LUNA: Quasi-Optimally Succinct Designated-Verifier Zero-Knowledge Arguments from Lattices

Ron Steinfeld
Monash University
Melbourne, Australia
Ron.Steinfeld@monash.edu

Amin Sakzad
Monash University
Melbourne, Australia
Amin.Sakzad@monash.edu

Muhammed F. Esgin
Monash University
Melbourne, Australia
Muhammed.Esgin@monash.edu

Veronika Kuchta
Florida Atlantic University
Boca Raton, USA
vkuchta@fau.edu

Mert Yassi
Monash University
Melbourne, Australia
Mert.Yassi@monash.edu

Raymond K. Zhao
CSIRO's Data61
Sydney, Australia
raymond.zhao@data61.csiro.au

ABSTRACT

We introduce the *first* candidate Lattice-based designated verifier (DV) zero knowledge sUccinct Non-interactive Argument (ZK-SNARG) protocol, named LUNA, with *quasi-optimal proof length* (quasi-linear in the security/privacy parameter). By simply relying on mildly stronger security assumptions, LUNA is also a candidate ZK-SNARK (i.e. argument of knowledge). LUNA achieves significant improvements in *concrete* proof sizes, reaching below 6 KB (compared to > 32 KB in prior work) for 128-bit security/privacy level. To achieve our quasi-optimal succinct LUNA, we give a new regularity result for 'private' re-randomization of Module LWE (MLWE) samples using discrete Gaussian randomization vectors, also known as a lattice-based leftover hash lemma with leakage, which applies with a discrete Gaussian re-randomization parameter that is *polynomial* in the statistical privacy parameter (avoiding exponential smudging), and hides the coset of the re-randomization vector support set. Along the way, we derive bounds on the smoothing parameter of the intersection of short integer solution (SIS), gadget, and Gaussian perp module lattices over the power of 2 cyclotomic rings. We then introduce a new candidate linear-only homomorphic encryption scheme called Module Half-GSW (HGSW), and apply our regularity theorem to provide smudging-free circuit-private homomorphic linear operations for Module HGSW. Our implementation and experimental performance evaluation show that, for typical instance sizes, Module HGSW provides favourable performance for ZK-SNARG applications involving lightweight verifiers. It enables significantly (around 5×) shorter proof lengths while speeding up CRS generation and encryption time by $4-16\times$ and speeding up decryption time by 4.3×, while incurring just $1.2-2\times$ time overhead in linear homomorphic proof generation operations, compared to a Regev encryption used in prior work in the ZK-SNARG context. We believe our techniques are of independent interest and will find application in other privacy-preserving applications of lattice-based cryptography.

1 INTRODUCTION

Zero-knowledge proof (ZKP) systems were introduced by the authors of [45] in 1985 to allow a prover holding some secret witness w for a statement x satisfying some NP relation R, to prove to a verifier holding x that such a witness w satisfying the relation exists (the soundness property), without revealing any information on w to the verifier beyond that revealed by the statement x (the zero-knowledge property). ZKPs have a myriad of applications in privacy-preserving cryptographic protocols. However, for statements with large witnesses w, the main limitation of classical ZKPs is that their proof size is proportional to the witness size. To support such applications, including verifiable computation [61] and privacy-preserving cryptocurrencies [12] it is desirable to have *succinct* ZKPs in which the proof (or argument) size is only *poly-logarithmic* in the running time of the NP relation's verification algorithm and the witness size. The first such Zero-Knowledge Succinct Non-interactive ARGument (ZK-SNARG)[1] system for NP languages was proposed by Kilian [48]. While the first ZK-SNARGs were theoretical results and resulted in long proofs in practice, significant practical improvements followed over the last decade. The shortest known ZK-SNARG constructions [46] achieve proof lengths in the order of a few hundred bytes but rely on quantum-insecure discrete-log assumptions.

Prior work on quantum-safe and lattice-based ZK-SNARGs. A large number of earlier works (see, e.g., [34, 35, 54] and references therein) focus on building quantum-safe ZKPs, but without succinctness. Towards achieving succinctness, ZK-SNARGs based on quantum-safe assumptions from symmetric-key cryptography are proposed [8, 11, 13, 14], but they currently do not achieve proof lengths below around 100KB for typical security parameters. For ZK-SNARGs based on conjectured quantum-safe lattice problems, there are currently two main approaches. The first approach constructs lattice-based ZK-SNARGs that are publicly verifiable but currently yield long proof sizes [5, 6, 15, 19, 20, 26, 27, 37]. The recent work [15] constructs significantly shorter publicly verifiable (PV) lattice-based ZK-SNARGs (LaBRADOR). However, the verification time of [15] is linear in the witness length and moreover, the LaBRADOR proof lengths reported in [15] for typical security parameters are still an order of magnitude longer than the proof lengths of our designated verifier ZK-SNARGs constructed in this paper for similar security parameters. The concrete proof length for other recent lattice-based PV SNARG constructions we cite above are expected to be longer than LaBRADOR due to the larger asymptotic lattice 'stretch' and 'slack' factors for the underlying

[1] For most of the remainder of the paper, we write ZK-SNARGs for simplicity; however most of the discussion also applies to ZK-SNARKs, i.e. Zero-Knowledge Succinct Non-interactive ARguments of Knowledge.

polynomial commitment schemes; see the comparison in Table 1 of [6]. The second approach, which we focus on in this paper, constructs lattice-based *Designated-Verifier* (DV) ZK-SNARGs, which require a preprocessing setup procedure by a designated-verifier run before the relations to be proved are known. In such a preprocessing DV (DV for short) model of ZK-SNARGs, proofs can only be verified by the DV holding a secret verification key. DV ZK-SNARGs still suffice for important privacy-preserving applications such as verifiable computation and indistinguishability obfuscation [9, 17]. The first lattice-based DV SNARG following the latter approach was introduced by Boneh et al. [17], and this lattice-based DV SNARG approach was later improved by [40, 47, 60][2]. The approach in these works constructs a DV SNARG using a cryptographic compiler introduced by Bitansky et al. [16], from two building blocks: (1) a *linear-only* (LO) homomorphic vector encryption scheme (i.e. a homomorphic encryption scheme with vector plaintexts where only linear homomorphic operations are computationally feasible). Throughout the rest of this paper, we will use the simulation-based *Linear Targeted Malleability* (LTM) flavour [16, 17] of LO security notion, rather than knowledge-based LOH assumptions that have been shown invalid by recent attacks for LWE-based encryption, see Appendix A for more discussion. (2) a Linear Probabilistically Checkable Proof (LPCP) system. Authors of [16] observed that if the linear-only encryption scheme satisfies a *re-randomization* property (so that the randomness in a ciphertext can be re-randomized without the secret key to produce a fresh ciphertext), then their compiler can produce a DV ZK-SNARG, i.e. a SNARG satisfying the zero-knowledge privacy property.

The work of [17] instantiated the candidate linear-only vector encryption from the lattice-based Regev encryption scheme. A follow-up work on quasi-optimal SNARGs was proposed by Boneh et al. in [18] and provided a construction for Boolean circuits from a Multi-Prover Interactive Proof (MIP) system. The main advantage over the first result in [17], this SNARG construction is the reduction of computational overhead on the prover side. Although achieving sub-optimal proof length, these lattice-based constructions do not provide succinct re-randomizable ciphertexts, so those constructions only provide plain SNARGs, but not ZK-SNARGs (i.e. no zero-knowledge property).

Gennaro et al. [40] introduced the first lattice-based construction of SNARGs that also achieved the zero-knowledge property and is built from square span programs (SSP). However, the proof size is very large, exceeding 0.5 GB. Nitulescu [60] presented a lattice ZK-SNARG from a quadratic arithmetic program (QAP), which is defined for arithmetic circuit satisfaction.

The state-of-the-art work on lattice-based DV ZK-SNARGs by authors of [47] (called ISW21 or simply ISW from hereon) provided a new construction of a shorter ZK-SNARG from LPCP for rank-one constraint systems (R1CS) using new approaches. An important new ingredient for the concrete proof succinctness of the ISW construction versus earlier lattice-based constructions is the use of a large *extension field* plaintext space for the underlying linear-only Regev encryption scheme (where the large extension field size provides a low ZK-SNARG soundness error), while keeping the field characteristic moderately small. The smaller field characteristic for fresh ciphertexts gives a smaller fresh ciphertext modulus length and leads to shorter proofs[3]. To support extension field plaintext spaces for Regev encryption, ISW uses a structured lattice variant of Regev encryption based on the hardness of the Module Learning With Errors (MLWE) problem over a polynomial ring R. The main advantage of the ZK-SNARG in [47] is a significant reduction of the proof size compared to earlier work in [40].

Lattice ZK-SNARG succinctness problem: smudging-based ZK. However, even with the improvements of the ISW scheme, the resulting ZK-SNARG proof length remains significantly higher than one would like, both from an asymptotic theoretical view, as well as a practical concrete parameters view. In particular, from the asymptotic theoretical view, the ZK-SNARG proof length in ISW is quadratic in the security parameter λ. For our asymptotic security analysis in this paper, we set the statistical ZK privacy security parameter κ of ISW to be equal to the computational soundness security parameter λ so there is just a single security/privacy parameter λ. For concrete estimates, ISW set the statistical privacy parameter to a low figure of $\kappa = 40$, but this would potentially allow ZK attacks with a non-negligible advantage 2^{-40}; ideally, one would want $\kappa \approx 128$ to match the typical soundness security parameter λ. In any case, this does not affect asymptotic estimates when κ is linear in λ. This is suboptimal, as one could hope to have a proof length quasilinear in λ (i.e. linear up to polylog factors). Also from a practical concrete parameter view, the shortest proof lengths in ISW are more than 15KB (even for a relatively low statistical ZK security parameter $\kappa = 40$) which is still about 20× the ciphertext length of the standard MLWE-based Kyber encryption scheme [21] for typical 128-bit soundness security parameter (the ISW proof length would increase significantly more if we aim, as in this paper, for a more standard privacy parameter such as $\kappa = 128$, see Table 1).

The main reason behind the suboptimal proof length of ISW and prior lattice-based ZK-SNARGs is the use of the *exponential smudging* technique to circumvent the difficulty of re-randomizing lattice-based ciphertexts of the underlying LO encryption scheme E for achieving circuit privacy of the underlying encryption scheme when used inside the Bitansky et al. [16] DV ZK-SNARG compiler. In particular, the first step of the SNARG prover algorithm in ISW and schemes based on the compiler in [16] consists in computing a linear combination c of fresh ciphertexts $\{c_i = E(\mu_i, e_i)\}_i$ from the preprocessing step: $c = \sum_i a_i \cdot c_i = \sum_i a_i \cdot E(\mu_i, e_i) = E(\sum_i a_i\mu_i, \sum_i a_i e_i)$. Here, the plaintexts μ_i are the verifier's query

[2] We remark that the works [40, 47] aim to construct quantum-safe arguments of *knowledge* (i.e. ZK-SNARKs), where the knowledge soundness relies on a knowledge-based linear-only (LO) hardness assumption on the underlying homomorphic encryption scheme. However, this lattice *LWE knowledge* assumption has very recently been shown to be invalid against quantum attacks [31]. Still, as shown in [16, 17] and remarked in [31], the same protocols [40, 47, 60], as well as our protocol in this paper, are also ZK-SNARGs with soundness under a weaker flavour of LO lattice assumption on the homomorphic encryption scheme called (statistically simulatable, strict) *Linear Targeted Malleability* (LTM), which is unaffected by the quantum attacks of [31]. Moreover, with a stronger (computationally simulatable) LTM assumption on the encryption scheme (also seemingly unaffected by the attacks in [31]), and a LPCP with knowledge-based soundness, the above protocols (and our LUNA) are also ZK-SNARKs [16]. We refer to Sec. 5.1 and Appendix A for further discussion.

[3] The reduction in proof length arises due to the harder underlying lattice problem with a smaller fresh ciphertext modulus, allowing a smaller lattice dimension parameter. The fresh ciphertext modulus does not directly impact proof lengths, as the ISW construction uses modulus switching techniques to reduce the final proof ciphertext modulus size.

