Assumptions

At a first glimpse, it might seem unjustified to have brought assumptions often used in the dLog setting into the lattice domain, where they are highly non-standard. Despite this fact, in this section we argue (i) that the q-PKE and q-PDH assumptions are weaker than the targeted linear-only malleability of [BCI+13, BISW17], and (ii) which consequences an attack on those assumptions would have.

Over the course of the last years, a long line of research in lattice-based cryptography has been trying to develop fully-homomorphic encryption schemes and bilinear pairing maps. So far, no bilinear map is known in the context of lattices, and some have argued that its existence would lead to efficient cryptographic primitives such as multilinear maps and indistinguishability obfuscation (iO). Furthermore, although there exist FHE schemes based on lattices, it is not clear how to achieve it without giving away additional information such as encryption of the secret key itself. Showing that it is possible to indeed compute non-linear homomorphisms on top of Regev's encryption scheme would be a major breakthrough in both research areas. We see this as a win-win situation.

Moreover, our assumptions are *weaker* than previously employed assumptions for lattice-based SNARGs. Indeed, the linear-only assumption of [BCI+13, BISW17] informally states that an adversary can only perform affine operations over the encodings provided as input. More specifically:

Definition 22 (EXT-LO, [BCI+13]). *An encoding scheme* Enc *satisfies* extractable linear-only target malleability *if for all* PPT *adversaries* $\mathcal{A}$ *and plaintext generation algorithm* M *there exists an efficient extractor* Ext *such that the advantage*

$$\mathsf{Adv}^{\text{ext-lo}}_{\mathsf{Enc},\mathsf{M},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) := \Pr\left[\text{EXT-LO}_{\mathsf{Enc},\mathsf{M},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda) = \textbf{\textit{true}}\right]$$

is $\mathsf{negl}(\lambda)$*, where* $\text{EXT-LO}_{\mathsf{Enc},\mathsf{M},\mathcal{A},\mathsf{Ext}_{\mathcal{A}}}(\lambda)$ *is defined as in Figure 9.*

We note that, despite [BCI+13] presents the above assumption for so-called *linear-only encryption schemes*, all such schemes are also encodings satisfying the properties of Definition 9.

It is not immediately clear to see what does this assumption imply in the case of LWE encodings (like the one we presented in section Section 3) or the one in [LP11], used in [BISW17]. Consider for example a set of parameters Γ allowing for $d-1$ homomorphic operations modulo p and the adversary $\mathcal{A}$ that, upon receiving as input d ciphertexts, computes d homomorphic linear operations on them. With non-negligible probability the error wraps around the modular representation, leading to a "decryptable" encoding (any element of $\mathbf{Z}_q^{n+1}$ is a valid encoding) but for which the adversary does not know an affine map. The authors of [BISW17] suggest to use *double-encryption* in these cases, i.e., present two different encodings of each value, and ask the adversary to homomorphically evaluate these terms twice. If the two ciphertexts do not encode the same element, the game is lost. Obviously, this comes at the cost of doubling the size of each encoding and doubling the computation time for the prover and the verifier.

Theorem 23. *If* Enc *is an* IND-CPA *extractable linear-only encoding scheme, it satisfies* q-PDH.

Proof. Let us consider an adversary $\mathcal{A}^{\text{PDH}}$ for the q-PDH assumption. We show that there exists an adversary able to break IND-CPA.

Consider the PPT machine $\mathcal{A}$ that samples uniformly at random two field elements, s_0 and s_1, then submits the two distinct chosen plaintexts s_0^{q+1}, s_1^{q+1} to the IND-CPA challenger. $\mathcal{A}$ queries the IND-CPA encoding oracle with s_0^k, s_1^k for $k = 0, \ldots, q, q+2, \ldots, 2q$. The oracle gives back some encodings $\sigma \coloneqq \left(\mathsf{E}(1), \mathsf{E}(s_b), \ldots, \mathsf{E}(s_b^q), \mathsf{E}(s_b^{q+2}), \ldots, \mathsf{E}(s_b^{2q})\right)$ for $b \in \{0,1\}$. $\mathcal{A}$ runs the q-PDH adversary on σ, thus obtaining (with non-negligible probability) some encoding $\mathsf{ct} \in \left\{\mathsf{E}(s_b^{q+1})\right\}$. By EXT-LO, there exists an extractor $\mathsf{Ext}^{\mathsf{LO}}$ which, given as input σ and the same random coins of the adversary $\mathcal{A}^{\text{PDH}}$, returns a polynomial p such that $p(s_b) = s_b^{q+1}$. Let $f(x) \coloneqq p(x) - x^{q+1}$. By q-PDH, $f(s_b) = 0$; by Schwartz-Zippel lemma, $f(s_{1-b}) \neq 0$ with probability $1 - 2q/|\mathbf{F}| = 1 - \mathsf{negl}(\lambda)$. $\mathcal{A}$ returns the bit b^* such that $f(s_{b^*}) = 0$, thus solving the IND-CPA challenge with overwhelming probability. □

