Amortized Efficient zk-SNARK from Linear-Only RLWE Encodings

Heewon Chung, Dongwoo Kim, Jeong Han Kim, and Jiseung Kim

***Abstract*—This paper addresses a new lattice-based designated zk-SNARK having the smallest proof size in the amortized sense, from the linear-only ring learning with the error (RLWE) encodings. We first generalize a quadratic arithmetic programming (QAP) over a finite field to a ring-variant over a polynomial ring $\mathbb{Z}_p[X]/(X^N+1)$ with a power of two N. Then, we propose a zk-SNARK over this ring with a linear-only encoding assumption on RLWE encodings. From the ring isomorphism $\mathbb{Z}_p[X]/(X^N+1) \cong \mathbb{Z}_p^N$, the proposed scheme packs multiple messages from $\mathbb{Z}_p$, resulting in much smaller amortized proof size compared to previous works.**

In addition, we present a refined analysis on the noise flooding technique based on the Hellinger divergence instead of the conventional statistical distance, which reduces the size of a proof. In particular, our proof size is 276.5 KB and the amortized proof size is only 156 bytes since our protocol allows to batch N proofs into a single proof. Therefore, we achieve the smallest amortized proof size in the category of lattice-based zk-SNARKs and comparable proof size in the (pre-quantum) zk-SNARKs category.

***Index Terms*—Post-quantum cryptography, RLWE, SNARK, zero-knowledge proofs.**

I. INTRODUCTION

A ZERO-knowledge proof is a protocol that enables a prover to convince a verifier of knowledge of witness without any unnecessary leakage of the witness [1]. Specifically, zero-knowledge succinct non-interactive argument of knowledge (zk-SNARKs) is, literally, a zero-knowledge proof which is one-round protocol whose proof size is small. Since its introduction, zk-SNARKs have drawn vast attention due to their versatility and diverse applications including cryptocurrency [2]–[4], deep learning [5] and database queries [6]. In addition, there is an attempt to standardize zero-knowledge proofs, named ZKProof [7], to apply them to the industry, and many famous companies such as Google and Microsoft take part in this workshop.

Most constructions proposed so far mainly depend on *pre-quantum* primitives and hardness assumptions such as pairing [8], [9], hidden order group [10], and discrete logarithm problem [11], [12]. There exist hash-based constructions [13]–[15] which are secure under the quantum computer, however, it takes relatively high verification cost and storage cost than group-based constructions since they contain many hash function iterations. On the other hand, lattice-based constructions have been proposed as promising candidates for post-quantum zk-SNARKs. However, the proposed lattice-based constructions are inefficient compared to group-based constructions [8]–[12] in all aspects, especially, in the proof size: while the group-based scheme [9] has 131 bytes of proof (with BN-128 curve), the best one [16] among lattice-based (designated) SNARKs requires 270 kilobytes of proof.

Here, we pay attention that the best one [16] and all other zk-SNARKs (except [17], [18]) from lattices exploit encoding schemes based on learning with errors (LWE) problem. However, several lattice-based constructions [19], [20] have shown that lattice hard problems with algebraic structures such as Ring LWE (RLWE) [21] or NTRU [22] have mathematical structures with which one can improve the efficiency of schemes. Therefore, replacing LWE by RLWE (in scheme construction) is one of the widely used techniques for improving the efficiency of a lattice-based encryption scheme, and we can raise the following natural and meaningful question:

Is it possible to enhance the efficiency of the lattice-based zk-SNARK with hard problems from algebraic lattices?

Related Work. Coming to the quantum revolution, building post-quantum zk-SNARKs with lattice-based cryptographic primitives has been highlighted as one of the challenging problems in the area of cryptography and security. We review some previous work [16]–[18], [23], [24] proposing designated verifier zk-SNARKs based on lattices.

Boneh *et al.* [17], [18] proposed the first lattice-based SNARG from linear-only encoding assumption on the encryption scheme based on (R)LWE problem. Specifically, the latter achieved the quasi-optimality in prover's cost via the linear-only vector encryption scheme over rings and linear PCP [25] with multiple provers. While those work provide the best asymptotic cost among others, authors left the construction of a zk-SNARK from lattices as an open problem.

On the other hand, Gennaro *et al.* [23] proposed zk-SNARK with square span program (SSP) [26] assuming that an encoding scheme from LWE problem also satisfies the similar classical hardness assumptions — q-power knowledge exponent (PKE), q-power Diffie Hellman (PDH) — from finite groups (previously, those assumptions are exploited to

Manuscript received August 19, 2021 revised January 27, 2023; approved for publication by Jung Hee Cheon, Division 1 Editor, March 19, 2023.

H. Chung is with DESILO Inc., Seoul, Republic of Korea, email: heewon.chung@desilo.ai.

D. Kim is with the Department of AI·SW Convergence, Dongguk University, Seoul, Republic of Korea, email: Dongwoo.Kim@dgu.edu.

J.-H. Kim is with the School of Computational Sciences, Korea Institute for Advanced Study (KIAS), Seoul, Republic of Korea, email: jhkim@kias.re.kr.

J. Kim is with the Department of Computer Science and Artificial Intelligence/CAIIT, Jeonbuk National University, Jeonju, Republic of Korea, email: jiseungkim@jbnu.ac.kr.

H. Chung and D. Kim contributed equally to this paper.

J. Kim is the corresponding author.

Digital Object Identifier: 10.23919/JCN.2023.000012

construct zk-SNARKs from pairing groups, e.g., [27], [28]). Similarly, Nitulescu [16] presented a lattice-based zk-SNARG with square arithmetic programs (SAP) assuming that an encoding scheme from LWE problem satisfies the 'linear-targeted malleability' assumption which is a slightly weaker assumption than the linear-only assumption. It also has the advantage that the size of the proof is smaller than the aforementioned lattice-based zk-SNARKs, with the proof π consisting of only two LWE encodings. Recently, Naganuma *et al.* [24] also proposed, via similar approach as above, a lattice-based zk-SNARK from quadratic arithmetic programs (QAP) and then compared their result to the previous work [16], [23] with implementation. As expected in theory, while SAP-based zk-SNARK [16] has smaller proof size and less verification cost, their QAP-based one [24] is better in other aspects: setup time, prover's cost, and the size of common reference strings. A concurrent and independent work [29] proposed a ring-variant of Pinocchio, named Rinocchio based on the quadratic ring programs (QRP) similarly as our ring-QAP. However, their construction is focused on SNARK which does not provide zero-knowledge property. For more details about the differences, we refer to Section III-B.

All of those works, including ours, provide zk-SNARKs with designated verifiers (i.e., the verification requires a private verification key) only, and constructing a publicly verifiable zk-SNARK from lattice is still an open problem.

A. Our Approach

We propose a new lattice-based zk-SNARK from RLWE problem, linear-only encoding assumption over this ring, and the notion of ring-QAP. Moreover, we provide a tight analysis on conventional noise flooding technique to reduce the size of RLWE encodings based on the Hellinger distance and due to this analysis, we can reduce not only the size of a proof in amortized sense but also the size of a single encoding.

Previously, only an LWE-based encoding scheme was exploited [16], [24] to construct zk-SNARK from lattices. To enhance the efficiency by leveraging the ring structures, we extend QAPs over $\mathbb{Z}_p$ to a ring-QAPs over a polynomial ring $R_p = \mathbb{Z}_p[X]/(X^N+1)$ with the generalized Schwartz-Zippel Lemma over R_p then employ an RLWE-based encoding scheme having an element of ring R_p as a message. It gives a zk-SNARK for arithmetic circuits over a ring R_p, to which one can apply the traditional message packing method, then we significantly reduce the proof size in amortized sense.

More precisely, when N is a power of 2 and $p = 1 \bmod 2N$, R_p is isomorphic to $\mathbb{Z}_p^N$, and a single ring element has one-to-one correspondence with an N dimension vector over $\mathbb{Z}_p$, which enables to pack multiple field elements to one ring element. Then, we can outsource N computations to an untrusted prover and reduce the computational complexity of the prover and the verifier as well as the proof size in the amortized sense.

In addition, to shorten the proof size, we provide a new analysis on the parameters of zk-SNARKs using the Hellinger distance rather than the statistical distance from the previous construction. For zero-knowledgeness, all conventional lattice-based zk-SNARKs [16], [23], [24] from square span programs (SSPs), SAPs, and QAPs must exploit a noise flooding technique to hide the error term in final encodings which will be disclosed to a verifier. In other words, for the error term $\mathbf{e}$ given in the final encodings, we must guarantee that no one can distinguish $\mathbf{e}+D$ from D where D is a certain distribution. To this end, previous work chose D as a uniform distribution on a large interval and employed the statistical distance as a measure to show the closeness of the above two distributions.

Unfortunately, the previous analysis with the statistical distance — providing a rough upper bound on adversaries success probability — requires that if $\kappa \approx \lambda$, where λ is the security parameter and κ is the -log of statistical distance between two distributions. In contrast, the closeness derived from Hellinger distance provides more tight analysis on the success probability of adversaries on (decision) security game, thereby requiring relaxed requirement, e.g., $\kappa' \approx \lambda/2$ where κ' is -log of Hellinger distance.

As a result, our protocol can also reduce the size of a single encoding in both asymptotic and concrete settings. Specifically, the size of single proof is about 276.5 KB when $\lambda = 110$, which is much smaller than that of the previous work in the lattice-based zk-SNARKs [16], [23], [24]. In addition, under 128 bit security parameter, our proof size is 156 bytes with an amortization cost and it is comparable to 138 bytes of Groth [9], the shortest proof size among all zk-SNARKs.

Concurrent Works. There are two independent and concurrent works[1] that improves the lattice-based SNARKs.

Ganesh *et al.* [29] propose a new SNARK (without ZK property) called Rinocchio for general ring arithmetic computations. To satisfy the soundness in the ring setting, they also employ the generalized Schwartz-Zippel lemma (Lemma 4) and the Ring-LWE encodings against quantum adversaries. On the other hand, Rinocchio is slightly different from our ring-QAP based zk-SNARK, similarly as the Pinocchio [27] is different from Groth's work [9]. For example, Rinocchio requires 9 RLWE encodings to describe the proof of arguments, but the proof of our zk-SNARK only consists of 3 RLWE encodings. In addition, [29] uses q-PDH and q-PKE assumptions over rings that are weaker than 'linear-only' encoding assumption that we use. While their work focused on the generality, we focus on better efficiency exploiting specific case with a ring of the form $\mathbb{Z}_p^n$. Furthermore, we provide a tighter analysis on noise flooding technique for zero-knowledgeness (while [29] does not) as will be described in the following subsection. Our analysis could be applicable to Rinocchio for building a lattice based zk-SNARK with a shorter proof.

Another work by Yuval Ishai *et al.* [30] also proposes zk-SNARKs for reducing the proof size. Their construction is built on Bitansky *et al.* [25] compiler with linear-only vector encryption suggested by Boneh *et al.* [17]. To minimize the proof size, they employ several methods including modulus switching[2] on the proof encoding, exploiting a linear PCPs and vector encryptions on quadratic extension field (of a base

[1]The first version of our draft was submitted in Feb 9 2021 while Rinocchio was published in ePrint in 10 Mar 2021; [30] was published during the review period.