SNARG scheme	Base encryption scheme	Quasi-opt. proof size (Yes/No)	Size CRS (GB) (compressed, full)	Proof (KB)	ZK property (Yes/No)	ZK technique	Base Encryption Runtimes (in secs) Setup +Enc	Add	Decrypt
BISW17 [17]	LWE-Regev	Yes	-	-	No	N/A	-	-	-
BISW18 [18]	RLWE-Regev	Yes	-	-	No	N/A	-	-	-
GMNO18 [40]	LWE-Regev	No	?*	?*	Yes	exp. smudging	-	-	-
ISW21 [47]	MLWE-Regev	No	(0.65, 21)	32.6	Yes	exp. smudging	626	11	0.0030
LUNA (Our work)	MLWE-HGSW	Yes	(0.63, 11) (1.25, 18) (2.06, 28)	8.3 5.8 5.6	Yes	poly. rerandom.	159	22	0.0007

Table 1: Comparison of lattice (ZK-)SNARGs for R1CS size of $N_g = 2^{16}$ for both soundness and zero-knowledge (if applicable) security level at 128 bits. Here, we estimated the sizes for the ISW21 protocol [47] for $\kappa = 128$ bit ZK privacy level, by extending the 'Shorter Proofs' parameters in [47] for $\kappa = 40$ bit ZK privacy level, while keeping their parameter choices for the initial noise ($s = 64$) and plaintext space modulus ($p = 2^{13} - 1$) (see Sec. 5.2 for further details). The two sizes given in CRS are those for compressed and non-compressed versions, respectively. The former version ignores the uniformly random part of the CRS that can be generated from a small seed. See Table 2 for more on parameter settings. The Setup, Prove and Verify computations of ZK-SNARG roughly correspond to Setup+Enc, Add, Decrypt computations of the base encryption, respectively. See Sec. 6 for more on implementation and runtimes. *We note that [47] pointed out that the suggested parameters in [40] provide only 15 bits of provable soundness. Therefore, in our table, we skip the proof and CRS sizes for [40].

challenges in the underlying linear PCP and e_i is the corresponding fresh randomness used to encrypt μ_i. The coefficients a_i are computed using the underlying linear PCP from the prover's witness. In the underlying SNARG with no ZK privacy, the proof consists of c, and the verifier knowing the decryption key for E can decrypt c to get the plaintext $\mu := \sum_i a_i \mu_i$, which can then be verified using the underlying linear PCP verification. However, as the decryption key is known to the verifier, c may also reveal the final ciphertext randomness $e := \sum_i a_i e_i$ to the verifier; this may in turn leak additional information about the prover's witness beyond what is revealed in μ and invalidate the ZK property. To prevent this leakage and obtain the ZK property, the exponential smudging technique is used in ISW and earlier lattice-based schemes consists in the prover masking e by adding an independent masking randomness e' and sending $c' = c + E(0, e') = E(\mu, e + e')$ as the proof. However, in lattice-based schemes, the $e + e'$ addition is over integer vectors, so to obtain κ-bit statistical ZK privacy with this smudging method[4], the size (standard deviation) of the entries of the smudging term e' must exceed the size of the entries of e by a factor exponential in κ. This exponential smudging then leads to ciphertext and hence ZK-SNARG proof lengths of at least $\Omega(\kappa\lambda) = \Omega(\lambda^2)$ assuming that $\kappa = \theta(\lambda)$. The above problem with ISW leads us to ask the following main open questions:

From a theoretical viewpoint, can we construct candidate lattice-based ZK-SNARGs with proof length quasilinear in the security parameter $\lambda = \kappa$? From a practical viewpoint, can we construct candidate ZK-SNARGs with concretely shorter proofs than those of ISW?

Our main goal in this paper is to address these questions, focusing on the minimization of ZK-SNARG proof length, perhaps by trading off other aspects, such as the computational runtimes.

Directions and challenges.

The first direction is to devise a more efficient method for circuit privacy of the underlying linear-only encryption scheme E, without resorting to exponential smudging. A natural approach is to look at circuit privacy techniques developed for lattice-based fully homomorphic encryption (FHE) schemes. Gentry's technique [41] relies on exponential smudging. The later works [23, 33] provide FHE circuit privacy without exponential smudging. However, [33] crucially relies on FHE-based bootstrapping, which is incompatible with LO encryption required for our ZK-SNARG setting, while [23] relies on the use of a general Leftover Hash Lemma with Leakage (LHLL), which applies over *unstructured* LWE over $\mathbb{Z}_q$, but not over polynomial rings R_q used in efficient constructions. Two recent concurrent and independent works to ours [22, 50] introduce extensions of the LHLL re-randomization result from [23], applied in the context of circuit privacy for FHE (rather than our 'Half-GSW' ZK-SNARG context). However, the result in [22] applies over R_q but is restricted to non-standard power-of-2 q, while the result of [50] is restricted to $\mathbb{Z}_q$ rather than R_q. We refer the reader to Appendix C for further discussion of these and other related works on re-randomization.

A second direction towards answering the above open questions was suggested by ISW [47], who asked whether the circuit privacy requirement for the underlying encryption scheme E and its associated smudging technique is needed for the ZK property of the resulting SNARG constructed with the compiler in [16] from the QAP-based Linear PCP (LPCP) used in [47]. In particular, ISW defined an 'honest-verifier ZK with leakage' (HVZKL) property for the underlying LPCP. This property essentially asserts that the ZK property of the LPCP is preserved even in the presence of the leakage of the final randomness e revealed to the verifier when no smudging is used in ISW SNARG. They observed that if this HVZKL property is satisfied for the underlying LPCP, then the ISW SNARG with no smudging achieves ZK. If the latter is true, it would give shorter ZK-SNARG proofs.

1.1 Our Contributions

In this paper, we make progress on the open question of constructing lattice-based ZK-SNARGs with quasi-optimal succinct proofs, addressing both directions mentioned above.

LUNA: a new candidate Lattice-based sUccinct Non-interactive Argument with quasi-optimal succinct proofs. Our main result addresses the first direction discussed above. We construct the first candidate Lattice-based sUccinct Non-interactive Arguments (LUNA) with quasi-optimal succinct proofs, namely proof length quasi-linear in the security parameter λ. By simply relying on mildly stronger security assumptions, LUNA is also a candidate ZK-SNARK, i.e. argument of knowledge, (see Sec. 5.1 and

[4] i.e. to make the distribution of $e + e'$ within statistical distance $\leq 2^{-\kappa}$ of that of e'

Appendix A). LUNA avoids the use of exponential smudging for achieving the ZK property with its inherent inefficient parameters. Following the first direction discussed above, we address the technical challenges of constructing a LO encryption scheme E with a circuit-private re-randomization procedure that does not require exponential smudging and preserves the algebraic structure of Module LWE needed in the ISW ZK-SNARG construction. Our E is derived from a suitable modification of a Module LWE variant of the GSW homomorphic encryption scheme [44]. We compare the main properties of LUNA to prior lattice-based SNARG constructions in Table 1. LUNA not only achieves the shortest asymptotic proof length to date in theory but also gives concrete practical savings of more than 5× in proof length versus ISW at the same security/privacy level of 128 bits. More concretely, our approach can achieve almost 4× proof size and 2× full CRS size reduction while having also smaller compressed CRS size. For increasing CRS sizes, the proof size can be further reduced as shown in Table 1. Our proof lengths aimed at 128-bit security/privacy level are even smaller than the 15-20KB proof lengths of the original ISW protocol at 40-bit privacy level. To obtain our main result, we introduce new tools of independent interest described next.

New regularity results for private re-randomization of MLWE samples. Our main technical contribution is a new regularity theorem for the following 'gadget-based' private re-randomization of Module LWE (MLWE) samples over the standard polynomial ring $R_q := \mathbb{Z}_q[x]/(x^d+1)$ (for d a power of 2) without smudging. Let $\boldsymbol{G}$ denote a 'power of 2' gadget matrix [57]. Take a set of MLWE samples $(\boldsymbol{A}, \boldsymbol{B})$ with $\boldsymbol{B} = \boldsymbol{AS}+\boldsymbol{E}$ over the polynomial ring $R_q := \mathbb{Z}_q[x]/(x^d+1)$ with q prime (or a product of two primes), where $\boldsymbol{A}$ is a uniformly random MLWE matrix, $\boldsymbol{S}$ is the MLWE secret matrix and $\boldsymbol{E}$ the small MLWE error matrix. The re-randomized MLWE sample is computed as $(\boldsymbol{u}^T, \boldsymbol{v}^T) := (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{B} + \boldsymbol{y}^T) = (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{AS} + \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T)$, where $\boldsymbol{x}$ is a re-randomization vector sampled from a discrete Gaussian distribution with small width parameter r satisfying $\boldsymbol{x}^T\boldsymbol{G} = \boldsymbol{a}^T$ for some scaling vector $\boldsymbol{a}^T$ and $\boldsymbol{y}$ is an independent discrete Gaussian with same parameter r. In the application to homomorphic scaling of GSW ciphertexts, the scaling vector $\boldsymbol{a}^T$ contains the homomorphic scaling factors for a corresponding vector of plaintexts. Our regularity result shows that the distribution of the re-randomized MLWE sample $(\boldsymbol{u}^T, \boldsymbol{v}^T)$ is statistically close to a distribution that is *independent* of the scaling vector $\boldsymbol{a}^T$ (ensuring circuit privacy in the Module GSW homomorphic scaling application), and moreover, the latter statistical distance can be made exponentially small ($\leq 2^{-\kappa}$) in the desired statistical security parameter κ for some *polynomial* choice of Gaussian parameter $r = \mathsf{poly}(\kappa)$. Our new result avoids the exponential blowup ($r = 2^{\Omega(\kappa)}$) in smudging-based re-randomization results as used in [47]. We therefore obtain a Module LWE analogue of the regularity theorem for private re-randomization of (unstructured) LWE samples in [23]. The latter LWE-based regularity result over $\mathbb{Z}_q$ uses general leftover hash lemmas over *fields* and does not directly extend to MLWE over rings R_q with non-trivial sub-ideals. Technically, our regularity proof for MLWE requires different and more involved lattice smoothing-based techniques to deal with this issue (see overview in Sec. 3). In particular, as a core result underlying our LHL with leakage (LHLL), which may be of independent interest, we study the smoothing parameter of the intersection of the three underlying perp lattices associated with the matrices $\boldsymbol{A}, \boldsymbol{E}, \boldsymbol{G}$. Along the way, we also give a simple lower bound on the minimum of the well-known Gadget primal lattice $\Lambda_q(\boldsymbol{G})$, which to our knowledge, has not explicitly appeared in the literature and may also be of independent interest. We refer to App. C for a further discussion of LHLL related work.

Half GSW and application to ZK-SNARGs. We present a new candidate LO (in the sense of *Linear Targeted Malleability* [16, 17], or LTM for short) vector encryption scheme with succinct ciphertexts that we call *Half GSW* (HGSW), whose IND-CPA security is based on the hardness of MLWE. Our HGSW scheme is obtained via simple modifications to an MLWE variant of the GSW fully homomorphic encryption scheme that involves removing a portion (typically half) of the GSW ciphertext. Our modifications of GSW are designed to remove the undesirable (in the context of linear-only encryption needed in ZK-SNARG applications) multiplicative homomorphism while supporting *succinct* circuit-private homomorphic linear scaling based on our above MLWE re-randomization regularity result, with ciphertext length quasilinear in the security and circuit privacy parameter $\lambda = \kappa$. Similar to previous candidate LO lattice-based encryption schemes (e.g. [17, 18, 47]), the LO property of our HGSW scheme relies on a plausible conjecture that we call 'HGSW Linear Targeted Malleability' (we remark that our HGSW LTM conjecture is simulation-based, and unaffected by recent attacks [31, 67] on LWE knowledge-based assumptions; see Appendix A for further discussion). This conjecture enjoys a 'win-win' flavour; if the conjecture turns out to be false, it is likely to imply more succinct somewhat homomorphic encryption schemes (as HGSW is more succinct than GSW). We note that our HGSW scheme can also be viewed as a collection of ciphertexts of the Regev encryption scheme for the message vector $\mu\boldsymbol{g}^T$, where $\boldsymbol{g}^T = (1, 2, \ldots, 2^{m_q-1})$ with $m_q = \log_2 q$ is the power of 2 gadget vector. Note that our optimized construction uses powers of some integer $\beta > 2$ in the gadget vector, to reduce CRS length. Thus the HGSW construction itself is not new, and indeed such an encryption scheme has been used in other contexts, e.g. [42]. However, to our knowledge, our work is the first application of such an encryption scheme in the context of *linear-only* encryption.