Theorem 24. *If* Enc *is an* IND-CPA *extractable linear-only encoding scheme, it satisfies* q-PKE.

Proof. We will show that Enc satisfies q-PKE, meaning there is no $\mathcal{A}^{\text{PKE}} \| \mathsf{Ext}_{\mathcal{A}}$ able to win the q-PKE game (cf. Figure 2).

Suppose by contradiction that there exists an adversary $\mathcal{A}^{\text{PKE}}$ able to produce a valid output $\mathsf{ct}, \widehat{\mathsf{ct}}$, i.e., such that $\alpha\mathsf{ct} - \widehat{\mathsf{ct}} = 0$. We show that as a consequence there exists an extractor $\mathsf{Ext}_{\mathcal{A}}$ that outputs the correct linear combination with non negligible probability.

Let M be the plaintext generation algorithm that, upon receiving the computational security parameter λ and $d = 2q + 2$ in unary form, samples $s \leftarrow_{\$} \mathbf{F}$ and outputs plaintexts $1, s, \ldots, s^q$. Let $\sigma \leftarrow (\mathsf{E}(1), \mathsf{E}(s), \ldots, \mathsf{E}(s^q), \mathsf{E}(\alpha), \mathsf{E}(\alpha s), \ldots, \mathsf{E}(\alpha s^q))$. The adversary $\mathcal{A}^{\text{PKE}}$, when run on this input σ, outputs (with non-negligible probability) $\mathsf{ct}, \widehat{\mathsf{ct}}$ such that $\alpha\mathsf{ct} - \widehat{\mathsf{ct}} = 0$ (via quadratic root detection algorithm).

Let us define the adversaries $\mathcal{B}_0$ and $\mathcal{B}_1$ for the game EXT-LO that, upon receiving as input σ, run the same instantiation of $\mathcal{A}^{\text{PKE}}$ and output ct - respectively $\widehat{\mathsf{ct}}$. By our claim of linear-only property, there exist the extractors Ext_0 and Ext_1 for $\mathcal{B}_0$ and $\mathcal{B}_1$, respectively, outputting $a_0, \ldots, a_q, b_0, \ldots, b_q$ and $a'_0, \ldots, a'_q, b'_0, \ldots, b'_q$ such that

$$\mathsf{ct} \in \left\{\mathsf{E}\left(\sum_i^d a_i s^i + \sum_i^d b_i \alpha s^i\right)\right\}, \quad \widehat{\mathsf{ct}} \in \left\{\mathsf{E}\left(\sum_i^d a'_i s^i + \sum_i^d b'_i \alpha s^i\right)\right\}$$

with non negligible probability.

Knowing that $\alpha\mathsf{ct} - \widehat{\mathsf{ct}} = 0$ implies either that the polynomial

$$P(X,Y) = X^2 \sum_i^d b_i Y^i + X \sum_i^d (a_i - b'_i) Y^i - \sum_i^d a'_i Y^i$$

is the zero polynomial, or that (α, s) are roots of $P(X,Y)$. The second case is ruled out by semantic security of the encoding scheme and Schwartz-Zippel lemma, by a reasoning similar to the proof of Theorem 23.

The case where $P(X,Y) = 0$ gives us $b_i = a'_i = 0, a_i = b'_i \forall i = 0, \ldots, q$. Therefore, we are able to define an extractor $\mathsf{Ext}_{\mathcal{A}}$ for q-PKE that outputs the coefficients a_i of the linear combination with non-negligible probability, showing that any successful adversary against q-PKE able to output $\mathsf{ct}, \widehat{\mathsf{ct}}$ such that $\alpha\mathsf{ct} - \widehat{\mathsf{ct}} \in \{\mathsf{E}(0)\}$, has knowledge of the coefficients a_i such that $\mathsf{ct} \in \left\{\mathsf{E}\left(\sum_i^d a_i s^i\right)\right\}$. □