[2]A widely employed technique in fully homomorphic encryption to reduce the modulus of a ciphertext without modifying the underlying messages.

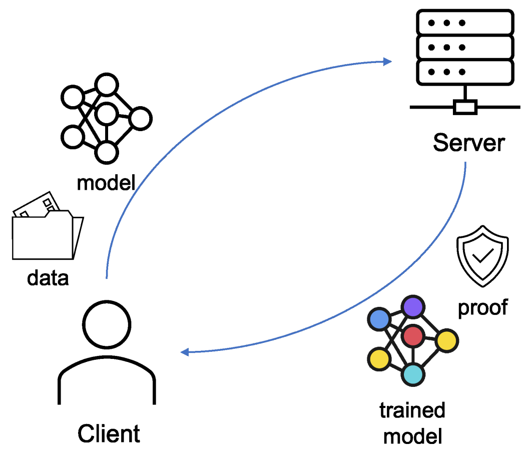

Fig. 1. Verifying ML training phase with zk-SNARK.

finite field $\mathbb{Z}_p$), etc. While they can achieve the smallest proof size among all zk-SNARKs based on lattice assumption, their construction does not support batching multiple proofs in contrasts to ours. As a quick comparison, our construction utilizes an extension *ring* $R_p = \mathbb{Z}_p[X]/(X^N+1)$ with high degree N targeting the smallest *amortized* proof size while their optimization utilizes quadratic extension fields $\mathbb{Z}_p[X]/(X^2+1)$ to reduce the single proof size. Thus, our proposal still remains the lattice-based zk-SNARKs having the smallest proof size in amortized sense.

B. Application — Verifiable Machine Learning Training

As an interesting application scenario of our proposed zk-SNARK, we present the verification of machine learning (ML) training. The ML training phase is composed of many computation steps where the portion of input data is used to update the model parameters. Assume that a client outsources to a server a training phase of ML model with data to be trained on. However, since the training phase is composed of many steps of computations on large data, a server may miss some portion of training steps and/or the training data. Therefore, both a client and a server have an incentive to verify and prove that the final output model is trained correctly with the given data. This is possible with zk-SNARKs where the client and the server act as a verifier and a prover, respectively, by generating and verifying the proof of training computation; see Fig. 1.

While one can use any zk-SNARKs in this scenario, our zk-SNARK — with reduced amortized proof size — can provide smaller overall proof size than the previous work. For detail, assume that the training phase is composed of many training steps each of which can be described as follows:

$$f_i(\vec{W}_i, D_i) = \vec{W}_{i+1},$$

where $\vec{W}_i$ and $\vec{W}_{i+1}$ are the model parameters before and after the i-th step f_i, respectively, and D_i is the (portion of) data used in each step. Then, the entire training phase can be verified by verifying every zk-SNARK proof for each step f_i[3] given that the prover sends all intermediate $\vec{W}_i$'s along with the proof to the verifier; hence, it requires CRS for the computation of each f_i's only (which will be identical in most cases). In contrasts, if we consider proving the entire training phase with initial and final parameters only, it requires problematically large amount of CRS requirement due to the huge size of entire computation. In the former case with affordable CRS size, our zk-SNARK whose proof is capable of proving and verifying many computations simultaneously provides much smaller proof size than the previous zk-SNARKs. Specifically, if there are n training steps, previous zk-SNARKs require n proofs while ours does only $\lceil n/N \rceil$ proofs where N is the maximum proof capability of ours in one proof encoding. Note that the verifier has all intermediate $\vec{W}_i$'s and arranges them correctly as input and output of parallel circuits then verifies correct computation of all f_i's with the zk-SNARK proof.

Note, in addition, that the number n of training steps are usually bigger or comparable to the amortization capability $N = 2,048$ in our zk-SNARK construction, realizing the best possible amortized proof size in most cases. On the other hand, the server may not want to disclose some of hyper-parameters — the external values used to control the learning process, e.g., learning rate, batch size, number of iterations, etc. — since it comprises a secret know-how for getting good training results. This can be kept secret by zero-knowledge property of zk-SNARK.

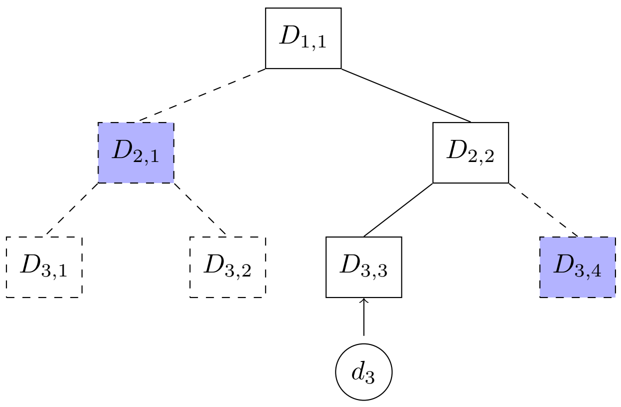

Fig. 2. A toy example of verifying Merkle proofs.

C. Application — Merkle Proof with Smaller Proof Size

Merkle trees, proposed by Ralph Merkle [31], is a binary tree in which the leaf node is a cryptographic hash of a data block, and non-leaf node is a cryptographic hash value of its child nodes. This technique is used to prove efficiently that some data block received from the other belongs to the tree. Therefore, Merkle trees are widely used in many applications, especially, peer-to-peer systems such as Git and BitTorrent and recently, many cryptocurrencies also employ Merkle trees to verify the data block received from other nodes.

[3]In usual ML training f_i's are almost the same for all steps. Our zk-SNARK can also handle different f_i's given that the circuit size of them are bounded.

For Merkle proofs, the prover provides a sequence of hash value needed to compute the hash value of its parent from the leaf node to the root node. Then, the verifier climbs Merkle trees and ensure the validity of the proof when the computed root hash value coincides with the public root value. To be more concrete, given a data D belonging to a binary tree of depth ℓ, the Merkle proof for D is $\pi = (\pi_1, \cdots, \pi_\ell)$, where π_i is the hash value for level i and π_1 is the publicly known Merkle root. For a cryptographic hash function H, $h_\ell = H(D)$, and for each $i \in \{2, 3, \cdots, \ell\}$, the verifier goes to level $i-1$ nodes from level i nodes by checking if h_{i-1} is $H(\pi_i, h_i)$ or $H(h_i, \pi_i)$ depending on the Merkle path of D to the root. Lastly, he can obtain h_1 and accepts the proof only if $h_1 = \pi_1$. In other words, the verification circuit can be represented by multiple evaluations of the same hash function H as follows: for $i \in \{2, 3, \cdots, \ell\}$,

$$h_{i-1} = \begin{cases} H(\pi_i, h_i), & \text{if level } i \text{ node is connected} \\ & \quad\quad \text{to a right leaf in level } i-1, \\ H(h_i, \pi_i), & \text{otherwise.} \end{cases}$$

Now, for the application of zk-SNARK, we assume, similarly as the previous application example, that the prover sends all π_i's, h_i's, and the information of the Merkle path along with a SNARK proof to a verifier. Then, the verifier arranges each input for each evaluation appropriately (as π_i, h_i or h_i, π_i) and verifies above computation with the SNARK proof. Since the verifier has the intermediate hash values, this process enables the verifier to check the dependency between levels. In this case, with our zk-SNARK, the size of proof can be reduced considerably since we need only $\lceil \ell/N \rceil$ proofs while previous one requires ℓ proofs. Ours will be also beneficial if one needs to prove/verify many Merkle proofs simultaneously. Moreover, with zk-SNARK, the computational complexity for the verifier can be less than that for the original Merkle verification where she needs to evaluate many hash functions.

We tried to implement Merkle proof to prove the efficiency. Even though `libsnark` provides a gadget for SHA-256, it does not fit in `SEAL` library. For this reason, we generate a random circuit with 2^{15} multiplicative gates and 2^5 the number of inputs instead. As far as we know, a circuit representing SHA-256 also has about 2^{15} multiplicative gates. Our experiment result is summarized in Table III in Section V. According to our implementation, the verification time is fast enough. Moreover, a proof contains many independent instances and it implies that it is beneficial to prove many instances at the same time such as Merkle proof. Additionally, our scheme is designed based on Groth's zk-SNARK scheme [9] whose one of key features is that complexity for the verifier is independent on the circuit. In this context, we believe that the current experiment indicates that verification using zk-SNARK is useful when the number of hash functions is sufficiently large.

D. Organization

Section II provides preliminaries about discrete Gaussian distributions, definitions of RLWE and zk-SNARK. Section III provides our main protocol, zk-SNARK from RLWE, which consists of ring-QAP, zk-SNARK from RLWE, security proofs and better noise flooding using Rènyi divergence. Section IV provides the size of the proof with various security parameters and comparison between ours and the previous work.

II. Preliminaries

Let $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ be the integers, rational, real, respectively, and $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ the set of integers modulo q represented as integers from $(-q/2, q/2] \cap \mathbb{Z}$, and $\mathbb{Z}[X]$ be the set of all polynomials with integer coefficients. Throughout this paper, we denote N for a power of two integer so that $X^N + 1$ is a cyclotomic polynomial, $R = \mathbb{Z}[X]/(X^N+1)$ and $R_q = R/qR = \mathbb{Z}_q[X]/(X^N+1)$. We use left-arrow notations in the following two cases: For a finite set S, $s \leftarrow S$ denotes that s is uniformly sampled from S. For a distribution $\mathcal{D}$, $s \leftarrow \mathcal{D}$ denotes that s is sampled from the distribution $\mathcal{D}$. A statistical distance between two discrete distributions D_1 and D_2, denoted by $\mathrm{SD}(D_1, D_2)$, is $\sum_{x \in X} \frac{1}{2} \Pr |D_1(x) - D_2(x)|$. We denote $\Pr[s \leftarrow \mathcal{D} \,|\, A]$ as the probability that an event A occurs when $s \leftarrow \mathcal{D}$.

A. Lattices and Discrete Gaussian Distribution

A lattice $\mathcal{L}$ is defined as an additive discrete subgroup of $\mathbb{R}^n$ and is represented by integral linear combinations of a basis $\mathbf{B} \in \mathbb{R}^{n \times r}$, i.e., $\mathcal{L} = \mathbf{B}\mathbb{Z}^r$. We first recall the definition and some properties of a Discrete Gaussian distribution over a lattice $\mathcal{L}$.

Definition 1 (Discrete Gaussian Distribution): Let $\mathcal{L}$ be a lattice contained in $\mathbb{R}^n$. Then, for any positive real number σ, we define a function ρ as follows.

$$\rho_\sigma(\mathbf{x}) = \exp(-\pi\|\mathbf{x}\|^2/2\sigma^2).$$

Then, we define a discrete Gaussian distribution $\chi_{\mathcal{L},\sigma}$ with a standard deviation σ whose probability density function is $\rho_\sigma(\mathbf{x})/\rho_\sigma(\mathcal{L})$ where $\rho_\sigma(\mathcal{L})$ is the sum of all points $\mathbf{x} \in \mathcal{L}$, *i.e.*, $\rho_\sigma(\mathcal{L}) = \sum_{\mathbf{x} \in \mathcal{L}} \rho_\sigma(\mathbf{x})$.