HGSW Implementation and Performance Evaluation of LUNA. To demonstrate the practicality of our HGSW encryption scheme in the context of ZK-SNARGs, we implemented HGSW and evaluated its performance for typical parameters[5]. Our implementation takes advantage of fast NTT-based ring arithmetic. Thanks to the shorter LWE dimension and modulus parameters enabled by our re-randomization result (avoiding the exponential smudging blowup in [47]) and our fast ring arithmetic, our Module HGSW implementation provides a performance *speedup* of $4 - 16\times$ versus the ISW Regev encryption implementation used in [47]'s ZK-SNARG, at the same 128-bit security level, for the total time of generating keys and encrypting the CRS in the ZK-SNARG setup phase, and a speed-up in decryption time by 4.3×. Our Module HGSW implementation incurs a $1.2 - 2\times$ time overhead for 128-bit security compared to the Regev encryption implementation used in the

[5] Our implementation is available at https://github.com/yassimert/LUNA

ISW ZK-SNARG [47], when used to perform homomorphic operations in generating ZK-SNARG proofs for R1CS instance sizes N_g up to 2^{20}. This overhead in proof generation is due to the gadget lattice Gaussian sampling re-randomization procedure in HGSW. However, for many DV ZK-SNARG applications, such as verifying delegated computations, where the prover is a powerful server and the client may be lightweight, we believe this may be an acceptable encryption performance overhead, considering the significantly shorter proof lengths sent to the verifier and faster verification in our HGSW-based LUNA.

Attack on the ZK property of the short SNARG in [47] with no smudging. As a bonus contribution, we present in Appendix D a negative result in the second direction mentioned above, namely a simple attack (based on heuristic assumptions) on the Zero-Knowledge property of the SNARG in ISW [47] with no smudging, instantiated with the QAP LPCP presented in [47]. Our attack demonstrates that the Honest Verifier ZK with Leakage (HVZKL) property assumed in [47] does not hold in general for existing LPCPs. To bypass this issue, LUNA does not rely on HVZKL of the LPCP.

Roadmap. In Sec. 2, we provide preliminaries (see supp. materials for further definitions). In Sec. 3, we present our new regularity results for private re-randomization of MLWE. Sec. 4 contains our construction of the module-based Half-GSW (HGSW) scheme. In Sec. 5, we apply HGSW and our re-randomization results to lattice-based ZK-SNARGs and ZK-SNARKs, and present LUNA and its concrete parameters. Sec. 6 presents our HGSW implementation and evaluation results. Some proofs and our attack are provided in the Appendices.

2 PRELIMINARIES

We denote column vectors by bold lower case and matrices by bold upper case. For a column vector $\boldsymbol{x}$, we denote the corresponding row vector by $\boldsymbol{x}^T$. For a matrix $\boldsymbol{M}$ we use $\|\boldsymbol{M}\|$ (resp. $\|\boldsymbol{M}\|_\infty$) to denote the maximal Euclidean norm (resp. infinity norm) over all rows of $\boldsymbol{M}$. The integer set $\{1, \dots, n\}$ is denoted by $[n]$. The zero matrix and identity matrix of dimensions $m \times n$ and n are denoted by $\boldsymbol{0}^{m\times n}$ and $\boldsymbol{I}_n$, respectively. The transpose and inverse operations on a matrix $\boldsymbol{M}$ are written as $\boldsymbol{M}^T$ and $\boldsymbol{M}^{-1}$, respectively. For a distribution $\mathcal{D}$, we write $x \hookleftarrow \mathcal{D}$ to say that x is sampled from $\mathcal{D}$. For an algorithm A, we use $a \leftarrow A$ to show the output of A is assigned to a. We use $\mathcal{U}(X)$ to denote a uniform distribution over X. We denote the base 2 and natural logarithm by log and ln, respectively. For a lattice Λ, a shift vector $\boldsymbol{c}$ and $s > 0$, we denote by $\mathcal{D}_{\Lambda+\boldsymbol{c},s}$ the discrete Gaussian distribution on the coset $\Lambda + \boldsymbol{c}$ of Λ with parameter s (see App. B).

2.1 Lattice Preliminaries

Lattices. A n-dimensional lattice Λ is a discrete additive subgroup of $\mathbb{R}^n$. For an integer $t \le n$ and a basis matrix $\boldsymbol{B} \in \mathbb{R}^{n\times t}$ of rank t, $\Lambda(\boldsymbol{B}) = \{\boldsymbol{Bx} \in \mathbb{R}^n | \boldsymbol{x} \in \mathbb{Z}^t\}$ is the lattice generated by the column vectors (i.e. basis vectors) of $\boldsymbol{B}$. If $n = t$, the lattice $\Lambda(\boldsymbol{B})$ is called full-rank. The *dual* lattice Λ^* of lattice Λ is defined as $\Lambda^* := \{\boldsymbol{w} \in \mathbb{R}^n : \forall \boldsymbol{v} \in \Lambda, \boldsymbol{w}^T\boldsymbol{v} \in \mathbb{Z}\}$. For $i \in [t]$, the i'th successive minimum $\lambda_i(\Lambda)$ is defined as $\lambda_i(\Lambda) := \inf\{r : \dim(\mathrm{Span}(\Lambda \cap B(r))) \ge i\}$, where $B(r)$ denotes the closed zero-centered Euclidean ball of radius r.

Definition 1 (q-ary Lattices). *For any positive integer $n \le l$ and q, and matrix $\boldsymbol{A} \in \mathbb{Z}_q^{l\times n}$ define the following l-dimensional lattices:*

$$\Lambda_q^\perp(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^l | \boldsymbol{x}^T\boldsymbol{A} = \boldsymbol{0} \bmod q\},$$

$$\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^l | \boldsymbol{v} = \boldsymbol{As} \bmod q, \text{ for some } \boldsymbol{s} \in \mathbb{Z}^n\}.$$

Definition 2 (Module Learning With Errors (MLWE) [51]). *Let λ be a fixed security parameter and $n = n(\lambda), l = l(\lambda), q = q(\lambda), d = d(\lambda)$, where d is a power of two. Let $R = \mathbb{Z}[x]/(x^d + 1)$ and $R_q = R/qR$ and $\chi = \chi(\lambda)$ be an error distribution over R_q. The (decisional) module learning with errors (MLWE) assumption $MLWE_{n,l,d,q,\chi}$ states that for $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{l\times n})$, $\boldsymbol{s} \hookleftarrow \chi^n$, $\boldsymbol{e} \hookleftarrow \chi^l$ and $\boldsymbol{u} \hookleftarrow \mathcal{U}(R_q^l)$ the following two distributions are indistinguishable*

$$(\boldsymbol{A}, \boldsymbol{As} + \boldsymbol{e}) \ \textit{and} \ (\boldsymbol{A}, \boldsymbol{u}).$$

Definition 3 (The g_{rand}^{-1} Algorithm [57]). *Let $\beta, q \in \mathbb{Z}$, with $\beta \ge 2$, $\boldsymbol{g}^T = (1, \beta, \beta^2, \dots, \beta^{m_q-1}) \in R^{1\times m_q}$, where $m_q = \lceil \log_\beta q \rceil$. There is a randomized, efficiently computable function $\boldsymbol{g}_{\mathrm{rand}}^{-1}(\cdot) : R_q \to R^{1\times m_q}$ such that $\boldsymbol{x}^T \leftarrow \boldsymbol{g}_{\mathrm{rand}}^{-1}(a)$ is sampled from a discrete Gaussian distribution with parameter r, such that $\boldsymbol{x}^T\boldsymbol{g} = a \mod q$ (i.e. $\boldsymbol{x} \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\boldsymbol{g})+\mathbf{c},r}$, where $\mathbf{c}$ is any fixed vector satisfying $\mathbf{c}^T\boldsymbol{g} = a \mod q$). Note that the output of $\boldsymbol{g}_{\mathrm{rand}}^{-1}$ is always a row vector. We extend the definition to vector inputs where $\boldsymbol{g}_{\mathrm{rand}}^{-1}$ is applied to each entry and the result is concatenated. Furthermore, let $\mathbf{G} = I_\rho \otimes \boldsymbol{g} \in R_q^{L\times\rho}$ with $\rho := L/m_q$. For a vector $\boldsymbol{a} \in R_q^\rho$, the vector $\boldsymbol{x}^T = \boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{a})$ satisfy $\boldsymbol{x}^T\mathbf{G} = \boldsymbol{a} \mod q$, where $\boldsymbol{x} \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\mathbf{G})+\mathbf{c},r}$ and $\mathbf{c}$ is any fixed vector satisfying $\mathbf{c}^T\mathbf{G} = \boldsymbol{a} \mod q$.*

Definition 4 (Smoothing Parameter [58]). *For an n-dimensional lattice $\Lambda \subseteq \mathbb{Z}^n$ and a positive real $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest real $r > 0$, such that $\rho_{1/r}(\Lambda^* \setminus \{\boldsymbol{0}\}) \le \epsilon$.*

3 NEW REGULARITY RESULTS FOR PRIVATE RE-RANDOMIZATION OF MLWE SAMPLES

Overview. We now give an overview of our re-randomisation Theorem 1 and its proof. The theorem asserts that given MLWE samples $(\boldsymbol{A}, \boldsymbol{B} = \boldsymbol{AS} + \boldsymbol{E}) \in R_q^{L\times n} \times R_q^{L\times\ell'}$ and a coset-supported discrete Gaussian re-randomisation vector $\bar{\boldsymbol{x}}^T := (\boldsymbol{x}^T, \boldsymbol{y}^T) \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(G)+\boldsymbol{c},r} \times \mathcal{D}_{R^{\ell'},r}$ for some coset vector $\boldsymbol{c}$, the re-randomised MLWE sample $(\boldsymbol{A}', \boldsymbol{B}') = (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{B} + \boldsymbol{y}^T) = (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{AS} + \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T)$ has a distribution that is, up to a negligible statistical distance, independent of the coset vector $\boldsymbol{c}$. We prove Theorem 1 below using the subsequent Lemmas 1–5. The proof of the Theorem consists of two parts presented in the following two subsections: (1) The first part (Part 1: LHL over R_q with leakage), given in Lemma 1, shows that the re-randomized matrix $\boldsymbol{A}' = \bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q$ (here, $\bar{\boldsymbol{A}}$ denote $\boldsymbol{A}$ with $\boldsymbol{0}$ matrix appended at bottom ℓ' rows) is statistically close to uniform over R_q with respect to the short randomizing randomness $\bar{\boldsymbol{x}}$, even conditioned on the leakage on $\bar{\boldsymbol{x}}$ given by $\bar{\boldsymbol{x}}\bar{\boldsymbol{E}} = \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T$, and (2) The second part (Part 2: Gaussian LHL), given in Corollary 5, shows that the distribution of the leakage component $\boldsymbol{x}^T\bar{\boldsymbol{E}}$ is statistically close to a skewed discrete Gaussian. This part 2 is followed by a natural adaptation of the arguments in [23] to the structured ring case, so we focus in this overview on part (1).