Lemma 1 (Tail Bounds): For $\sigma > 0$ and $T > 0$, it holds that

$$\Pr[e \leftarrow \chi_\sigma \ : \ |e| > T\sigma] \le \frac{2}{T\sqrt{2\pi}} \exp(-T^2/2).$$

Lemma 2 ([32], Tail Bounds of Inner Product): For $\sigma > 0$ and $T > 0$, it holds that

$$\Pr[\mathbf{e} \leftarrow \chi_\sigma^n \ : \ \langle \mathbf{e}, \mathbf{c} \rangle \ge T\sigma\|\mathbf{c}\|] < 2\exp(\pi T^2).$$

B. Linear-Only Encoding Scheme from RLWE

We introduce an encoding scheme which is a building block of our zk-SNARK construction. The encoding scheme has the ring R_p and R_q as the message space and the encoding space, respectively for some integers p and q.

Definition 2 (RLWE Encoding Scheme): Our RLWE Encoding scheme is composed of three algorithms KeyGen, Enc, Dec as follows: ($\Delta = \lfloor q/p \rfloor$, χ_σ denotes the discrete Gaussian distribution with a standard deviation σ)

- $\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$: Sample $\mathsf{s} \leftarrow R_q$. Output $\mathsf{sk} = \mathsf{s}$ as a secret key.

- $\mathsf{Enc}(\mathsf{sk}, \mathbf{m}) \to \mathsf{ct}$: To encrypt $\mathbf{m} \in R_p$, sample $\mathbf{a} \leftarrow R_q$ and $\mathbf{e} \leftarrow \chi_\sigma^N$ for a discrete Gaussian distribution χ_σ. Then, compute $\mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e} + \Delta \mathbf{m} \bmod q$, and output a ciphertext $\mathsf{ct} := (\mathbf{a}, \mathbf{b})$.
- $\mathsf{Eval}(\boldsymbol{c}, d, \{\mathsf{ct}\}_{i=1}^I) \to \mathsf{ct}$: For a given vector $\boldsymbol{c} \in R_p^I$ and I ciphertexts $\mathsf{ct}_i = (a_i, b_i)$, output the ciphertext $\mathsf{ct} := (\boldsymbol{c} \cdot (a_1, a_2, \cdots, a_I), \boldsymbol{c} \cdot (b_1, b_2, \cdots, b_I) + \Delta d)$
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mathbf{m}$: To decrypt $\mathsf{ct} = (\mathbf{a}, \mathbf{b})$, compute $\mathbf{d} = \mathbf{b} - \mathbf{as} \bmod q$ and output $\lfloor \frac{p\mathbf{d}}{q} \rceil \bmod p$

This encoding scheme is indeed a symmetric key encryption from [33] and is semantically secure under the RLWE assumption (Definition 4 given below). We also remark that an addition of and a scalar multiplication on ciphertexts *homomorphically* correspond to those operations on the underlying messages, or more precisely: for all $(m_i)_{i \in I} \in R_p^I$, $\boldsymbol{c} \in R_p^I$, and $d \in R_p$, the following probability is bigger than $1 - \mathsf{negl}(\lambda)$:

$$\Pr\left[\begin{matrix}\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \{\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, m_i)\}_{i=1}^I\end{matrix} \middle| \begin{matrix}\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\boldsymbol{c}, d, \{\mathsf{ct}_i\}_{i=1}^I)) \\ = \boldsymbol{c} \cdot (m_1, m_2, \cdots, m_I) + d\end{matrix}\right],$$

given that $pI\|\mathbf{e}\| < q/2p$. This allows a party (without sk) to output a ciphertext whose underlying message is an affine combination of the underlying messages of given ciphertexts.

Essential to our construction of zk-SNARK is the assumption that above encoding scheme is a *linear-only* encoding scheme. Roughly, it assumes that the *only* way for a PPT adversary to generate a valid new ciphertext is linearly combining the given ciphertexts. The formal definition is as follows, and a generalized version of this assumption (with messages composed of vectors) was also exploited in [17], [18] to construct SNARG from (R)LWE assumptions.

Definition 3 (Linear-Only Encoding [17], [25]): Fix a security parameter λ. An encoding scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ over a ring $\mathfrak{R}$ is a linear-only encoding scheme if for any PPT adversary $\mathcal{A}$, there exists an efficient extractor $\mathcal{X}_\mathcal{A}$ such that for all auxiliary inputs $z \in \{0,1\}^\lambda$, and any plaintext generation algorithm $\mathcal{M}$ (which outputs some elements from $\mathfrak{R}$), we have that for $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $(a_1, a_2, \cdots, a_m) \leftarrow \mathcal{M}(1^\lambda)$, $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{sk}, a_i)$ for all $i \in [m]$, $\mathsf{ct}' \leftarrow \mathcal{A}(\{\mathsf{ct}_i\}_{i\in[m]}; z)$, $(\boldsymbol{\pi}, b) \leftarrow \mathcal{X}_\mathcal{A}(\{\mathsf{ct}_i\}_{i\in[m]}; z)$, $a' \leftarrow (a_1, a_2, \cdots, a_m) \cdot \boldsymbol{\pi} + b$,

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}') \neq a'] = negl(\lambda). \tag{1}$$

Remark 1: The fact that our encoding scheme resembles usual fully homomorphic encryption (FHE) schemes does not contradict our assumption that it is a Linear-Only Encoding. In FHE, it is necessary for the ciphertext after non-scalar multiplication to be decrypted with specified secret key [33] or to be performed key-switching procedure [34] for correct decryption, both of which are not allowed in our encoding scheme.

We finally describe the ring learning with errors (RLWE) assumption as follows.

Definition 4 (Ring LWE assumption): Let R be $\mathbb{Z}[X]/(X^N + 1)$ with a power of two integer N, and $R_q = R/qR$. Then, a decision ring LWE (RLWE) assumption is hard to distinguish the two distributions

- $\{(\mathbf{a}_i, \mathbf{b}_i = \mathbf{a}_i\mathbf{s} + \mathbf{e} \bmod q) : \mathbf{a}_i, \mathbf{s} \leftarrow R_q \text{ and } \mathbf{e} \leftarrow \chi_\sigma^N\}\}$
- $\{(\mathbf{a}, \mathbf{u}) : \mathbf{a}, \mathbf{u} \leftarrow R_q\}$

where χ_σ is a discrete Gaussian distribution with a standard deviation σ defined on $\mathbb{Z}$.[4]

The following lemma is a corollary from [36].

Lemma 3 (Hardness of RLWE [36]): Let N be a power of two integer, and R be the ring $\mathbb{Z}[X]/(X^N + 1)$ and $R_q = R/qR$, where $q = 1 \bmod 2N$. If $\sigma = \alpha q > \sqrt{N}(Nm/\log(Nm))^{1/4} \cdot \omega(\sqrt{\log N})$, then the RLWE problem with respects to parameters N, q, m, χ where χ a discrete Gaussian with standard deviation σ, is quantumly at least as hard as approximate the shortest vector problem within a factor $\widetilde{O}(N(Nm/\log(Nm))^{1/4}/\alpha)$ in an ideal of the ring $\mathbb{Z}[\zeta_{2N}]$ where ζ_{2N} is the $2N$-root of the unity.

Remark 2 (SIMD Operation): When p is a prime such that $p \equiv 1 \bmod 2N$, the message space R_p of the above encoding scheme is isomorphic as a ring to $\mathbb{Z}_p^N$. Therefore, we can simultaneously encode N messages from $\mathbb{Z}_p$ to a single encoding, and a single operation (addition or scalar multiplication) on the ciphertexts or messages of R_p corresponds to those on many messages of $\mathbb{Z}_p$, which is called a single instruction multiple data (SIMD) operation.

C. Succinct Non-Interactive Arguments

In this section, we provide definitions of the notion of succinct non-interactive zero-knowledge argument of knowledge (zk-SNARK).

Definition 5 (Non-interactive proof system): Let $\mathcal{R}$ be a relation which comprises pairs $(\phi, \omega) \in \mathcal{R}$. We call ϕ the statement and ω the witness. A non-interactive proof system for a relation $\mathcal{R}$ comprises three algorithms as follows:

- $\mathsf{Setup}(\mathcal{R})$: The setup algorithm provides a common reference string crs and a simulation trapdoor τ for the relation $\mathcal{R}$.
- $\mathsf{Prove}(\mathcal{R}.\mathsf{crs}, \phi, \omega)$: The prove algorithm outputs a proof π.
- $\mathsf{Verify}(\mathcal{R}.\mathsf{crs}, \phi, \pi)$: The verify algorithm provides 0 (reject) or 1 (accept).

If a verifier can only verify a proof using the verifiable common string vrs which contains secret information, the proof system is called the 'designated' non-interactive proof system.

Completeness roughly says that an honest prover can convince an honest verifier. We formally describe the completeness.

Definition 6 (Completeness): $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Sim})$ algorithms are said to have completeness if they satisfy that for all

$$\Pr\left[\begin{matrix}(\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}(\mathcal{R}); \\ \pi \leftarrow \mathsf{Prove}(\mathcal{R}, \mathsf{crs}, \phi, \omega)\end{matrix} \middle| \mathsf{Verify}(\mathcal{R}, \mathsf{crs}, \phi, \omega) = 1\right], \tag{2}$$

[4]Formal definition of RLWE is different to the definition 4, but if N is power of two, then the definition of RLWE is the same as definition 4. We refer to [35].

is bigger than $1 - \mathsf{negl}(\lambda)$, where $\mathsf{negl}(\lambda)$ is a negligible function in λ.

The knowledge soundness says that if a prover can provide a valid proof, then there is an efficient algorithm for extracting the witness for the given statement with the same inputs and random coins. We formally describe the knowledge soundness.

Definition 7 (Knowledge soundness): For any polynomial time adversary $\mathcal{A}$, there exists a polynomial time extractor $\mathcal{X}_{\mathcal{A}}$ such that

$$\Pr\left[\begin{array}{l|l}(\mathcal{R}, z) \leftarrow \mathsf{R}(1^\lambda); & \\ (\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}(\mathcal{R}); & (\phi,\pi) \notin \mathcal{R} \\ ((\phi,\pi);\omega) \leftarrow (\mathcal{A}\|\mathcal{X}_{\mathcal{A}})(\mathcal{R}, z, \mathsf{crs}) & \mathsf{Verify}(\mathcal{R}.\mathsf{crs}, \phi, \pi) = 1\end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $\mathsf{negl}(\lambda)$ is a negligible function in λ.

The zero-knowledge roughly says that a prover cannot leak any information except for the truth of the statement. We also formally describe the zero-knowledge.

Definition 8: [ϵ_{zk}-Zero-Knowledge] For all $(\phi, \omega) \in \mathcal{R}$ and an adversary $\mathcal{A}$, the following equality holds.

$$\Pr\left[\begin{array}{l|l}(\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}(\mathcal{R}); & \\ \pi \leftarrow \mathsf{Prove}(\mathcal{R}, \mathsf{crs}, \phi, \omega) & \mathcal{A}(\mathcal{R}, z, \mathsf{crs}, \pi) = 1\end{array}\right]$$
$$\approx \Pr\left[\begin{array}{l|l}(\mathsf{crs}, \tau) \leftarrow \mathsf{Setup}(\mathcal{R}); & \\ \pi \leftarrow \mathsf{Sim}(\mathcal{R}, \mathsf{crs}, \phi, \tau) & \mathcal{A}(\mathcal{R}, z, \mathsf{crs}, \pi) = 1\end{array}\right],$$

where $\approx$ means that the difference of two probability is bounded by $\epsilon_{zk} \ll 1$.

Now we present the definition of succinctness and finally, zk-SNARK.

Definition 9 (Succinctness): A non-interactive proof system is succinct if the proof size and verification time is polynomial in the security parameter λ and $|\phi| + \log |w|$ where ϕ and w are input and witness of the relation $\mathcal{R}$, respectively.

Definition 10 (zk-SNARK): If a non-interactive proof system satisfies the completeness, knowledge soundness, zero-knowledge, and succinctness, then it is called zk-SNARK. In addition, if it requires a secret information for a verifier (to Verify), it is called the designated zk-SNARK.

III. ZK-SNARK FROM RLWE

In this section, we propose a zk-SNARK from RLWE. Here, we use a ring $R_p = \mathbb{Z}_p[X]/(X^N + 1)$ as a message space with a power-of-two N and a prime p such that $p = 1 \bmod 2N$ to fully exploit slot-wise computations. Indeed, the ring isomorphism $R_p \cong \mathbb{Z}_p^N$ allows N slot-wise computations. Then, we can simultaneously verify at most N possibly distinct circuits (having the same bound on the size) in a single decryption process. Previously, to verify circuits with N pairs of input and output, a verifier must perform N decryption processes.

Intuition of zk-SNARK construction: Groth [9] and ours. Since our zk-SNARK construction resembles that of Groth [9], we briefly overview the intuition behind the construction of [9] and the distinguished aspect of ours. Both constructions are based on the QAP (detail will be given below) where the divisibility of $v_{C,i}(x) \cdot w_{C,i}(x) - y_{C,i,o}(x)$ by $t_C(x)$ is equivalent to the correct evaluation of a circuit C with an input i and output o. The divisibility is checked by letting a prover to provide the quotient $h_{C,i,o}(x)$ along with (the part of) polynomials $v_{C,i}(x), w_{C,i}(s), y_{C,i,o}(x)$ satisfying the divisibility condition with $t_C(x)$. On the other hand, for efficient verification and succinct proof, instead of working directly on those polynomials, a verifier (or a trusted party) encodes the random point r (and its corresponding powers) with an linearly homomorphic encoding scheme so that a prover can generate a proof without knowing r (necessary for soundness). Two significant differences of our construction from Groth [9] are (i) We use RLWE encoding scheme for better amortized proof size, which additionally requires noise flooding technique for zero-knowledge; (ii) We exploit generalized version of QAP for a finite ring (instead of field) to deal with the messages space R_p of the RLWE encoding scheme.

A. Ring-Quadratic Arithmetic Program (Ring-QAP)

Previously, QAP has been used to confirm arithmetic circuit satisfiability over finite field $\mathbb{F}$, so every element which appears at the above definition is contained in $\mathbb{F}$. However, since the message space of the RLWE based encoding is not a field, but a ring, the existing QAP definition cannot capture the case. Thus, the necessity for ring-QAP is natural, which is the generalization of the previous QAPs from a finite field $\mathbb{F}$ to a ring $\mathfrak{R}$. We first introduce a definition of ring-QAP.

Definition 11 (Ring-QAP; adapted from [27]): A QAP $\mathcal{Q}$ over a ring $\mathfrak{R}$ comprises three sets of $m + 1$ polynomials $\mathcal{V} = \{v_k(x)\}, \mathcal{W} = \{w_k(x)\}, \mathcal{Y} = \{y_k(x)\}$ (over $\mathfrak{R}$), for $k \in \{0, 1, \ldots, m\}$, and a target polynomial $t(x) \in \mathfrak{R}[x]$. Suppose $C : \mathfrak{R}^n \to \mathfrak{R}^{n'}$ is an arithmetic circuit that takes as input n elements of $\mathfrak{R}$ and outputs n' elements, for a total of $n + n'$ I/O elements. Then we say that $\mathcal{Q}$ computes C if: $(a_1, a_2, \ldots, a_{n+n'}) \in \mathfrak{R}^{n+n'}$ is a valid assignment of C's inputs and outputs, if and only if there exist coefficients $(a_{n+n'+1}, a_{n+n'+2}, \ldots, a_m)$ such that $t(x)$ divides $p(x)$, where:

$$p(x) = \left(v_0(x) + \sum_{i=1}^m a_i v_i(x)\right)\left(w_0(x) + \sum_{i=1}^m a_i w_i(x)\right) - \left(y_0(x) + \sum_{i=1}^m a_i y_i(x)\right).$$

Remark 3 (Description of $\mathcal{V}, \mathcal{W}, \mathcal{Y}$): Recall that, in the original QAP [27] over a finite field, the target polynomial $t(x)$ is defined by $\prod_g (x - r_g)$ with distinct roots r_g's, each corresponding to each multiplication gate. Then, polynomials $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$ are constructed in a way that their evaluation values on r_g, i.e., $(v_0(r_g) + \sum_{i=1}^m a_i v_i(r_g))$, $(w_0(r_g) + \sum_{i=1}^m a_i w_i(r_g))$, and $(y_0(r_g) + \sum_{i=1}^m a_i y_i(r_g))$ are respectively, left input, right input, and output of the multiplication gate corresponding to r_g. In our Ring-QAP, the target polynomial $t(x)$, along with $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$ are defined in the same way as those of the original QAP, but with a caution in choosing r_g's due to the following Schwartz-Zippel lemma on the ring.

For the soundness of zk-SNARKs, the Schwartz-Zippel lemma should be required. The original lemma only provides an upper bound of the probability that the evaluation of nonzero multivariate polynomials at a random point from some finite set is zero. Thus, it does not also capture a polynomial ring case, but fortunately Schwartz [37] and Bishonoi *et al.* [38] deal with the ring variant of Schwartz-Zippel lemma as follows.

Lemma 4 (Generalized Schwartz-Zippel Lemma [37], [38]): Let $\mathfrak{R}$ be a finite ring, and let $S \subseteq \mathfrak{R}$ be a set satisfying that

$$\text{for all } x, y \in S \text{ such that } x \neq y, \quad x - y \text{ is invertible.}^5$$

Then, for all n-variate nonzero polynomial $f : \mathfrak{R}^n \to \mathfrak{R}$ of total degree D,

$$\Pr_{x \leftarrow S^n}[f(x) = 0] \leq \frac{D}{|S|}.$$

Example 1 (Set S with Maximal Cardinality): For a prime p, when a ring $R_p = \mathbb{Z}_p[X]/(X^N+1)$ is isomorphic to $\mathbb{Z}_p^N$, a set $S := \{(a, a, \ldots, a) : a \in \mathbb{Z}_p\} \subseteq R_p$ satisfies the desired condition of the above lemma with $\mathfrak{R} = R_p$. Note that $|S| = p$ and S has the maximal cardinality among all such subsets: if a set S' has cardinality bigger than p, then by pigeon hole principle, there exist distinct $x, y \in S'$ having the same value in at least one of its coordinate (hence, $x - y$ is not invertible).

To exploit the above lemma in our case, we choose $S := \{a \cdot \mathbf{1}^N : a \in \mathbb{Z}_p\}$, where $\mathbf{1}$ is a vector of ones; all coefficients are one. Thus, we directly obtain the Theorem 1 that the probability that f can be vanished at a random point can be described as follows.

Theorem 1 (R_p-QAP with maximal cardinality): For a ring $R_p \cong \mathbb{Z}_p^N$ (with prime p) and any arithmetic circuit $C : R_p^n \to R_p^{n'}$ of fan-in 2 with m wires and d multiplication gates, if $p \geq d$, then there exists a QAP $\mathcal{Q} = (\mathcal{V} = \{v_k(x)\}_{k=0}^m, \mathcal{W} = \{w_k(x)\}_{k=0}^m, \mathcal{Y} = \{y_k(x)\}_{k=0}^m, t(x))$ computing C. More precisely,

$$t(x) := \prod_{i=0}^{d-1}(x - r_i),$$

and $\mathcal{V}, \mathcal{W}, \mathcal{Y}$ can be defined by combining $\{\lambda_j(x)\}_{j=0}^{d-1}$ where

$$\lambda_j(x) := \prod_{i=0,(i\neq j)}^{d-1} \frac{x - r_i}{r_j - r_i},$$

for distinct roots $r_0, \cdots, r_{d-1} \in A := \{a \cdot \mathbf{1}^N : a \in \mathbb{Z}_p\} \subseteq R_p$.

B. Our Designated zk-SNARK from RLWE

We now describe our zk-SNARK with RLWE-based linear-only encoding (Section II-B), which is composed of three algorithms: $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$. Roughly speaking, the protocol is a natural conversion from a DL-based encoding into the RLWE-based linear-only encoding with R_p-QAP with maximal cardinality where $R_p = \mathbb{Z}_p[x]/(x^N + 1)$ and N is a power of 2. We assume $p, q \equiv 1 \bmod 2N$ which implies that $R_p \cong \mathbb{Z}_p^N$ and $R_q \cong \mathbb{Z}_q^N$.

Let $\mathcal{R}$ be a relation that a prover wants to prove, which is represented by an arithmetic circuit with n inputs, n' outputs, and m wires (composed of input and output of the circuit, output of multiplication gates, and constant addition and multiplication gates). Let $\mathcal{V} = \{v_k(x)\}, \mathcal{W} = \{w_k(x)\}, \mathcal{Y} = \{y_k(x)\}$, and $t(x)$ be the ring-QAP (Definition 11) corresponding to this arithmetic circuit. Then, for the valid statements $(a_1, \ldots, a_{n+n'})$ and witnesses $(a_{n+n'+1}, \ldots, a_m)$ of the relation $\mathcal{R}$, it holds that

$$\left(v_0(x) + \sum_{i=1}^m a_i v_i(x)\right)\left(w_0(x) + \sum_{i=1}^m a_i w_i(x)\right) - \left(y_0(x) + \sum_{i=1}^m a_i y_i(x)\right) = h(x)t(x).$$

for some polynomial $h(x)$ of degree at most the number of multiplication gates.

$(\mathbf{crs}, \mathbf{vrs}) \leftarrow \mathbf{Setup}(\mathcal{R})$. This algorithm receives a relation $\mathcal{R}$ as an input and outputs a common refernce string crs. In addition, our scheme only supports a designated verifier and Setup outputs an additional information, called vrs. The trusted third party (TTP) chooses random elements $\alpha, \beta, \delta, r \leftarrow R_p$, and generates the master secret key of RLWE encoding $\mathsf{s} \leftarrow R_q$. Then, TTP computes crs and vrs as follows:

$$\mathsf{crs} = \left\{ \begin{array}{l} \mathsf{Enc}(\alpha), \mathsf{Enc}(\beta), \mathsf{Enc}(\delta), \{\mathsf{Enc}(0_i)\}_{i\in[n\log q + 2\lambda]}, \\ \left\{\mathsf{Enc}\left(\frac{\beta u_i(r) + \alpha v_i(r) + w_i(r)}{\delta}\right)\right\}_{i=n+n'+1}^m, \\ \left\{\mathsf{Enc}(r^i), \mathsf{Enc}\left(\frac{r^i t(r)}{\delta}\right)\right\}_{i=0}^d \end{array} \right\},$$

$$\mathsf{vrs} = \{\mathsf{sk}, \alpha, \beta, \delta, r\}.$$

Here Enc denotes an encoding algorithm for RLWE as defined in Section II-B and $\mathsf{Enc}(0_j)$'s are encodings of zero. TTP makes public crs, however, vrs is sent to the designated verifier and it should be kept secret.