The proof of part (1) analyzes the uniformity of the conditional distribution of $\boldsymbol{x}^T\boldsymbol{A} \bmod q$, given $\boldsymbol{E}' := \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T$ and $(\boldsymbol{A}, \boldsymbol{E})$. Prior work [23] on an analogous regularity with leakage result over the field $\mathbb{Z}_q$ for q prime applied a generalized leftover hash argument [32] based only on the entropy of $\boldsymbol{x}^T$ and the leaked entropy in $\boldsymbol{E}'$. Such an entropy-based approach fails over a ring with non-trivial subideals such as our ring R_q (for instance, if all coordinates of $\boldsymbol{x}^T$ are in such a subideal of R_q, then so will $\boldsymbol{A}'$ and uniformity over R_q will not be achieved).

Instead, we give a very different and more technically involved proof from [23], that applies over R_q, based on lattice discrete Gaussian smoothing arguments [58] and directly analyzing the uniformity of the conditional distribution in scope using the relevant geometric smoothing bounds of the underlying lattices. Our proof combines and extends prior smoothing bounds for related lattices. In particular, we show that the conditional distribution in scope $D(\boldsymbol{v}|\boldsymbol{e}) := \Pr[\boldsymbol{x}^T\boldsymbol{A} = \boldsymbol{v}^T | \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T = \boldsymbol{e}^T]$ is proportional to a sum of a Gaussian with parameter r over a coset $\boldsymbol{x}_0^T + \Lambda'$ of the lattice $\Lambda' := \Lambda_q^{\perp}(\bar{\boldsymbol{A}}) \cap \Lambda^{\perp}(\bar{\boldsymbol{E}}) \cap \Lambda_q^{\perp}(\bar{\boldsymbol{G}})$. The lattice Λ' consists of the intersection of the orthogonal q-ary (also known as 'SIS') lattices corresponding to the matrices $\boldsymbol{A}$ and $\boldsymbol{G}$ and the orthogonal lattice (over $\mathbb{Z}$, not mod q) of $\bar{\boldsymbol{E}} := \begin{pmatrix} \boldsymbol{E} \\ \boldsymbol{I}_{\ell'} \end{pmatrix}$, where the coset $\boldsymbol{x}_0^T$ depends on $\boldsymbol{v}, \boldsymbol{e}$ and $\boldsymbol{c}$, and $\bar{\boldsymbol{G}}$ denote $\boldsymbol{G}$ with $\boldsymbol{0}$ matrix appended at bottom ℓ' rows. Smoothing arguments [58] imply that the above coset Gaussian sum is almost independent of the coset if r exceeds the smoothing parameter $\eta(\Lambda')$ of the intersection lattice Λ'.

The uniformity proof for our distribution, therefore, reduces to studying upper bounds on $\eta(\Lambda')$. To our knowledge, the smoothing parameter for such intersection lattices has not been previously studied, and it does not seem possible to give a 'black box' bound on the smoothing parameter of intersection lattices from bounds on the underlying lattices being intersected. Instead, we provide such a novel bound by careful usage of the properties of the underlying lattices and extension of bounds on their minima. To do so, we generalize a transference bound approach used in [2] to study the smoothing parameter of $\Lambda^{\perp}(\bar{\boldsymbol{E}})$ (not mod q) by looking at the corresponding q-ary lattice and its dual. Namely, our proof gives an upper bound for the smoothing parameter $\eta(\Lambda')$ of the rank Ld lattice Λ' using an upper bound on the Ld-minimum $\lambda_{Ld}(\Lambda)$ of the $(L + \ell')d$-dimensional q-ary lattice $\Lambda := \Lambda_q^{\perp}(\bar{\boldsymbol{A}}) \cap \Lambda_q^{\perp}(\bar{\boldsymbol{E}}) \cap \Lambda_q^{\perp}(\bar{\boldsymbol{G}})$, which in turn reduces by a transference bound to lower bounding the $\ell'd + 1$ minimum of the dual lattice Λ^*. The dual lattice Λ^* is a scaled *sum* $\Lambda_q(\boldsymbol{A}) + \Lambda_q(\bar{\boldsymbol{E}}) + \Lambda_q(\boldsymbol{G})$ of the three underlying dual lattices. Note that the $\ell'd+1$ minimum of Λ^* may be lower than that of any one of the underlying three lattices due to cancellations in the sum. To handle this difficulty, our proof considers three subcases of lattice vectors of the form $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{v}_A + \boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G$ in Λ^* achieving the $\ell'd + 1$ lattice minimum, and lower bounding the norm of $\boldsymbol{w}$ in each subcase:

1 **Subcase 1**: $\boldsymbol{v}_A \neq 0 \bmod q$. Here, we lower bound the norm of $\boldsymbol{w}$ by extending the probabilistic union bound argument for lower bounding $\Lambda_q(\boldsymbol{A})$ over modules in [59, 65]; the components $\boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G$ are handled by including those components in the union probability argument.

2 **Subcase 2**: $\boldsymbol{v}_A = 0 \bmod q$ and $\boldsymbol{v}_E$ has 'small' norm compared to q. Here, if $\boldsymbol{v}_G \neq 0$ then the norm of $\boldsymbol{w}$ differs by the 'small' error $\boldsymbol{E}\boldsymbol{v}_E$ from the minimum of the Gadget lattice $\Lambda_q(\boldsymbol{G})$; we lower bound the latter Gadget lattice minimum by a direct argument, which may be of independent interest. If $\boldsymbol{v}_G = 0$, $\boldsymbol{w}$ is in the column span of the rank $\ell'd$ matrix $\boldsymbol{E}$ and cannot be an $\ell'd + 1$th minimum (except with neg. probability if a wrap around mod q occurs), as also observed in [64].

3 **Subcase 3**: $\boldsymbol{v}_A = 0$ and $\boldsymbol{v}_E$ is 'large' compared to q. Due to the identity matrix at the bottom of $\boldsymbol{E}$, this causes bottom part of $\boldsymbol{w}$ to be large.

Application of the re-randomization Theorem. In the next Section, we will apply this Theorem for circuit-private linear homomorphic computation of our HGSW encryption scheme. Namely, given a block of ν ciphertexts $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_\nu$ for message vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_\nu$, we will compute a ciphertext $\boldsymbol{C}$ for the linear combination message vector $\boldsymbol{\mu} = \sum_i a_i \boldsymbol{\mu}_i$ as $\boldsymbol{C} = \sum_i \boldsymbol{x}_i^T \boldsymbol{C}_i + \boldsymbol{y}^T$. Here, each randomized vector $\boldsymbol{x}_i^T$ encodes the corresponding scaling coefficient a_i by sampling $\boldsymbol{x}_i^T$ from a discrete Gaussian over the set of solutions to $\boldsymbol{x}_i^T \boldsymbol{g} = a_i$, i.e the Gadget lattice coset $\Lambda_q^{\perp}(\boldsymbol{g}) + \boldsymbol{c}_i$, where $\boldsymbol{c}_i$ is any solution to $\boldsymbol{c}_i^T \boldsymbol{g} = a_i$. The vector $\boldsymbol{c} = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_\nu)$ in Theorem 1 will therefore encode the scaling coefficients $(a_1, \ldots, a_\nu)$ and we will apply it to show that the final ciphertext $\boldsymbol{C}$ hides the coefficients encoded in $\boldsymbol{c}$ (note also that $\bar{\boldsymbol{x}}^T = (\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_\nu^T, \boldsymbol{y}^T)$ in Theorem 1).

Theorem 1 (Private Re-randomization of MLWE Samples). *Let $R_q := \mathbb{Z}_q[x]/(x^d + 1)$ with d a power of 2, $q = p\bar{q}$ with prime $\bar{q} = 2\ell_q + 1 \bmod 4\ell_q$, where $\ell_q \geq 2$ is a power of 2 so that $x^d + 1$ splits into ℓ_q irreducible factors $f^{(u)}(x)$ mod $\bar{q}$ for $u \in [\ell_q]$, where each $f^{(u)}(x)$ has degree d/ℓ_q. Let $L, \nu \in \mathbb{Z}$, $0 < \epsilon \leq 1/2$, $\boldsymbol{G} = I_\nu \otimes \boldsymbol{g} \in R_q^{L \times \nu}$ with $\boldsymbol{g}^T = (1, \beta, \ldots, \beta^{m_q - 1})$ for an integer $\beta \geq 2$, $\boldsymbol{c} \in R_q^L$ be arbitrary, $m_q := \lceil \log_\beta(q) \rceil$, $\nu := L/m_q \geq 1$, and $\bar{q} \geq 3\beta^2/(\beta - 1)$. For $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{L \times n})$, $\boldsymbol{E} \hookleftarrow \mathcal{D}_{R,s}^{L \times \ell'}$, and $\bar{\boldsymbol{x}} \hookleftarrow \mathcal{D}_{(\Lambda_q^{\perp}(\boldsymbol{G}) + \boldsymbol{c}) \times R^{\ell'}, r}$, let $\bar{\boldsymbol{E}} := \begin{pmatrix} \boldsymbol{E} \\ \boldsymbol{I}_{\ell'} \end{pmatrix} \in R^{(L+\ell') \times \ell'}$ and $\bar{\boldsymbol{A}} := \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{0}_{\ell' \times n} \end{pmatrix} \in R^{(L+\ell') \times n}$. Let $E_\infty := s\sqrt{\frac{2\ln(L\ell'd/\epsilon) + \ln\ln(L\ell'd/\epsilon)}{2\pi}} \geq 1$ with $\epsilon/(L\ell'd) \leq 0.001$. If*

$$q > \max\left(2(L + \ell')dcp\sqrt{1 + 4s^2 Ld/(2\pi)}, 2r(s\sqrt{Ld} + 1)\sqrt{\ln(2\ell'd/\epsilon)/\pi}\right), \quad (1)$$

$$r \geq \max\left((L + \ell')dc\sqrt{\ln(2Ld(1 + \epsilon^{-1}))/\pi}, r_G\right) \text{ and } \ell'd2^{-Ld} \leq \epsilon, \quad (2)$$

with $c := \max(c_1, c_2, c_3)$, where $c_1 \geq 2$ satisfies

$$p_1(c_1) := \sum_{r=0}^{\ell_q - 1} \frac{\binom{\ell_q}{r} \bar{q}^{((1 - r/\ell_q)n + \nu + \ell')d} \cdot (2\sqrt{d}\bar{q}^{1 - r/\ell_q}/c_1 + 1)^{Ld}}{\bar{q}^{Ld(1 - r/\ell_q)}} \leq \epsilon, \quad (3)$$

$$c_2 := 4\beta, \text{ and } Ld2^{-\ell'd} \leq \epsilon, \quad (4)$$

$$c_3 := 8\beta s\sqrt{d\ell'/(2\pi)} \text{ with } 4s^2 d\ell' \geq 2\pi, \quad (5)$$

$$r_G := \left(\sqrt{m_q}(\beta - 1) + \sqrt{\ell'd}((m_q - 1)(\beta - 1) + 1)E_\infty\right) \cdot \sqrt{\frac{\ln(2Ld(1 + \epsilon^{-1}))}{\pi}}, \quad (6)$$

then we have

$$\Delta\Big((\bar{\boldsymbol{x}}^T \bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T \bar{\boldsymbol{E}} \bmod q, \boldsymbol{A}, \boldsymbol{E}), (\mathcal{U}(R_q^n), \mathcal{D}_{\mathbb{Z}^{\ell'd}, r \cdot \mathrm{rot}(\bar{\boldsymbol{E}})}, \boldsymbol{A}, \boldsymbol{E}\Big) \leq 26\epsilon.$$

Remark 1. *We note that Theorem 1 still holds if $\boldsymbol{G}$ is replaced by any other lattice with a minimum distance greater than a constant* *fraction of q. The only reason we stated this result specific to $\boldsymbol{G}$ is that later we will use this in or* HGSW *and ZK-SNARG constructions.*

Asymptotic parameter setting. We give sample asymptotic parameter settings for Theorem 1 to show how it can be instantiated with parameters $q, r, L = \text{poly}(\kappa)$ for $\epsilon := 2^{-\kappa}$, to achieve a desired statistical distance security parameter κ. Let j^* be defined as the large integer such that $2\bar{q}^{1-j^*/\ell_q}/c_1 \leq 1$. A straightforward computation shows that the sum of the first j^* terms in condition (3) is at most $2^{-\kappa}$ if $c_1 := 2^{k/(Ld)+2}\bar{q}^{1/\alpha}$, where $\alpha := L/(n + \nu + \ell')$, and the sum of the remaining $\ell_q - j^*$ terms is at most $2^{-\kappa}$ if $L \geq n + (\nu + \ell')\ell_q + \ell_q/\log(\bar{q})(1 + (\log(d) + \kappa)/d)$. With this setting for c_1, the condition on the left hand side of (1) is satisfied if $\bar{q} > \left((2(L + \ell')d)\sqrt{1 + 9s^2Ld/(2\pi)}2^{k/Ld+2}\right)^{\alpha/(\alpha-1)}$ which leads to $c_1 \geq 2^{\kappa/Ld+2}[(2(L + \ell')d)\sqrt{1 + 9s^2Ld/(2\pi)}]^{1/\alpha}$. For example, if we choose some $d = \widetilde{\theta}(\kappa)$, $n, \ell', s = O(1)$, then it is sufficient to set some $\beta = \theta(1)$, $\ell_q = o(\log\kappa) = \widetilde{O}(1)$, $\alpha = \theta(\log\kappa) = \widetilde{O}(1)$, $L = \alpha \cdot (n + \ell' + \nu) = \widetilde{O}(1)$ to get $c = \widetilde{O}(\kappa^{\max(0.5, 1.5/(\alpha-1)}) = \widetilde{O}(\kappa^{0.5})$, $r = \widetilde{O}(\kappa^2)$, and $q = \widetilde{O}(\kappa^2)$.