$\pi \leftarrow \mathbf{Prove}(\mathbf{crs}, a_1, \ldots, a_m)$. To generate a proof π, a prover executes Prove algorithm which receives crs, statements and witnesses as an input. He chooses random elements $\gamma_u, \gamma_v \leftarrow R_p$ and generates three encodings $\mathsf{Enc}(A(r))$, $\mathsf{Enc}(B(r))$, and $\mathsf{Enc}(C(r))$ through homomorphic summations and scalar multiplications where

$$A(r) = \alpha + \sum_{i=0}^m a_i u_i(r) + \gamma_u \delta,$$

$$B(r) = \beta + \sum_{i=0}^m a_i v_i(r) + \gamma_v \delta,$$

$$C(r) = \sum_{i=n+n'+1}^m \left(\frac{\beta u_i(r) + \alpha v_i(r) + w_i(r) + h(r)t(r)}{\delta}\right) + \gamma_v A(r) + \gamma_u B(r) - \gamma_u\gamma_v\delta^2,$$

and for $\mathbf{r}_A, \mathbf{r}_B, \mathbf{r}_C \leftarrow \{0,1\}^{N\log q + 2\lambda}$ and $I \in \{A, B, C\}$, computes re-randomized encodings

$$\mathsf{Enc}(I(r)) \leftarrow \mathsf{Enc}(I(r)) + (0, \mathbf{e}_I^*) + \sum_{j=1}^{N\log q + 2\lambda} r_{I,j}\mathsf{Enc}(0_j) \bmod qR,$$

where $\mathbf{e}_I^*$ is sampled from a distribution that outputs large elements to smudge the error terms in RLWE encodings. We will formally describe how to sample $\mathbf{e}_I^*$ in the next Section III-C.

$1/0 \leftarrow$ **Verify**$(\pi, vrs, a_1, \ldots, a_{n+n'})$. A (designated) verifier who has vrs can check the validity of $\pi = (\mathsf{Enc}(A(r)), \mathsf{Enc}(B(r)), \mathsf{Enc}(C(r)))$. The verifier can obtain a tuple (A, B, C) through executing a decryption algorithm from π. Then, he tests

$$AB = \alpha\beta + C\delta + \sum_{i=0}^{n+n'} a_i(\beta u_i(r) + \alpha v_i(r) + w_i(r)). \tag{3}$$

and accepts the proof if the test passes.

C. Noise Flooding with Optimized Parameters

A verifier in our protocol decrypts the RLWE ciphertexts using a secret key to obtain messages. The decryption process of the RLWE based encoding gives a verifier the error terms as well as corresponding message. Due to the construction of RLWE ciphertexts, error terms may contain some information about affine computations which are conducted on encrypted data and thus information about the error term must be hidden. To overcome this restriction, previous works [16], [23], [24] introduced a noise flooding technique where one adds a large values to hide an existing error term.

Noise flooding in the previous work. In previous work, prover injects a sufficiently large error $\mathbf{e}^*$ to a proof ciphertext $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e})$ so that the added ciphertext $(\mathbf{a}, \mathbf{b} + \mathbf{e}^* \bmod qR)$ has an error $\mathbf{e} + \mathbf{e}^* \bmod qR$ that is statistically close to $\mathbf{e}^*$. Then, following lemma guarantees that no adversary can obtain any significant information on the error term $\mathbf{e}$ from the decryption of a proof ciphertext.

Lemma 5 ([23]): Let B_1, B_2 be positive numbers and x be a fixed number in an interval $[-B_1, B_1]$. Let Y be the uniform distribution defined on an interval $[-B_2, B_2]$. Then, the statistical distance between a distribution Y and $Y + x$ is bounded by B_1/B_2.

Specifically, the lemma implies that $B_2 = B_1 \times 2^\kappa$ bounds the statistical distance between two distributions to be $2^{-\kappa}$. Then, from the probability preservation property of statistical distance, it gives that the scheme with noise flooding satisfies the zero-knowledge property (Definition 8) with $\epsilon_{zk} = 2^{-\kappa}$.

Our noise flooding with tighter parameters. In this paper, we propose to investigate the computational costs of distinguishing two distributions with the notion of Hellinger distance as recently proposed by [39]. From this, by computing the cost more tightly, we can use better parameters while providing the same zero-knowledge property. More specifically, we compute more tight lower bound of the computational costs required for an adversary to break the zero-knowledge then set the parameter accordingly.

We remark that, conventional argument based on the statistical distance (as above) requires that a new error to be larger than the initial error in ratio *exponential* to the statistical parameter. To circumvent this limitation, there have been several approaches (especially lattice-based cryptography) proposed to use closeness measures other than statistical distance on distributions [39]–[46]. Our approach can be seen as an adaptation of [39] for the zero-knowledge property in lattice-based encoding scheme.

At first, we introduce the Hellinger distance and its property, a key ingredient for our better noise flooding technique.

Definition 12 (Hellinger distance): Let D_1, D_2 be two discrete distributions over a domain X. The Hellinger distance between D_1 and D_2 is defined by

$$H(D_1, D_2) = \sqrt{1 - \sum_{x \in X} \sqrt{D_1(x) D_2(x)}}.$$

If $H(D_1, D_2)$ is smaller than 2^{-t}, we say that a pair (D_1, D_2) is 2^{-t}-Hellinger close pair.

[39] recently showed that replacing a distribution D_1 with the other distribution D_2 in the security game for the decision problem loss only a few bit security if (D_1, D_2) is $2^{-\kappa/2}$-Hellinger close pair. More formally, they proved the following lemma.

Lemma 6 (Theorem 5 in [39]): Let Π^{D_1} be a cryptographic primitive with black box access to a distribution D_1 and G^{D_1} be a decision security game regarding Π^{D_1}. Suppose that (D_1, D_2) is $2^{-\kappa/2}$-Hellinger close pair. Then, if Π^{D_1} achieves κ-bit security, then Π^{D_2} satisfies $\kappa - 6.847$-bit security.

Now, with this lemma, we can show that adding a noise from appropriate discrete Gaussian distribution achieves the goal of noise flooding technique as follows: Let D_1 and D_2 be discrete Gaussian with the standard deviation σ' centered at zero and $\mathbf{e}$, respectively. Then, from the above lemma with Π^{D_i} as a designated zk-SNARK from RLWE with black box access to D_i, it suffices to show that $H(D_1, D_2) \leq 2^{-\kappa/2}$, which results in G^{D_2}, a security game for the zero-knowledge (Definition 8), is κ-bit secure, i.e., the advantage of adversary is less than $2^{-\kappa}$ (note that G^{D_1} is already $\geq \kappa + 6.847$ secure since it does not have any information on the error term $\mathbf{e}$).

The following lemma provides a sufficient size of σ' in the above argument given that $\|\mathbf{e}\| \leq B$.

Lemma 7: Let P and Q be discrete Gaussian distributions with the standard deviation σ' centered at zero and y, respectively, such that $|y| \leq B$. Then, it satisfies that

$$H(P, Q)^2 \leq 1 - \exp(-\frac{B^2}{8\sigma'^2}).$$

Proof: We will regard P, Q as continuous Gaussian distributions because σ' that we will use is sufficiently large. Then, from the definition of Gaussian distribution and Hellinger distance,

$$H(P, Q)^2 = 1 - \frac{1}{\sqrt{2\pi}\sigma'} \int_{\mathbb{R}} \exp\left(-\frac{\frac{1}{4}(x-y)^2 + \frac{1}{4}x^2}{\sigma'^2}\right) dx.$$

The integral can be converted as follows.

$$\exp(-\frac{\frac{1}{8}y^2}{\sigma'^2}) \times \int_{-\infty}^{\infty} \exp(-\frac{1}{2\sigma'^2} \cdot (x - \frac{1}{2}y)^2) dx$$

Using the fact that $\int_{\mathbb{R}} \exp(-cu^2)du = (c/\pi)^{-1/2}$ for all $c > 0$, we obtain

$$\exp(-\frac{y^2}{8\sigma'^2}) \times \sigma'\sqrt{2\pi}.$$

Substituting this to the first equation gives the claim. ■

Now, to satisfy κ-bit security in zero-knowledge (i.e., $\epsilon_{zk} = 2^{-\kappa}$ in Definition 8), it requires that

$$1 - \exp(-\frac{B^2}{8\sigma'^2}) < 2^{-\kappa}.$$

In other words,

$$\frac{B}{\sigma'} \geq \frac{\sqrt{-\frac{1}{\ln(1-2^{-\kappa})}}}{2\sqrt{2}} = O(2^{-\kappa/2}). \tag{4}$$

Parameters. Let κ be the statistical parameter and B the size of the error term in the final encodings (before noise flooding). To achieve the κ-bit security, it suffices to set σ' as above (4). Then, the remaining part is to choose q such that $q/2p$ is bigger than $\Omega(\sigma')$ to achieve the correctness. On the other hand, according to the previous analysis that exploited the statistical distance as a measure of closeness of two distributions, σ' is approximately set to $\Omega(2^{\kappa}B)$, which implies that $q/2p \geq \Omega(2^{\kappa}B)$. Consequently, in our tight analysis based on the Hellinger distance, q is polynomial in κ rather than $\exp(\kappa)$ as in the conventional analysis.

More specifically, in later Section IV, we will present improved concrete parameters due to the analysis with the Hellinger distance, and estimate the size of proof based on the improved parameters.

D. Security Proofs

Theorem 2: Let κ be the statistical security parameters, and λ be the security parameter. Let $N = N(\lambda)$, $q = q(\lambda)$ and $\sigma = \sigma(\lambda)$ be RLWE parameters in Lemma 3 satisfying that $B = 8p\sigma\sqrt{m + N\log q + 2\lambda + 3}$, where m is the number of wires in target circuit C. Assume that our RLWE encoding scheme (Definition 2) is a Linear-Only Encoding scheme (Definition 3). Then, for the circuit C, the scheme described in Section III-B is a designated zk-SNARK (Definition 10).

Clearly, it is straightforward to prove the completeness. Moreover, our scheme consists of three RLWE encodings which are polynomial size in λ, and the verification procedure takes polynomial time in λ, so the succinctness also holds.

We now introduce a leftover hash lemma [47] which is necessary to prove zero-knowledge property.

Lemma 8 (Specialized leftover hash lemma): For non-negative integers $n, q, 2, t$ and real number ϵ, if $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times t}$ and $\mathbf{r} \leftarrow \mathbb{Z}_2^t$, then we have

$$\mathrm{SD}((\mathbf{A}, \mathbf{A}\cdot\mathbf{r}), (\mathbf{A}, \mathbf{u})) \leq \frac{1}{2}\cdot\sqrt{2^{-t}\cdot q^n},$$

where $\mathbf{A}\cdot\mathbf{r}$ is computed in $\mathbb{Z}_q$, and $\mathbf{u} \leftarrow \mathbb{Z}_q^n$. Thus, if $t > N\log q + 2\log(1/\epsilon)$, then two distributions are $\mathrm{SD}((\mathbf{A}, \mathbf{A}\cdot\mathbf{r}), (\mathbf{A}, \mathbf{u})) \leq \epsilon$.

Lemma 9 (Zero-knowledge): The protocol has zero knowledge under the parameters in Theorem 2.

Proof: We build a simulated proof π' that follows the same distribution as a proof π. The algorithm comprises two steps; constructing elements and generating RLWE ciphertexts using crs. First, choose $A', B' \leftarrow R_p$, and compute

$$C' = \frac{A'B' - \alpha\beta - \sum_{i=0}^{n+n'} a_i(\beta u_i(r) + \alpha v_i(r) + w_i(r))}{\delta}.$$

Then, using crs, we can generate three RLWE ciphertexts $\mathsf{Enc}(A'), \mathsf{Enc}(B')$, and $\mathsf{Enc}(C')$ and output a proof $\pi' = (\mathsf{Enc}(A), \mathsf{Enc}(B), \mathsf{Enc}(C))$. Then, the simulated proof can pass the verification (3).

As the last step, we need to prove that π and π' are statistically or computationally indistinguishable. Each encoding in π and π' consists of a pair $(\mathbf{a}, \mathbf{b} \bmod qR)$ with $\mathbf{b} = \mathbf{a}\cdot\mathbf{s} + \mathbf{e} + \frac{q}{p}\mathbf{m}$. By the leftover hash lemma 8, the first component of any encoding looks like a random element in $R_q \cong \mathbb{Z}_q^N$. More precisely, every element in R_q can be regarded as a vector in $\mathbb{Z}_q^N$, so we apply the lemma 8 to randomize the first component of each encoding when $N\log q + 2\lambda$ encodings of zero are provided.

On the other hand, the noise flooding technique in Lemma 5 shows that $\mathbf{e}$ is independent of any witness since the error term looks like a random element. Therefore, two proofs are indistinguishable. ■

Lemma 10 (Knowledge soundness): The protocol has knowledge soundness under the parameters in Theorem 2.

Proof: Suppose that there exists an adversary $\mathcal{A}$ which can break knowledge soundness with a non-negligible probability. We will construct a knowledge extractor $\mathcal{X}$ based on $\mathcal{A}$.

Let $\pi = (A(r), B(r), C(r))$ be a tuple of RLWE ciphertexts. Then, $\mathcal{A}$ which allows affine computations can obtain follows.

$$\begin{aligned} A = &A_\alpha\alpha + A_\beta\beta + A_\delta\delta + A(r) \\ &+ \sum_{i=n+n'+1}^{m} A_i(\frac{\beta u_i(r) + \alpha v_i(r) + w_i(r)}{\delta}) \\ &+ A_{h(r)}\frac{t(r)}{\delta}, \end{aligned}$$

where $A_\alpha, A_\beta, A_\delta, \{A_i\}_{i=n+n'+1}^m$ are scalars in R_p and $A(r), A_{h(r)}$ are polynomials of degree d with coefficients in R_p. Similarly we obtain representations about B and C.

Our construction allows slot-wise computations by ring operations, and the verification (3) can be considered as slot-wise computations, i.e., independent computations on each $\mathbb{Z}_p$. Note that a verifier in our protocol outputs accept when the equation holds for all slots, and it is enough to show slot-wise knowledge soundness.

Since $\mathcal{A}$ can break the soundness, $\mathcal{A}$ can pass the verification equation on each slot. For simplicity, we use a notation tilde to denote slot-wise results. Then, for each slot, the verification equation is considered as follows.

$$\widetilde{A}\widetilde{B} = \widetilde{\alpha}\widetilde{\beta} + \widetilde{C}\widetilde{\delta} + \sum_{i=0}^{n+n'} \widetilde{a}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r)). \tag{5}$$

Moreover, after $\mathcal{A}$ computes affine operations rings, $\mathcal{A}$ can obtain equations

$$\begin{aligned}\widetilde{A} = \widetilde{A}_\alpha\widetilde{\alpha} + \widetilde{A}_\beta\widetilde{\beta} + \widetilde{A}_\delta\delta + \widetilde{A}(r) \\ + \sum_{i=n+n'+1}^{m} \widetilde{A}_i(\frac{\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r)}{\widetilde{\delta}}) \\ + \widetilde{A}_{h(r)}\frac{\widetilde{t}(r)}{\widetilde{\delta}},\end{aligned} \qquad (6)$$

where tilded elements are included in a finite field $\mathbb{Z}_p$ for some prime p. Similarly, $\mathcal{A}$ obtains representations about $\widetilde{B}$ and $\widetilde{C}$.

Now, we reconsider the random elements $\widetilde{\alpha}, \widetilde{\beta}, \widetilde{\delta}$ as formal variables. Then, $\widetilde{A}\widetilde{B}$ contains formal variables $\widetilde{\alpha}^2$, $\widetilde{\beta}^2$ and $1/\widetilde{\delta}^2$, but they are not included in the right-hand side of the verification (5) in each slot. Thus, for passing the verification process, $\widetilde{A}_\alpha\widetilde{B}_\alpha\alpha^2$ must be the zero, which implies $\widetilde{A}_\alpha$ or $\widetilde{B}_\alpha$ is zero. Without loss of generality, we assume that $\widetilde{B}_\alpha = 0$. Similarly, we compare the coefficients of $\widetilde{\alpha}\widetilde{\beta}$ with $\widetilde{\beta}^2$.

$$\begin{aligned}&\text{coeff of } \widetilde{\alpha}\widetilde{\beta} \text{ in LHS of (5)} = \widetilde{A}_\alpha\widetilde{B}_\beta + \widetilde{A}_\beta\widetilde{B}_\alpha \\ &\text{coeff of } \widetilde{\alpha}\widetilde{\beta} \text{ in RHS of (5)} = 1 \\ &\text{coeff of } \widetilde{\beta}^2 \text{ in LHS of (5)} = \widetilde{A}_\beta\widetilde{B}_\beta \\ &\text{coeff of } \widetilde{\beta}^2 \text{ in RHS of (5)} = 0\end{aligned}$$

Thus, it holds that $\widetilde{A}_\alpha\widetilde{B}_\beta + \widetilde{A}_\beta\widetilde{B}_\alpha = \widetilde{A}_\alpha\widetilde{B}_\beta = 1$ and $\widetilde{A}_\beta\widetilde{B}_\beta = 0$. Without loss of generality, we also assume that $\widetilde{A}_\alpha = \widetilde{B}_\beta = 1$ and $\widetilde{B}_\beta = 0$. For a coefficient of $1/\delta^2$, we observe that

$$\begin{aligned}&\left(\widetilde{A}_{h(r)}\widetilde{t}(r) + \sum_{i=n+n'+1}^{m} \widetilde{A}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r))\right) \\ &\times \left(\widetilde{B}_{h(r)}\widetilde{t}(r) + \sum_{i=n+n'+1}^{m} \widetilde{B}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r))\right) \\ &= 0.\end{aligned}$$

Moreover, for the coefficients of $\widetilde{\alpha}/\widetilde{\delta}$ and $\widetilde{\beta}/\widetilde{\delta}$, we observe that

$$\begin{aligned}&\left(\sum_{i=n+n'+1}^{m} \widetilde{A}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r))\widetilde{A}_{h(r)}\widetilde{t}(r)\right) \times \widetilde{B}_\alpha \\ &= 0, \\ &\left(\sum_{i=n+n'+1}^{m} \widetilde{B}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r) + \widetilde{B}_{h(r)}\widetilde{t}(r)\right) \times \widetilde{A}_\alpha \\ &= 0, \\ &\left(\sum_{i=n+n'+1}^{m} \widetilde{A}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r))\widetilde{A}_{h(r)}\widetilde{t}(r)\right) \times \widetilde{B}_\beta \\ &= 0, \\ &\left(\sum_{i=n+n'+1}^{m} \widetilde{B}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r) + \widetilde{B}_{h(r)}\widetilde{t}(r)\right) \times \widetilde{A}_\beta \\ &= 0.\end{aligned}$$

Since $\widetilde{A}_\alpha = \widetilde{B}_\beta = 1$ and $\widetilde{A}_\beta = \widetilde{B}_\alpha = 0$, each component of a coefficient term of $1/\widetilde{\delta}^2$ is zero. Hence, $\widetilde{A}$ and $\widetilde{B}$ are rewritten as follows.

$$\begin{aligned}\widetilde{A} &= \widetilde{\alpha} + \widetilde{A}_\delta\widetilde{\delta} + \widetilde{A}(r) \\ \widetilde{B} &= \widetilde{\beta} + \widetilde{B}_\delta\widetilde{\delta} + \widetilde{B}(r)\end{aligned}$$

Moreover, it holds that

$$\begin{aligned}\widetilde{A}\widetilde{B} &= (\widetilde{\alpha} + \widetilde{A}_\delta\delta + \widetilde{A}(r))(\widetilde{B}_\beta\widetilde{\beta} + \widetilde{B}_\delta\delta + \widetilde{B}(r)) \\ &= \widetilde{\alpha}\widetilde{\beta} + \widetilde{C}\widetilde{\delta} + \sum_{i=0}^{n+n'} \widetilde{a}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r)).\end{aligned}$$

Thus, the verification equation (5) implies that

$$\begin{aligned}&\widetilde{B}(r)\widetilde{\alpha} + \widetilde{A}(r)\widetilde{\beta} + \widetilde{A}(r)\widetilde{B}(r) \\ &\quad + \sum_{i=n+n'+1}^{m} (\widetilde{C}_i\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r)) + \widetilde{h}(r)\widetilde{t}(r) \\ &= \sum_{i=0}^{n+n'} \widetilde{a}_i(\widetilde{\beta}\widetilde{u}_i(r) + \widetilde{\alpha}\widetilde{v}_i(r) + \widetilde{w}_i(r)),\end{aligned}$$

since $\widetilde{\delta}$ are considered as a formal variable. Moreover, since $\widetilde{\alpha}$ and $\widetilde{\beta}$ are also formal variables, we also observe that

$$\begin{aligned}\widetilde{B}(r) &= \sum_{i=0}^{n+n'} \widetilde{a}_i\widetilde{v}_i(r) + \sum_{i=n+n'+1}^{m} \widetilde{C}_i\widetilde{v}_i(r), \\ \widetilde{A}(r) &= \sum_{i=0}^{n+n'} \widetilde{a}_i\widetilde{u}_i(r) + \sum_{i=n+n'+1}^{m} \widetilde{C}_i\widetilde{u}_i(r), \\ \widetilde{A}(r)\widetilde{B}(r) &= \sum_{i=0}^{n+n'} \widetilde{a}_i\widetilde{w}_i(r) \\ &\quad + \sum_{i=n+n'+1}^{m} \widetilde{C}_i\widetilde{w}_i(r) + \widetilde{h}(r)\widetilde{t}(r).\end{aligned}$$

We set $\widetilde{a}_i = C_i$ for $i \in \{n + n' + 1, \cdots, m\}$. Then, it holds that $\widetilde{B}(r) = \sum_{i=0}^{m} \widetilde{a}_i\widetilde{v}_i(r)$ and $\widetilde{A}(r) = \sum_{i=0}^{m} \widetilde{a}_i\widetilde{u}_i(r)$. Moreover, for a variable r, we also observe that $\widetilde{A}(r)\widetilde{B}(r) = \sum_{i=0}^{m} \widetilde{a}_i\widetilde{w}_i(r) + \widetilde{h}(r)\widetilde{t}(r)$, which implies that for each slot, the set $\{\widetilde{a}_i\}_{i=n+n'+1}^{m} = \{\widetilde{C}_i\}_{i=n+n'+1}^{m}$ is a witness of the statement $\{\widetilde{a}_i\}_{i=1}^{n+n'}$. The slot-wise knowledge soundness completes the knowledge soundness of our construction.