3.1 Private rerandomization of MLWE - Part 1: LHL over R_q with leakage

The LHL Lemma used in [23] (Lemma 3.5) uses a general LHL with leakage result over $\mathbb{Z}_q$, which is not known to work over R_q. Instead, we aim to derive a LHL over R_q with linear leakage, using smoothing arguments.

Lemma 1 (LHL with leakage over R_q). *Let $R_q := \mathbb{Z}_q[x]/(x^d + 1)$ with d a power of 2, $q = p\bar{q}$ with prime $\bar{q} = 2\ell_q + 1 \bmod 4\ell_q$, where $\ell_q \geq 2$ is a power of 2 so that $x^d + 1$ splits into ℓ_q irreducible factors $f^{(u)}(x)$ mod $\bar{q}$ for $u \in [\ell_q]$, where each $f^{(u)}(x)$ has degree d/ℓ_q. Let $L, \nu \in \mathbb{Z}$, $0 < \epsilon \leq 1/2$, $\boldsymbol{G} = I_\nu \otimes \boldsymbol{g} \in R_q^{L\times\nu}$ with $\boldsymbol{g}^T = (1, \beta, \ldots, \beta^{m_q-1})$ for an integer $\beta \geq 2$, $m_q := \lceil\log_\beta(q)\rceil$, $\nu := L/m_q$, and $\bar{q} \geq 3\beta^2/(\beta - 1)$. For $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{L\times n})$, $\boldsymbol{E} \hookleftarrow \mathcal{D}_{R,s}^{L\times\ell'}$, and $\bar{\boldsymbol{x}} \hookleftarrow \mathcal{D}_{(\Lambda_q^\perp(\boldsymbol{G})+\boldsymbol{c})\times R^{\ell'}, r}$, let $\bar{\boldsymbol{E}} := \begin{pmatrix} \boldsymbol{E} \\ \boldsymbol{I}_{\ell'} \end{pmatrix} \in R^{(L+\ell')\times\ell'}$, $\bar{\boldsymbol{A}} := \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{0}_{\ell'\times n} \end{pmatrix} \in R^{(L+\ell')\times n}$. If*

$$q > \max\left(2(L + \ell')dcp\sqrt{1 + 4s^2Ld/(2\pi)}, 2r(s\sqrt{Ld} + 1)\sqrt{\ln(2\ell'd/\epsilon)/\pi}\right), \quad (7)$$

$$r \geq (L + \ell')dc\sqrt{\ln(2Ld(1 + \epsilon^{-1}))/\pi} \textit{ and } \ell'd2^{-Ld} \leq \epsilon, \quad (8)$$

with $c := \max(c_1, c_2, c_3)$, where $c_1 \geq 2$ satisfies

$$p_1(c_1) := \sum_{r=0}^{\ell_q-1} \frac{\binom{\ell_q}{r}\bar{q}^{((1-r/\ell_q)n+\nu+\ell')d} \cdot (\sqrt{d}\bar{q}^{1-r/\ell_q}/c_1 + 1)^{Ld}}{\bar{q}^{Ld(1-r/\ell_q)}} \leq \epsilon, \quad (9)$$

$$c_2 := 4\beta, \textit{ and } Ld2^{-\ell'd} \leq \epsilon, \quad (10)$$

$$c_3 := 8\beta s\sqrt{d\ell'/(2\pi)} \textit{ with } 4s^2d\ell' \geq 2\pi, \quad (11)$$

then we have

$$\Delta\Big((\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}} \bmod q, \boldsymbol{A}, \boldsymbol{E}), (\mathcal{U}(R_q^n), \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}} \bmod q, \boldsymbol{A}, \boldsymbol{E})\Big) \leq 21\epsilon.$$

We have already outlined above the main steps of the proof of Lemma 1 based on lattice smoothing arguments, and how it reduces the problem to lower bounding the $\ell'd + 1$'th minimum of the sum lattice $\Lambda^* := \Lambda_q(\bar{A}) + \Lambda_q(\bar{E}) + \Lambda_q(\bar{G})$ of the three underlying dual lattices. Here, due to space limits, we summarize the results for the subcases studied for vectors $\boldsymbol{w} = \bar{\boldsymbol{A}}\boldsymbol{v}_A + \bar{\boldsymbol{E}}\boldsymbol{v}_E + \bar{\boldsymbol{G}}\boldsymbol{v}_G$ in Λ^* to obtain the lower bound, and refer to the Appendix for the complete proofs.

Subcase 1. For *Subcase 1* ($\boldsymbol{v}_A \neq 0 \bmod \bar{q}$) we use a probabilistic approach to lower bound $\|\boldsymbol{w}\|$ over the randomness of $\boldsymbol{A}$ and using a union bound over $\boldsymbol{v}_E, \boldsymbol{v}_G$ by extending the approach from [59, 65] for lower bounding the minimum of Module SIS lattices, and obtain the following result.

Lemma 2. *Let $R_{\bar{q}} := \mathbb{Z}_{\bar{q}}[x]/(x^d + 1)$ with d a power of 2, $\bar{q} = 2\ell_q + 1 \bmod 4\ell_q$, where $\ell_q \geq 2$ is a power of 2 so that $x^d + 1$ splits into ℓ_q irreducible factors $f^{(u)}(x)$ mod $\bar{q}$ for $u \in [\ell_q]$, where each $f^{(u)}(x)$ has degree d/ℓ_q. Let $\boldsymbol{w} := \boldsymbol{A}\boldsymbol{v}_A + \boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G \in R_{\bar{q}}^L$ with $(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G) \in R_{\bar{q}}^n \setminus \boldsymbol{0} \times R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^\nu$ and $c_1 \geq 2$. Then*

$$\|\boldsymbol{w}\|_2 \geq \bar{q}/c_1 \quad (12)$$

except with probability p_1 over the choice of $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_{\bar{q}}^{L\times n})$, where

$$p_1 \leq \sum_{r=0}^{\ell_q-1} \frac{\binom{\ell_q}{r}\bar{q}^{((1-r/\ell_q)n+\nu+\ell')d} \cdot (2\sqrt{d}\bar{q}^{1-r/\ell_q}/c_1 + 1)^{Ld}}{\bar{q}^{Ld(1-r/\ell_q)}}. \quad (13)$$

Subcase 2. For subcase 2, with $\boldsymbol{w} \in \Lambda_{\bar{q}}(\boldsymbol{M})$ but not in the column span of $\boldsymbol{E}$ over $\mathbb{Z}$ where $\boldsymbol{v}_A = 0 \bmod \bar{q}$ and $\boldsymbol{v}_E \neq 0 \bmod \bar{q}$ with 'short' $vec\boldsymbol{v}_E$, write $\boldsymbol{w} = \boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G \bmod \bar{q}$. We first prove the following lemma, which upper bounds the minimum distance of a lattice generated by Gadget matrix $\boldsymbol{G}$. This result is stated as general as possible as it might be of an independent interest in other cryptography contexts.

Lemma 3 (Minimum Distance of Gadget Matrix $\boldsymbol{G}$). *Let $\boldsymbol{G} = I_\nu \otimes \boldsymbol{g} \in R_{\bar{q}}^{L\times\nu}$ with $\boldsymbol{g}^T = (1, \beta, \ldots, \beta^{m_q-1})$ for an integer $\beta \geq 2$, $m_q \geq \lceil\log_\beta(\bar{q})\rceil$ and $\nu := L/m_q$, with $\gcd(\beta, \bar{q}) = 1$ and $\bar{q} \geq 3\beta^2/(\beta - 1)$. Then we have*

$$\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) \geq \frac{\bar{q}}{2\beta}. \quad (14)$$

Using the above lower bound on the norm of $\boldsymbol{G}\boldsymbol{v}_G$ and a Schwarz inequality upper bound on the norm of $\boldsymbol{E}\boldsymbol{v}_E$ for 'short' $\boldsymbol{v}_E$, we obtain the following main result of Subcase 2.

Lemma 4. *Let $L, d, \bar{q}, \ell', \nu \geq 2$ be integers. Let also $c_2 := 4\beta$ for an integer $\beta \geq 2$, $c_3 := 8\beta s\sqrt{d\ell'/(2\pi)}$ with $4s^2d\ell' \geq 2\pi$, $\gcd(\beta, \bar{q}) = 1$ and $\bar{q} \geq 3\beta^2/(\beta - 1)$. Let $\bar{\boldsymbol{M}} := (\bar{\boldsymbol{E}}, \bar{\boldsymbol{G}}) \in R_{\bar{q}}^{(L+\ell')\times(\ell'+\nu)}$ with $(\bar{\boldsymbol{E}}, \bar{\boldsymbol{G}})$ as defined above, so that every $\boldsymbol{w} \in \Lambda_{\bar{q}}(\bar{\boldsymbol{M}})$ can be written as $\boldsymbol{w} = \bar{\boldsymbol{E}}\boldsymbol{v}_E + \bar{\boldsymbol{G}}\boldsymbol{v}_G$ with $(\boldsymbol{v}_E, \boldsymbol{v}_G) \in R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^\nu$. Then*

$$\Pr_{\boldsymbol{E}\hookleftarrow\mathcal{D}_{R,s}^{L\times\ell'}}\Big[\exists(\boldsymbol{v}_E, \boldsymbol{v}_G), \|\boldsymbol{v}_E\| \leq \bar{q}/c_3 : \boldsymbol{w} \in \Lambda_{\bar{q}}(\bar{\boldsymbol{M}}) \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell'd},$$

$$\|\boldsymbol{w}\| < \bar{q}/c_2\Big] \leq Ld2^{-d\ell'}. \quad (15)$$

3.2 Private Rerandomization of MLWE - Part 2: Gaussian LHL

We now state our adaptation of the Gaussian Leftover Hash Lemma from [23] to our module case with Gaussian $\boldsymbol{E}$.