■

IV. Proof Size Estimation

We now estimate the size of proof π of our designated zk-SNARK from RLWE. First, we provide concrete parameters of our protocol for circuits with 2^{16} gates for achieving the 110, 128, and 164-bit security, respectively. Due to the fancy analysis with respect to the Hellinger distance (equivalently, Rènyi divergence of order 1/2), concrete parameters improve considerably. Specifically, we describe the size of the proof of our scheme and then compare it with that of previous works.

Concrete parameters. We set the parameters to satisfy the following.

TABLE I
CONCRETE PARAMETERS OF OUR DESIGNATED ZK-SNARK FROM RLWE. HERE d IS THE NUMBER OF MULTIPLICATIVE GATES. WE FIX $T = 8$ AND $d = 2^{15}$ FOR FAIR COMPARISON.

λ	N	$\log\alpha$	$\log q$	$\log p$	$\log\sigma'$	m
164	2048	-104	260	32	146	2^{22}
128	2048	-104	208	32	101	2^{22}
110	2048	-104	180	32	71	2^{22}

- Our designated zk-SNARK has 164-bit security estimated by Albrecht *et al*'s LWE security estimator [48] with the `reduction_cost_model=BKZ.sieve` cost model.[6] With this model, the parameters of the previous work only satisfy 110-bit security, but not 164-bit security that was claimed by authors. Thus, we provide several types of parameter suggestions as follows.
  - For fair comparison, the bit-size of the message space, and other parameters related to circuits are the same as previous work [23], [24].
  - We provide new parameters satisfying 164-bit security that previous work desired.
  - We also provide a parameter achieving the 128-bit security to compare our amortized proof size and the smallest proof size of the group based zk-SNARK [9].
- To make a fair and easy comparison with previous work, we follow the way of selecting parameters in the previous papers as much as possible.

Let N, q and $\sigma = \alpha q$ be parameters of RLWE instances, and p be a 32-bit prime such that $p = 1 \bmod 2N$. A tight analysis based on the Hellinger distance instead of statistical distance loss 6.847 bit security [39]. In other words, to satisfy 32-bit statistical security that is the same as previous one, we need to consider parameters which require 39-bit statistical robust.

More precisely, for fair comparisons with previous work, we consider an arithmetic circuit with at most 2^{16} gates and $d = 2^{15}$ multiplication gates, which can cover many example applications such as the SHA-256 evaluation. Then, setting a tailcut parameter $T = 8$, $B = |e|$ is $8p\sigma\sqrt{m+t+3}$ where m is the number of wires in ring-QAP and $t = N\log q + 2\lambda$. Furthermore, σ' should satisfy that

$$H(e + \chi_{\sigma'} \| \chi_{\sigma'}) = \sqrt{1 - \exp\left(-\frac{1}{4}(\frac{B}{\sigma'})^2\right)} \le \frac{1}{2^{20}}.$$

Finally, we set $m = 2^{22}$ as in [24] so that $\sigma' \approx 2^{19} \cdot B$ and $8\sigma' < q/2p$ for the correctness (of encoding scheme). Then, it holds that

$$8 \cdot (2^{19} \cdot 8p\sigma\sqrt{m+t+3}) < q/2p.$$

For readability, we list the parameters of our zk-SNARK in Table I with various security parameter λ.

Interestingly, the Hellinger distance provides a significant practical improvement independent of our introduction of RLWE encoding. Moreover, with our RLWE encoding and ring-QAP, our protocol is much more efficient in the amortized sense than previous zk-SNARKs from SSPs and QAPs. For the same circuit satisfiability, Gennaro *et al.* [23] and Naganuma *et al.* [24] chose LWE parameters $(N, \log\alpha, \log q)$ as $(1400, -180, 736)$ for 110-bit security. On the other hand, we choose RLWE parameters $(2048, -98, 160)$ for achieving the same security. Thus, we reduce not only the size of an encoding in amortized sense but also the size of a single encoding.

TABLE II
COMPARISON OF PROOF SIZE OF EACH ZK-SNARKS.

	λ	Proof Size Total	Proof Size Amortized	PQ	Programs	Computational Assumption
Ours	110	276.5 KB	135 B	✓	Ring QAP	linear-only, RLWE
Ours	128	319.5 KB	156 B	✓	Ring QAP	linear-only, RLWE
Ours	164	399.4 KB	195 B	✓	Ring QAP	linear-only, RLWE
[24]	110	405 KB	-	✓	QAP	linear-only, LWE
[16]	110	270 KB	-	✓	SAP	linear-only[7], LWE
[23]	110	640 KB	-	✓	SSP[8]	PKE, PDH, LWE
[9]	128	138 B[9]	-	✗	QAP	PKE, PDH

Proof size. We can now estimate the size of the proof π for our scheme. Our proof π comprises three RLWE encodings, and the size of each encoding is about $2N\log q$ bits because of $R_q = \mathbb{Z}[X]/(X^N + 1)$ is the encoding space. Then, our encoding has the size 2048×260 bits ≈ 113.1 KB and the proof size is about 399.4 KB under 164-bit security since the proof π consists of three encodings. On 110-bit security, we can see that our scheme has about 276.5 KB of proof size which is smaller than all previous work [16], [23], [24] from lattices, e.g., Nitulescu [16] has 270 KB of proof, which is the smallest among previous lattice-based work.

If we consider the amortized proof size, our scheme is even comparable to the best result from the previous zk-SNARKs (without post-quantum security). More precisely, since our scheme allows N verification simultaneously for each proof and our proof size is about 284.2 KB under 128-bit security, the size of amortized proof is only 156 bytes with $N = 2048$ and it is almost the same as 138 bytes of Groth [9] which has the shortest proof size among all zk-SNARKs. The proof size for each scheme is summarized in Table 2.

Size of common reference string. In lattice-based zk-SNARK, the common reference string (crs) is composed of encodings for proving circuit evaluation and for leftover hash lemma (for zero-knowledge). In our proposal, the number of encodings in crs is the same as that in [24] which built a lattice-based zk-SNARK from QAP as ours. One difference is that our encoding from RLWE has $2N\log q$-bits which can be

[6]After we submitted this paper, a new estimator, called lattice-estimator, was published. However, we still use a previous estimator, named LWE-estimator because of the consistency of this paper.

[7]In original proposal, [16] is relied on linear-targeted malleability assumption, a weaker assumption than linear-only assumption. However, to achieve zk-SNARK, it also requires linear-only assumption or a similar one with efficient extractor.

[8]Here, we assume the evaluation circuit of SHA-256, which corresponds to an arithmetic circuit with 2^{16} gates or less [23].

[9]With bn-128 curve, https://github.com/zcash/zcash/issues/2465

reduced to $N \log q = 2048 \cdot 180$-bits with pseudorandom generator, while the one from LWE in [24] has $\log q' = 736$ bits (when reduced similarly with pseudorandom generator). When we consider the size of crs in *amortized sense* (with N amortization), however, the size of each encoding in our crs can be $\approx \log q = 180$ bits which is much smaller than that of [24].

Prover complexity. While our focus is on reducing the (amortized) proof size of the zk-SNARK as other work in the literature, we can also compare the prover/verifier complexity of our work with the previous works. Note that our SNARK requires ring multiplications over $\mathbb{Z}_q[X]/(X^N+1)$ which may cost $\Theta(N^2)$ operations over $\mathbb{Z}_q$ while previous SNARKs from LWE requires constant multiplications over $Z_q^{N'}$ which costs $\Theta(N')$ only. We remark that this problem can be mitigated by applying Number Theoretic Transform to our solution, which can reduce the cost to be $\Theta(N \log N)$ (in this case, we must take the ciphertext modulus $q \equiv 1 \bmod N$ so that the ciphertext space $R_q \cong \mathbb{Z}_q^N$). Then, our prover/verifier complexity can be roughly $\log N$ in amortized sense — it is now better than the previous work having N — given that we utilize the full batch N for the proof.

Extension to other circuits. We believe that our conversion and analysis can be applied to previous zk-SNARKs from SSP and SAP beyond the QAP. In particular, if someone wants to convert a SAP based zk-SNARK from LWE to RLWE assumption for achieving more smaller proof rather than our QAP based zk-SNARK, then, under the 128-bit security, he/she can obtain that 213 KB proof size, and it could be regarded as 104 bytes proof size for a single verification due to the amortized sense. However, as Naganuma *et al.* [24] mentioned, the scheme might be less efficient than QAP based zk-SNARK.

V. Experimental Results

Experimental setup. We implement our new lattice-based designated zk-SNARK and present the experiment results for our protocol. On implementation, we adopted `libnsark` library [49] for the zk-SNARK part and Microsoft `SEAL` library [50] for the RLWE encoding part then integrated them.[10] Our experiments were conducted on Linux Ubuntu 22.04.01 LTS with AMD EPYC 7502 CPU and 32 GB memory.

In our experiment, we generated a random circuit with 2^{15} multiplicative gates and 2^5 number of inputs, which can also cover the SHA-256 evaluation. Then, we measured the proof generation and verification time under the various security parameters given in Table I.

Prover time. Table III presents the proof generation time for each parameter. In our implementation, the main operation for

[10]To this end, we made some minor changes in each library, e.g., `SEAL` only supports a maximum 54-bit coefficient modulus space for $N = 2048$, while we require at least 180-bit.

TABLE III
Timing results with $T = 8$, $d = 2^{15}$, and number of inputs 2^5.

λ	Key Generation (s)	Prover Time		Verifier Time	
		Total (s)	Amortized (ms)	Total (s)	Amortized (ms)
110	13.46s	6.65s	3.2ms	0.011s	0.005ms
128	14.02s	7.19s	3.5ms	0.017s	0.008ms
164	16.95s	8.43s	4.1ms	0.023s	0.012ms

prover is a linear combination between RLWE encoding and a ring element (instead of multi-exponentiation in other zk-SNARKs). In Table III, it takes about 7 seconds to generate a proof under the parameter with $\lambda = 128$. For simplicity, we measured the time for generating a proof with only one instance, while an RLWE encoding supports batching multiple proofs by nature. More specifically, the RLWE encoding with $N = 2048$ and $\log q = 208$ can have 2048 messages simultaneously, and thus the amortized time for generating a proof for one instance is about 3.5 milliseconds.