Lemma 5 (Gaussian LHL over modules with Gaussian $\boldsymbol{E}$). *Let $0 \leq \epsilon < 1/2$, $\boldsymbol{G} = I_\nu \otimes \boldsymbol{g} \in R_q^{L\times\nu}$ with $\boldsymbol{g}^T = (1, \beta, \ldots, \beta^{m_q-1})$ and $\nu := L/m_q$, $\boldsymbol{E} \hookleftarrow \mathcal{D}_{R,s}^{L\times\ell'}$, and $\bar{\boldsymbol{E}}$ as above, let $E_\infty :=$*

$s\sqrt{\frac{2\ln(L\ell' d/\epsilon)+\ln\ln(L\ell' d/\epsilon)}{2\pi}} \geq 1$ *with* $\epsilon/(L'\ell d) \leq 0.001$*. Then, if* $r \geq \left(\sqrt{m_q}(\beta-1)+\sqrt{\ell' d}((m_q-1)(\beta-1)+1)E_\infty\right)\cdot\sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}}$*, we have*

$$\Delta\left((\boldsymbol{x}^T\bar{\boldsymbol{E}},\bar{\boldsymbol{E}}),(\mathcal{D}_{\mathbb{Z}^{\ell' d},r\cdot\mathrm{rot}(\bar{\boldsymbol{E}})},\bar{\boldsymbol{E}})\right)\leq 5\epsilon.$$

4 HALF-GSW CANDIDATE LO ENCRYPTION

4.1 Module Half-GSW (HGSW)

Since we only need the good private scaling property, we can modify the Full-GSW construction such that we keep the scaling property and remove the multiplicative homomorphism property, which is not needed for the ZK-SNARG construction. We introduce the 'Half-GSW' scheme, where we keep only the bottom m_q rows of the Full-GSW ciphertext, which can also be viewed as a collection of m_q Regev ciphertexts for the plaintexts $2^0\mu,\dots,2^{m_q-1}\mu$. For our ZK-SNARG construction, it is necessary that the only way for an adversary to create a valid ciphertext is to take linear combinations of given valid ciphertexts. Therefore, as in [47], we include a sparsification parameter τ and encrypt the extended message $\bar{\boldsymbol{\mu}}=[\boldsymbol{\mu}^T|(\boldsymbol{T}\boldsymbol{\mu})^T]\in R_p^{\ell'}$ (instead of $\boldsymbol{\mu}$), where $\boldsymbol{T}$ is a random matrix. The decryption checks that the recovered message has this form to circumvent oblivious sampling of a valid ciphertext.

Module Half-GSW Construction. We now define our Half-GSW encryption scheme HGSW. For this scheme, we use a ciphertext modulus $q=p\bar{q}$ for some prime plaintext modulus p, and we work over rings $R_q:=\mathbb{Z}_q[x]/(x^d+1)$ and $R_p:=\mathbb{Z}_p[x]/(x^d+1)$ with d a power of 2 as in the previous section. The scheme consists of three algorithms HGSW.Setup, HGSW.Encrypt, HGSW.Decrypt. Let $\boldsymbol{g}^T=(1,\beta^1,\dots,\beta^{m_q-1})\in R_q^{m_q}$ and $\ell'=\ell+\tau$, where τ is the sparsification parameter. HGSW is a stateful deterministic encryption scheme. It takes the message index i (a counter) as input and all randomness is in secret key and generated in Setup. It encrypts $\leq m$ vector messages $\mu_i\in R_p^\ell$, $i\in[m]$.

- HGSW.Setup$(1^\lambda,1^\ell)$: On input a security parameter λ, samples $\boldsymbol{S}\leftarrow\mathcal{D}_{R,s}^{n\times\ell'}$, the matrices $\boldsymbol{A}\hookleftarrow\mathcal{U}(R_q^{mm_q\times n})$ and $\boldsymbol{E}\hookleftarrow\mathcal{D}_{R,s}^{mm_q\times\ell'}$ and the transformation matrix $\boldsymbol{T}\hookleftarrow\mathcal{U}(R_p^{\tau\times\ell})$. For $i\in[m]$, we denote by $\boldsymbol{E}_i\in R^{m_q\times\ell'}$ and $\boldsymbol{A}_i\in R_q^{m_q\times n}$ the ith blocks of consecutive m_q rows from $\boldsymbol{E}$ and $\boldsymbol{A}$, respectively. The secret key is sk $=(\boldsymbol{S},\boldsymbol{T},\boldsymbol{A},\boldsymbol{E})$.
- HGSW.Encrypt$(i,\mathrm{sk},\boldsymbol{\mu})$: Given the message index i, secret key sk $=(\boldsymbol{S},\boldsymbol{T},\boldsymbol{A},\boldsymbol{E})$ and a message vector $\boldsymbol{\mu}_i^T=(\mu_{i,1},\dots,\mu_{i,\ell})\in R_p^\ell$, computes the vector $\bar{\boldsymbol{\mu}}_i^T=[\boldsymbol{\mu}_i^T|(\boldsymbol{T}\boldsymbol{\mu}_i)^T]\in R_p^{\ell'}$. Parse $\bar{\boldsymbol{\mu}}_i^T=(\bar{\mu}_{i,1},\dots,\bar{\mu}_{i,\ell'})$. The algorithm then computes the ciphertext
$$\boldsymbol{C}_i=\begin{bmatrix}\boldsymbol{A}_i & \boldsymbol{A}_i\boldsymbol{S}+\boldsymbol{E}_i\end{bmatrix}+\frac{q}{p}\cdot\boldsymbol{H}_i\in R_q^{m_q\times(n+\ell')},\text{ where}$$
$$\boldsymbol{H}_i:=\begin{bmatrix}\boldsymbol{0}^{m_q\times n}, & \bar{\mu}_{i,1}\boldsymbol{g}, & \dots, & \bar{\mu}_{i,\ell'}\boldsymbol{g}\end{bmatrix}\in R_q^{m_q\times(n+\ell')}.$$
- HGSW.Add$(\{\boldsymbol{C}_i\}_{i\in[m]},\{a_i\}_{i\in[m]})$: Let L denote the add block size parameter, where $v:=L/m_q$ is a positive integer (number of ciphertexts per block). For each add block index $j\in[m/v]$, and each ciphertext index $i\in[v]$ in the j'th block, given the scaling factor $a_{jv+i}\in R$ for the ciphertext $\boldsymbol{C}_{jv+i}$, sample a randomized scaling factor $\tilde{a}_{jv+i}^T:=\boldsymbol{g}_{\mathrm{rand}}^{-1}(a_{jv+i})$ (see Def. 3) and compute the re-randomized scaled ciphertext
$$\tilde{\boldsymbol{C}}_{jv+i}:=\tilde{a}_{jv+i}^T\cdot\boldsymbol{C}_{jv+i}\in R_q^{1\times(n+\ell')}$$
(**Remark:** note that $\tilde{\boldsymbol{C}}_{jv+i}$ is equal to
$$\begin{bmatrix}\boldsymbol{b}_i^T\boldsymbol{S}+\boldsymbol{e}_i^T\end{bmatrix}+\frac{q}{p}\cdot\begin{bmatrix}\boldsymbol{0}^n,\bar{\mu}_{i,1}a_i,\dots,\bar{\mu}_{i,\ell'}a_i\end{bmatrix}\in R_q^{1\times(n+\ell')},$$
where $\boldsymbol{b}_i^T=\tilde{a}_{jv+i}^T\cdot\boldsymbol{A}_{jv+i}$, and $\boldsymbol{e}_i^T=\tilde{a}_{jv+i}^T\cdot\boldsymbol{E}_{jv+i}$). Finally, for $j=0,\dots,m/v-1$, sample $\boldsymbol{y}_j$ from discrete Gaussian $\mathcal{D}_r^{\ell'}$, and compute the sum
$$\boldsymbol{c}^*:=\sum_{j=0}^{m/v-1}\left(\sum_{i=1}^{v}\tilde{\boldsymbol{C}}_{jv+i}+[\boldsymbol{0}^n,\boldsymbol{y}_j^T]\right)\in R_q^{1\times(n+\ell')}.$$
(**Remark:** the output of HGSW.Add does not match the format of a ciphertext created by HGSW.Encrypt. Instead, it is a one-row Regev-type ciphertext).
- HGSW.Decrypt$(\boldsymbol{S},\boldsymbol{c}^*)$: given a ciphertext $\boldsymbol{c}^*$ and the secret key sk $=(\boldsymbol{S},\boldsymbol{T},\boldsymbol{A},\boldsymbol{E})$, computes the inner product of $\boldsymbol{c}^*$ with $\bar{\boldsymbol{S}}^T=\left[-\boldsymbol{S}^T,\boldsymbol{I}_{\ell'}^T\right]$, i.e. $\bar{\boldsymbol{H}}:=\langle\boldsymbol{c}^*,\bar{\boldsymbol{S}}\rangle$ and computes $\bar{\boldsymbol{\mu}}:=\lceil(p/q')\cdot\bar{\boldsymbol{H}}\rfloor\in R_p^{\ell'}$. Parse $\bar{\boldsymbol{\mu}}=[\bar{\boldsymbol{\mu}}_1,\bar{\boldsymbol{\mu}}_2]$, where $\bar{\boldsymbol{\mu}}_1=\boldsymbol{\mu}\in R_p^\ell$ and $\bar{\boldsymbol{\mu}}_2\in R_p^\tau$. If $\bar{\boldsymbol{\mu}}_2\neq\boldsymbol{T}\bar{\boldsymbol{\mu}}_1$ then return $\perp$, else return $\bar{\boldsymbol{\mu}}_1\in R_p^\ell$. (**Remark:** the given decryption algorithm applies only to the output of HGSW.Add, which was mentioned to be a one-row Regev-type ciphertext).

The main advantage of our Module HGSW is that it keeps the property of homomorphic scaling and drops the multiplicative homomorphism property. As a result, this scheme can be used in our ZK-SNARG from LPCP construction.

As noted in Appendix A, knowledge-based notions of the linear-only property for LWE-based encryption schemes have recently been shown insecure against quantum attacks. Instead, we introduce the following weaker (and not knowledge-based) linear only hardness assumption on HGSW, which will still be sufficient for our construction of a ZK-SNARG (i.e. argument with regular, rather than knowledge-based, soundness).

Conjecture 1 (HGSW Linear Targeted Malleability)**.** *For security parameter λ and the parameters p,d,τ as defined in the construction of* HGSW*, if $1/|R_p|^\tau=p^{-d\tau}$ is negligible in λ, then* HGSW *satisfies strictly linear targeted malleability (see Def. 13 in Appendix).*

Remark 2. *For a ZK-SNARG from linear-only FHE construction, we only require the properties of homomorphic scaling and homomorphic addition, as in the inner-product homomorphism in ZK-SNARG [47]. Therefore it is sufficient if we compute only one row (one Regev ciphertext) of the m_q rows of $\boldsymbol{g}_{\mathrm{rand}}^{-1}(a\boldsymbol{g})$, and apply the homomorphic additions on that one. It means that the ZK-SNARG proof will only consist of short Regev ciphertexts. The longer* HGSW *ciphertexts will only be needed in the generation of the common reference string.*

Lack of Multiplicative Homomorphism. As we have seen earlier, the multiplicative homomorphism is undesirable for the soundness of a secure ZK-SNARG construction. For our HGSW scheme, the GSW multiplicative homomorphism idea seems to fail due to the missing half of the GSW ciphertext. For example, suppose $\ell'=1$ and consider two HGSW ciphertexts $\boldsymbol{C}_1=[\boldsymbol{C}_{1,1},\boldsymbol{C}_{1,2}]=[\boldsymbol{A}_1,\boldsymbol{A}_1\boldsymbol{S}+\boldsymbol{E}_1+\mu_1\boldsymbol{g}]$ and $\boldsymbol{C}_2=[\boldsymbol{C}_{2,1},\boldsymbol{C}_{2,2}]=[\boldsymbol{A}_2,\boldsymbol{A}_2\boldsymbol{S}+\boldsymbol{E}_2+\mu_2\boldsymbol{g}]$. Then to compute a ciphertext $\boldsymbol{C}_3$ for the product $\mu_1\mu_2$, we could try to compute $\boldsymbol{C}_3=$

$\boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{C}_{2,2}) \cdot \boldsymbol{C}_1 = [\boldsymbol{C}_{3,1}, \boldsymbol{C}_{3,2}] = [\boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{C}_{2,2})\boldsymbol{A}_1, \boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{C}_{2,2})\boldsymbol{A}_1\boldsymbol{S} + \boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{C}_{2,2})\boldsymbol{E}_1 + \mu_1\boldsymbol{C}_{2,2}] = [\boldsymbol{C}_{3,1}, \boldsymbol{C}_{3,1}\boldsymbol{S} + \boldsymbol{g}_{\mathrm{rand}}^{-1}(\boldsymbol{C}_{2,2})\boldsymbol{E}_1 + \mu_1\boldsymbol{E}_2 + \mu_1\mu_2\boldsymbol{g} + \mu_1\boldsymbol{A}_2\boldsymbol{S}]$. Note that $\boldsymbol{C}_3$ is *not* a valid ciphertext for $\mu_1\mu_2$ due to the last 'large' term $\mu_1\boldsymbol{A}_2\boldsymbol{S}$ in $\boldsymbol{C}_{3,2}$. It seems that without the missing 'top half' of the GSW ciphertext, we cannot get rid of such extra terms to get a multiplicative homomorphism. This is the motivation for our linear targeted malleability conjecture for HGSW.

Modulus switching. To achieve noise reduction after applying the required number of homomorphic additions, we use the modulus switching technique introduced in [24]. While the initial modulus q is needed to be large enough to allow the homomorphic operations, it can be reduced to q' by directly applying modulus switching from [24] to our Regev ciphertext computed in HGSW.Add.

4.2 Correctness and Security Analysis

Theorem 2 (Additive Homomorphism (Correctness)). *Let λ be a security parameter and $p, q, n, m_q, \beta, \ell', \ell, \tau$ be as defined in Module HGSW Construction. Suppose $\mathcal{D}_s$ be a Gaussian with parameter s. If $p, q, n, m_q, \beta, \ell', \ell, \tau = \mathsf{poly}(\lambda)$, then this Construction is additively homomorphic with respect to $S = \mathcal{D}_r^m \subseteq R_p^m$ for all $m = m(\lambda)$. In particular, if $n > 8$ and*

$$q > 2ps\sqrt{(r^2(mm_q + m/v)d + 1)\ln(2((mm_q + m/v)d + 1)/\epsilon)/\pi}, \quad (16)$$

then (19) *in Def. 12 (App. B.6) holds with probability ϵ for all $\boldsymbol{y} \in S$.*

Circuit Privacy Remark: the j'th non-zero component in R_q of $a\frac{q}{p} \cdot \boldsymbol{H}$ has the form $a\mu_j\frac{q}{p} \bmod q = (a\mu_j \bmod p)\frac{q}{p} \bmod q$, where we have used the fact that p divides q. Here, it seems that we *cannot* obtain circuit privacy if p does not divide q and we replace in the encryption scheme $\frac{q}{p}$ by its rounded version $\lceil\frac{q}{p}\rfloor = \frac{q}{p} + \epsilon$ for $|\epsilon| \leq 1/2$. With this, we get $a\mu_j\lceil\frac{q}{p}\rfloor \bmod q = (a\mu_j \bmod p)\frac{q}{p} + a\mu_j\epsilon \bmod q$, and the term $a\mu_j\epsilon$ depends on $a\mu_j \bmod q$, not just $a\mu_j \bmod p$, and therefore can leak more about a.

Theorem 3 (Statistical Circuit Privacy of HGSW). *Let $\epsilon > 0$ and $p, q, n, m_q, \beta, \ell', v, L, s, r$ be as defined in the Module HGSW Construction. If these parameters satisfy the conditions of Theorem 1 , with $\epsilon = \mathsf{negl}(\lambda)$ then the* HGSW *construction is statistically circuit private. In particular, for every circuit privacy adversary $\mathcal{A}$, there exists an efficient simulator $\mathcal{S}$ such that*

$$\Pr[\mathsf{Game}^{\mathsf{circ-priv}}_{\Pi_{\mathsf{HGSW.Encrypt}},\mathcal{A},\mathcal{S}}(1^\lambda) = 1] \leq 1/2 + 18(m/v) \cdot \epsilon.$$

The IND-CPA security of our Module HGSW follows directly from [44] and Module-LWE assumption.

Theorem 4 (CPA Security of Module HGSW). *For a security parameter λ let $p = p(\lambda), q = q(\lambda), n = n(\lambda), \mathcal{D} = \mathcal{D}(\lambda)$ be the lattice parameters and ℓ be the plaintext dimension. Let $Q = \mathsf{poly}(\lambda)$ denote the number of queries to the encryption oracle. Under the hardness of $\mathsf{MLWE}_{n,m_q,d,q,\mathcal{D}}$ assumption with $m_q = n + Q$, the* HGSW *construction is Q-query IND-CPA secure.*

Asymptotic parameter settings. Based on the hardness of MLWE against known lattice attacks (attack time $T = 2^{\widetilde{\Omega}(nd\log q/\log^2(q/s))}$, which we require to be $\geq 2^\lambda$), we can satisfy conditions of the above Theorems with HGSW ciphertext length quasi-linear in the security/privacy parameter λ and poly-log in the number of homomorphic plaintext additions m. For example, similarly to Sec. 3, if choose some $d = \widetilde{\theta}(\lambda), n, \ell', p, s, v, \tau = O(1)$, then it is sufficient to set some $\beta = \theta(1), \ell_q, \ell_p = O(1), \alpha = \theta(\log(\lambda m)) = \widetilde{O}(1), L = \alpha\cdot(n+\ell'+v) = \widetilde{O}(1)$, $r = \widetilde{O}(\lambda^2)$, and $q = \widetilde{O}(\lambda^3\sqrt{m})$. The HGSW ciphertext length before homomorphic addition is $m_q d(n + \ell')\log q = O(\log^2 m) \cdot \widetilde{O}(\lambda)$, whereas after homomorphic addition the Regev ciphertext length is $d(n + \ell')\log q = O(\log m) \cdot \widetilde{O}(\lambda)$.

5 APPLICATION OF HGSW TO ZK-SNARG

This section is dedicated to an application of HGSW to ZK-SNARG. Our construction of LUNA follows directly by applying the cryptographic compiler from [16]. Similar to [47], we provide a construction based on linear PCPs for R1CS systems. The main difference to [47] is the underlying linear-only encryption scheme. In our case, we use HGSW defined in Section 4.1.

We propose to use the following encoding scheme of field elements into the CRT slots of R_p. According to the Chinese Remainder Theorem it holds that $R_p = \mathbb{Z}_p[x]/(x^d + 1)$ is isomorphic (denoted by $\cong$) to $\prod_{i\in[\ell_p]}\mathbb{Z}_p[x]/(f_i(x))$ for irreducible polynomials $f_i(x)$ which are factors of $x^d + 1 \mod p$. As shown in [25], there is an isomorphism $R_p \cong \prod_{i\in[\ell_p]}\mathbb{Z}_p[x]/(f_i(x))$ with $\deg(f_i) = f$ for all $i \in [\ell_p]$. Furthermore, $\mathbb{Z}_p[x]/(f_i(x)) \cong \mathbb{F}(p^f)$ where $f = d/\ell_p$. While compiling LPCP into a ZK-SNARG, we encode ℓ_p LPCP plaintexts $\mu_1, \ldots, \mu_{\ell_p} \in \mathbb{F}(p^f)$ of the HGSW encryption scheme into the ℓ plaintext slots in $\mathbb{Z}_p[x]/(f_i(x))$.

5.1 LUNA: Our ZK-SNARG Construction

For a family of R1CS systems $\mathcal{CS} = \{\mathcal{CS}_N\}_{N\in\mathbb{N}}$ defined over a finite field $\mathbb{F}$, the ZK-SNARG construction consists of two building blocks: a linear PCP and an additively-homomorphic vector encryption for $\mathbb{F}^k$.

- Let $\Pi_{\mathsf{LPCP}} = (\Pi_{\mathsf{LPCP}}.\mathsf{Query}, \Pi_{\mathsf{LPCP}}.\mathsf{Prove}, \Pi_{\mathsf{LPCP}}.\mathsf{Verify})$ be a k-query linear PCP for $\mathcal{CS}$. Let m denote the query length of Π_{LPCP}.
- Let HGSW = (HGSW.Setup, HGSW.Encrypt, HGSW.Add, HGSW. Decrypt) be our additively-homomorphic half-GSW symmetric encryption over $\mathbb{F}^k$.

Our designated-verifier LUNA = (LUNA.Setup, LUNA.Prove, LUNA.Verify) is defined as follows:

- LUNA.Setup$(1^\lambda, 1^N)$: On input the security parameter λ and the system index N, run $(\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Setup}(1^N)$, where $\boldsymbol{Q} \in \mathbb{F}^{m\times k}$ with $\mathbb{F} = \mathbb{F}(p^f)$ and f is the degree of splitting factors of $(x^d + 1) \mod p$. For $i \in [m]$, let $\boldsymbol{q}_i^T$ denote the i-th row of $\boldsymbol{Q}$. Run $\mathsf{sk} \leftarrow \mathsf{HGSW.Setup}(1^\lambda)$ and compute $\boldsymbol{C}_i = \mathsf{HGSW.Encrypt}(i, \mathsf{sk}, \boldsymbol{q}_i)$ for each $i \in [m]$. Output $\mathsf{crs} = (N, \{\boldsymbol{C}_i\}_{i\in[m]})$ and the verification key $\mathsf{st} = (\mathsf{st}_{\mathsf{LPCP}}, \boldsymbol{S})$.
- LUNA.Prove$(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$: On input common reference string $\mathsf{crs} = (N, \{\boldsymbol{C}_i\}_{i\in[m]})$, a statement $\boldsymbol{x}$ and a witness $\boldsymbol{w}$, compute a proof of the underlying LPCP system $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$, i.e. $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_m)$. Then homomorphically compute the response of linear PCP as $\boldsymbol{c}^* \leftarrow \mathsf{HGSW.Add}(\{\boldsymbol{C}_i\}_{i\in[m]}, \{\pi_i\}_{i\in[m]})$. The prover outputs the proof $\boldsymbol{\pi}^* = \boldsymbol{c}^*$.
- LUNA.Verify$(\mathsf{st}, \boldsymbol{\pi}^*, \boldsymbol{x})$: On input $\mathsf{st} = (\mathsf{st}_{\mathsf{LPCP}}, \mathsf{sk})$, the statement $\boldsymbol{x}$ and the proof $\boldsymbol{\pi}^* = \boldsymbol{c}^*$, the verifier computes $\boldsymbol{a} = \sum_{i=1}^m \pi_i\boldsymbol{q}_i^T \leftarrow \mathsf{HGSW.Decrypt}(\boldsymbol{S}, \boldsymbol{c}^*)$. If $\boldsymbol{a} = \perp$, the verifier outputs 0, otherwise

it runs the verification of LPCP, $\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}_{\mathsf{LPCP}}, \boldsymbol{x}, \boldsymbol{a})$ and outputs its output.

Security. As discussed in App. A, we avoid relying on knowledge-based flavours of LO property for HGSW due to the invalidity of such assumptions for LWE-based schemes as recently shown in [31]. Instead we use the ZK-SNARG soundness result of [16, 17], that only requires a weaker simulation-based LO requirement on the underlying encryption scheme called (strict) Linear Targeted Malleability (LTM), see Def. 13 in Appendix B.6. Informally, the latter requirement only requires that is infeasible for an efficient adversary to compute non-linear homomorphic operations on ciphertexts, but it does not require a knowledge extractor for the linear coefficients. Note that the LTM-based ZK-SNARG soundness result is against non-adaptive attacks, where the statement is not chosen adaptively by the adversary after seeing the CRS. However, as pointed out in [16], this may only be a proof limitation, as there is no known adaptive soundness attack on the ZK-SNARG.