Verifier time. Table III presents the verification time. As expected by the (3) (in Section III-B), the verifier complexity is independent of the circuit size and only depends on the number of inputs. According to our experiment, it takes about 11ms to verify a proof with the number of inputs 2^5, and amortized time for verifying a proof is about 0.005 ms.

Acknowledgment

Heewon Chung was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-01373, Artificial Intelligence Graduate School Program (Hanyang University)) and the research fund of Hanyang University (HY-202000000790015). Jeong Han Kim was partially supported by National Research Foundation of Korea (NRF) Grants funded by the Korean Government (MSIP) (NRF-2016R1A5A1008055 & 2017R1E1A1A0307070114) and by a KIAS Individual Grant(CG046002) at Korea Institute of Advanced Study.

References

[1] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM J. Comput.*, vol. 18, no. 1, pp. 186–208, 1989.
[2] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symposium on Security and Privacy (S&P),* 2014.
[3] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symposium on Security and Privacy (S&P),* 2014.
[4] J. Bonneau, I. Meckler, V. Rao, and E. Shapiro, "Coda: Decentralized cryptocurrency at scale," Cryptology ePrint Archive, Report 2020/352, 2020.
[5] Z. Ghodsi, T. Gu, and S. Garg, "Safetynets: Verifiable execution of deep neural networks on an untrusted cloud," in *Proc. NIPS,* 2017.
[6] Y. Zhang, D. Genkin, J. Katz, D. Papadopoulos, and C. Papamanthou, "v SQL: Verifying arbitrary SQL queries over dynamic outsourced databases," *in Proc. IEEE Computer Society*, 2017.
[7] ZKProof, "https://zkproof.org."
[8] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proc. IEEE Symposium on Security and Privacy (S&P),* 2013.

[9] J. Groth, “On the size of pairing-based non-interactive arguments,” in *Proc. EUROCRYPT,* 2016.
[10] B. Bünz, B. Fisch, and A. Szepieniec, “Transparent SNARKs from DARK compilers,” in *Proc. EUROCRYPT,* 2020.
[11] S. Setty, “Spartan: Efficient and general-purpose zkSNARKs without trusted setup,” in *Proc. CRYPTO,* 2020.
[12] J. Lee, “Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments,” in *Proc. TCC,* 2021.
[13] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, “Scalable, transparent, and post-quantum secure computational integrity,” Cryptology ePrint Archive, Report 2018/046, 2018.
[14] J. Zhang, T. Xie, Y. Zhang, and D. Song, “Transparent polynomial delegation and its applications to zero knowledge proof,” in *Proc. IEEE Symposium on Security and Privacy (S&P),* 2020.
[15] A. Chiesa, D. Ojha, and N. Spooner, “Fractal: Post-quantum and transparent recursive proofs from holography,” in *Proc. EUROCRYPT,* 2020.
[16] A. Nitulescu, “Lattice-based zero-knowledge snargs for arithmetic circuits,” in *LATINCRYPT,* 2019.
[17] D. Boneh, Y. Ishai, A. Sahai, and D. J. Wu, “Lattice-based snargs and their application to more efficient obfuscation,” in *Proc. EUROCRYPT,* 2017.
[18] D. Boneh, Y. Ishai, A. Sahai, and D. J. Wu, “Quasi-optimal snargs via linear multi-prover interactive proofs,” in *Proc. EUROCRYPT,* 2018.
[19] O. Regev, “On lattices, learning with errors, random linear codes, and cryptography,” *J. ACM*, vol. 56, no. 6, pp. 1–40, 2009.
[20] C. Gentry, C. Peikert, and V. Vaikuntanathan, “Trapdoors for hard lattices and new cryptographic constructions,” in *Proc. STOC*, 2008.
[21] V. Lyubashevsky, C. Peikert, and O. Regev, “On ideal lattices and learning with errors over rings,” *J. ACM*, vol. 60, no. 34, pp. 1–35, 2013.
[22] L. Ducas, V. Lyubashevsky, and T. Prest, “Efficient identity-based encryption over ntru lattices,” in *Proc. ASIACRYPT,* 2014.
[23] R. Gennaro, M. Minelli, A. Nitulescu, and M. Orrù, “Lattice-based zk-SNARKs from square span programs,” in *Proc. ACM CCS*, 2018.
[24] K. Naganuma *et al.*, “Post-quantum zk-SNARK for arithmetic circuits using qaps,” in *Proc. IEEE AsiaJCIS,* 2020.
[25] N. Bitansky, A. Chiesa, Y. Ishai, O. Paneth, and R. Ostrovsky, “Succinct non-interactive arguments via linear interactive proofs,” in *Proc. TCC*, 2013.
[26] G. Danezis, C. Fournet, J. Groth, and M. Kohlweiss, “Square span programs with applications to succinct NIZK arguments,” in *Proc. ASIACRYPT*, 2014.
[27] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, “Quadratic span programs and succinct NIZKs without PCPs,” in *Proc. EUROCRYPT*, 2013.
[28] B. Parno, J. Howell, C. Gentry, and M. Raykova, “Pinocchio: Nearly practical verifiable computation,” in *Proc. IEEE Symposium on Security and Privacy (S&P),* 2013.
[29] C. Ganesh, A. Nitulescu, and E. Soria-Vazquez, “Rinocchio: SNARKs for ring arithmetic,” Cryptology ePrint Archive, Report 2021/322, 2021, https://eprint.iacr.org/2021/322.
[30] Y. Ishai, H. Su, and D. J. Wu, “Shorter and faster post-quantum designated-verifier zkSNARKs from lattices,” in *Proc. ACM CCS*, 2021.
[31] R. C. Merkle, “A digital signature based on a conventional encryption function,” in *Proc. CRYPTO,* 1987.
[32] W. Banaszczyk, “Inequalities for convex bodies and polar reciprocal lattices inR^n,” *Discrete & Computational Geometry*, vol. 13, no. 2, pp. 217–231, 1995.
[33] Z. Brakerski and V. Vaikuntanathan, “Fully homomorphic encryption from ring-LWE and security for key dependent messages,” in *Proc. CRYPTO,* 2011.
[34] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) fully homomorphic encryption without bootstrapping,” *Trans. Comput. Theory*, vol. 6, no. 3, pp. 1–36, 2014.
[35] M. Rosca, D. Stehlé, and A. Wallet, “On the ring-LWE and polynomial-LWE problems,” in *Proc. EUROCRYPT*, 2018.
[36] L. Ducas and A. Durmus, “Ring-LWE in polynomial rings,” in *Proc. PKC*, 2012.
[37] J. T. Schwartz, “Fast probabilistic algorithms for verification of polynomial identities,” *J. ACM*, vol. 27, no. 4, pp. 701–717, 1980.
[38] A. Bishnoi, P. L. Clark, A. Potukuchi, and J. R. Schmitt, “On zeros of a polynomial in a finite grid,” *Combinatorics, Probability and Computing*, vol. 27, pp. 310–333, 2018.
[39] K. Yasunaga, “Replacing probability distributions in security games via hellinger distance,” in *Proc. ITC*, 2021.
[40] A. Langlois, D. Stehlé, and R. Steinfeld, “GGHLite: More efficient multilinear maps from ideal lattices,” in *Proc. EUROCRYPT*, 2014.
[41] S. Bai *et al.*, “Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance,” *J. Cryptology*, vol. 31, no. 2, pp. 610–640, 2018.
[42] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen, “On the hardness of learning with rounding over small modulus,” in *Proc. TCC*, 2016.
[43] D. Micciancio and M. Walter, “Gaussian sampling over the integers: Efficient, generic, constant-time,” in *Proc. CRYPTO*, 2017.
[44] D. Micciancio and M. Walter, “On the bit security of cryptographic primitives,” in *Proc. EUROCRYPT*, 2018.
[45] M. Abboud and T. Prest, “Cryptographic divergences: New techniques and new applications,” in *Proc. SCN*, 2020.
[46] S. Watanabe and K. Yasunaga, “Bit security as computational cost for winning games with high probability,” in *Proc. ASIACRYPT*, 2021.
[47] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, “A pseudorandom generator from any one-way function,” *SIAM J. Comput.*, vol. 28, no. 4, pp. 1364–1396, 1999.
[48] M. R. Albrecht, R. Player, and S. Scott., “On the concrete hardness of learning with errors.” *J. Math. Cryptology*, vol. 9, pp. 169–203, 2015.
[49] Scipr-lab, “https://github.com/scipr-lab/libsnark.”
[50] Microsoft, “https://github.com/microsoft/SEAL.”

Heewon Chung received the B.S. in mathematics from the Korea Advance Institute Science and Technology (KAIST), Daejon, Republic of Korea, in 2009 and M.S. degrees and the Ph.D. degree in Mathematics from Seoul National University, Seoul, Republic of Korea, in 2017. Since 2022, he has been a Cryptographic Researcher in DESILO Inc in Republic of Korea. His recent research interest includes solving scalability problem in blockchain using zero-knowledge proofs (SNARKs, IVC) and practical applications using homomorphic encryption. From 2016 to 2017, he was a Research Assistant with the Agency for Science, Technology, and Research (A*STAR) in Singapore. From 2018 to 2019, he was a Manager with Korea Telecom in Republic of Korea. From 2020 to 2021, he was a Postdoctoral Researcher in Hanyang University, Seoul, Republic of Korea.

Dongwoo Kim received the B.S. and Ph.D. degrees in mathematical sciences from Seoul National University, Seoul, South Korea, in 2013 and 2020, respectively. Since 2023, he has been an Assistant Professor in the Department of AI·SW Convergence, Dongguk University, Seoul, Republic of Korea. From 2020 to 2023, he had been a Principal Engineer in security and cryptography at Western Digital Research, Milpitas, CA, USA. Before that, he has been a Researcher at the Industrial and Mathematical Data Analytics Research Center, Seoul National University. His research interests include the improvement of homomorphic encryption, verifiable computation, and other cryptographic primitives for practical applications.

Jeong Han Kim studied Physics and Mathematical Physics at Yonsei University (Seoul, Korea), and earned his Ph.D. in Mathematics at Rutgers University. He was a researcher at AT&T Bell Labs and at Microsoft Research, and was Underwood Chair Professor of Mathematics at Yonsei University. He is currently a Professor of the School of Computational Sciences at the Korea Institute for Advanced Study.

His main research fields are combinatorics and discrete mathematics. His best known contribution to the field is his proof that the Ramsey number $R(3,t)$ has asymptotic order of magnitude $t^2/\log t$. He received the Fulkerson Prize in 1997 for his contributions to Ramsey theory.

His awards also include Sloan Dissertation Fellowship(1992), Sloan Research Fellowship(1997), Role Model Award for Scientists and Engineers (2007), Kyung-Ahm Prize (2008) and Sam-il Cultural Awards (Natural sciences, 2020).

Jiseung Kim earned his B.S. in Mathematics from Chonnam National University in 2009, followed by a Ph.D. in the Mathematical Sciences from Seoul National University in Seoul, South Korea, in 2020. From 2020 to 2022, he worked as a Research Scientist in the School of Computational Science at the Korea Institute for Advanced Study. He currently serves as an Assistant Professor in the Department of Computer Science and Artificial Intelligence at Jeounbuk National University, located in Jeounju, Republic of Korea. His research interests focus on the mathematical analysis of algebraic lattice-based cryptography and hard problems.