Theorem 5 (ZK-SNARG Security - Adapted from [16], [17, Thm 4.6], [47, Thm 3.23])**.** *If Π_{LPCP} satisfies statistical* soundness *against linear provers and the underlying vector encryption scheme* HGSW *is IND-CPA secure (for up to m messages) and satisfies* strictly linear-only targeted malleability*, then* LUNA *is a (non-adaptive) designated-verifier SNARG for $\mathcal{CS}$ in the preprocessing model. Moreover, if in addition Π_{LPCP} is honest-verifier zero-knowledge and the underlying encryption scheme* HGSW *is statistically circuit private, then* LUNA *is a statistically zero-knowledge DV ZK-SNARG.*

As discussed in Appendix A, our construction is also a ZK-SNARK, i.e. a zero-knowledge succinct non-interactive argument of *knowledge*, if we assume a stronger strict LTM assumption with *computationally efficient simulation* on the HGSW encryption scheme (as opposed to the strict LTM assumption with computationally unbounded simulation in Theorem 5), and that the underlying LPCP satisfies knowledge soundness. As for the ZK-SNARG result, the soundness is only proved against non-adaptive attacks.

Theorem 6 (ZK-SNARK Security - Adapted from [16, Lem. 6.3], [47, Thm 3.23])**.** *If Π_{LPCP} satisfies* statistical knowledge soundness *against linear provers and the underlying vector encryption scheme* HGSW *is IND-CPA secure (for up to m messages) and satisfies* strictly linear-only targeted malleability with computationally efficient simulation*, then* LUNA *is a (non-adaptive) designated-verifier SNARK for $\mathcal{CS}$ in the preprocessing model. Moreover, if in addition Π_{LPCP} is honest-verifier zero-knowledge and the underlying encryption scheme* HGSW *is statistically circuit private, then* LUNA *is a statistically zero-knowledge DV ZK-SNARK.*

5.2 Parameter Setting for LUNA

In our work, we adopt a set of notations similar to those in [47] to make it easy to connect the two works together. The notations are summarized in Table 4 in the appendices. For the parameters and requirements common in both [47] and our work, we employ a similar strategy to choose such parameters. For example, as in [47], we assume that the number of variables, N_w is roughly equal to the number of constraints, N_g, i.e., $N_w \approx N_g$. Particularly, we take $N_g = 2^{16}$ or $N_g = 2^{20}$, which are used as common example settings in prior works, including [47]. Of course, for certain parameters, we have different requirements and optimizations, in which case we rely on our new results.

Plaintext dimension. First of all, for all parameter settings, the plaintext dimension over R_p is set to $\ell = \lceil 4\rho/\ell_p \rceil$. As in [47], there are 4ρ PCP queries in total to be encrypted since each linear PCP has 4 queries and we repeat ρ times to amplify soundness. While the plaintext space in [47] is a field and, therefore, does not split (i.e., $\ell_p = 1$ in [47]), we can pack ℓ_p messages into a single R_p element. As a result, we get $\ell = \lceil 4\rho/\ell_p \rceil$. Note that this setting is the same as in [47] with $\ell_p = 1$.

Ciphertext sparsification. For all parameter settings, the sparsification parameter is set to $\tau = \lceil 128/(d \log p) \rceil$. The reason behind this choice is based on the LTM conjecture (similar to [47]) adapted to our base encryption scheme (see Conjecture 1). Observe that for $\tau = \lceil 128/(d \log p) \rceil$, we have $p^{-\tau d} \leq 2^{-128}$ as in [47]. The difference in our case is that we work over a *ring* R_q instead of a field. However, the rationale described in [47] extends to the ring case as follows. For any fixed vector $(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ recovered in decryption, the probability that $\boldsymbol{\mu}_2 = \boldsymbol{T}\boldsymbol{\mu}_1$ over R_p is equal to the probability that $\boldsymbol{\mu}_2 = \boldsymbol{T}\boldsymbol{\mu}_1$ over all the fields $R_p^{(i)}$ that R_p splits into. Since over each $R_p^{(i)}$, the probability is $p^{-\tau d'}$ for $d' = \dim(R_p^{(i)})$, overall we end up with the same requirement $p^{-\tau d} \leq 2^{-128}$.

Modulus switching. Theorem 3.19 in [47] provides a general modulus switching result for Regev-like encryption schemes. Since our final ciphertext after homomorphic scaling has a similar Regev structure, we can apply the results of [47, Theorem 3.19]. Particularly, since we have the same notations for p, n, d, s, q' as in [47], we can simplify the modulus switching requirement to

$$q' > 12pnds, \tag{17}$$

where we use the same constant $C = 6$ as in [47] for Gaussian "tail-cut" bound.

Observation 1. *Note that from the MLWE security perspective the product nd in* (17) *is roughly fixed, and therefore, an approach to reduce the proof length is by reducing the Gaussian parameter s and/or the plaintext modulus p. This stems from the fact that the proof length is equal to $(n + \ell')d \log q' = (n + \ell + \tau)d \log q'$. We will exploit this observation when choosing s and p.*

PCP Knowledge Error. The knowledge (and soundness) error of the PCP is tightly related to the size of the finite field $\mathbb{F}$ over which PCP is instantiated. Particularly, as in [47], the knowledge error ε is at most $2N_g/(|\mathbb{F}|-N_g)$, assuming the number of variables $N_w \approx N_g$, where N_g is the number of constraints in the system. In our construction, we have $\varepsilon \leq 2N_g/(p^f - N_g)$ since $|\mathbb{F}| = p^f$ where $f = d/\ell_p$ is the degree of the irreducible factors of $x^d + 1 \bmod p$. Observe that the ability to choose a larger f allows us to reduce the size of p significantly, which in turn allows the reduction of q' due to Observation 1. For all parameter settings, we set the number of repetitions as the smallest integer ρ such that $\varepsilon^\rho \leq 2^{-128}$.

Correctness for m homomorphic additions. According to the correctness requirement of our ZK-SNARG, Theorem 2, we choose q large enough to ensure that (16) is satisfied. Here, we set $\epsilon = 2^{-128}$ and also have $m \approx 2N_g$ as Π_{SNARG}.Prove involves about $2N_g$ homomorphic additions assuming $N_w \approx N_g$.

CPA/MLWE security. Due to Theorem 4, the required CPA security of the base encryption scheme (HGSW) relies on MLWE assumption with a secret key in R_q^n (i.e., total dimension of nd) and secret/error distribution of discrete Gaussian with parameter s. To establish a fair comparison with [47], we calculated the "root Hermite factor" δ_{LWE} of the parameter settings in [47] and found it to be $\delta_{\mathsf{LWE}} \approx 1.00427$. The root Hermite factor is a common metric to measure the hardness of solving lattice problems in practice. Therefore, we also aim for a similar root Hermite factor when setting the lattice parameters and use the LWE estimator [7] to compute δ_{LWE}. For lattice attacks, the number of MLWE samples does not play a major role and we assume that the attacker has access to (at least) the optimal number of samples.

In choosing s, we need to consider algebraic attacks[4] and the number of MLWE samples revealed to the adversary, which can in fact be quite large for the PCP-based SNARG approach we employ. However, even for $s = 1$, we observed from the LWE estimator that the estimated complexity (time to success probability ratio) of algebraic attacks are well above 2^{128} operations for the dimension parameters we consider. To be conservative and avoid having a very sparse secret, we set $s = 2$ to optimize the proof length in light of Observation 1. Overall, in our choice of parameters, we ensure that the parameters $(n, d, \log q)$ with $s = 2$ lead to a root Hermite factor of $\delta_{\mathsf{LWE}} \approx 1.00427$.

Zero-knowledge. The ZK property of our LUNA relies on our new private re-randomization results, particularly Theorem 1. These new results of our work (that only impose a poly(κ) condition on the system modulus q for κ-bit zero-knowledge security) are the main reason for the improvements in the SNARG proof size of our approach. As a consequence, we ensure that all conditions in Theorem 1 are satisfied, which particularly means choosing a large enough Gaussian width r for the scaling vectors and a large enough modulus q. In these conditions, we set $\epsilon = 2^{-\kappa}$ for $\kappa = 128$.

Sample parameter sets for LUNA. In light of all constraints and settings described above, we provide a set of sample parameter settings in Table 2. Note that the proof output is a Regev-like ciphertext of $(n + \ell')d \log q'$ bits. In the table, 'crs size' refers to the setting where the random first n columns of the ciphertexts in the CRS are ignored as they can be generated from a small seed in practice. As a result, a crs size is equal to $m_q \ell' d \log q \cdot 2N_g$ bits. The 'crs size full' column refers to the uncompressed full CRS size (including the random n columns) and therefore is equal to $m_q(n + \ell')d \log q \cdot 2N_g$ bits. We note that by choosing $s = 1$, the proof sizes in Table 2 can be reduced by 6-8%.

Parameters of ISW ZK-SNARG at $\kappa = 128$-bit privacy level. As our main motivation in this work is reducing the proof size, we build on the "shorter proof" parameters of ISW [47] to estimate their proof size for $\kappa = 128$. Here, the main change is due to the (exponential) noise smudging, which requires $128 - 40 = 88$ bits larger q compared to the setting of $\kappa = 40$. Therefore, we get $\log q \approx 186$ (instead of $\log q \approx 98$). With the increased q, we need to set a larger dimension parameter for MLWE. Particularly, for the same Gaussian parameter $s = 64$ and ring dimension $d = 2$ in [47], we observed using the MLWE estimator that $n = 3600$ leads to a root Hermite factor of ≈ 1.00427 as before. Since the parameter n is doubled compared to ISW, the modulus q' after mod switching also doubles and so $\log q' = 36$. The other parameters are kept the same as in ISW, i.e., $p = 2^{13} - 1$, $\rho = 26$, $\ell = 109$ and $\tau = 5$. For $N_g = 2^{16}$, this produces a proof of 32.64 KB, a compressed CRS of 663 MB, and a full CRS of 21 GB for ISW at $\kappa = 128$, given in Table 1.

Discussion. Compared to ISW, our proposal reduces the proof size by almost 4× while even achieving smaller CRS sizes. Alternatively, a proof size reduction of close to 6× can be achieved at the cost of increasing CRS sizes without diverging too much from the CRS sizes of ISW. The reason behind sometimes larger CRS in our case is that our ciphertexts in the CRS are matrices (instead of vectors) that are m_q times bigger in dimension. We need this matrix structure due to the use of GSW-like base encryption. We note that our goal in this section is to demonstrate optimized results for varying settings. Such a variation in many parameters may not be desirable in practical implementations. Hence, in Sec. 6, we evaluate our proposal under a fixed parameter setting for increasing constraint sizes N_g.

Asymptotic parameter settings. Asymptotically, we can use the same parameter settings for HGSW as in the previous Section, with $m = 2N_g$. Therefore, based on MLWE hardness against known lattice attacks, the asymptotic proof length is logarithmic in circuit size and quasi-linear in the security parameter λ.

6 IMPLEMENTATION AND EVALUATION

We now summarize our implementation of HGSW encryption and experimental performance evaluation of this implementation[6]. The main goal of our evaluation is to assess the practical performance overhead cost of our re-randomized HGSW vector encryption versus the Regev-based vector encryption used in [47], in a ZK-SNARG context. Since our main focus in this paper is on improving the underlying encryption scheme used in the ZK-SNARG, we do not evaluate the performance costs of the underlying LPCP. The LPCP implementation in the libSNARK package used in the evaluation in [47] requires significant modifications to support the plaintext space extension fields used in HGSW, and we leave such LPCP implementation modifications and optimizations for future work.

6.1 Implementation Techniques

Software and hardware. Our HGSW encryption scheme is written in C++ and compiled using gcc 11.4.0, with the C++17 standard. In order to run the implementation, PALISADE Homomorphic Addition Software Library [28] must be installed. All measurements related to our implementation, and the libsnark-based implementation of ISW [47], were made on the *same* machine running Linux Mint 21.2, equipped with an Intel Xeon E-2314 4-core 2.8 GHz CPU, and 128 GB RAM. Intel Turbo Boost Technology was disabled and -O3 optimization flag was used to enable gcc compiler optimizations for all the experiments.

Implementation parameters. In the parameter settings of our HGSW implementation, we choose $\log q \approx 62.98$, making sure it is the multiplication of two primes, $\bar{q}$ and $p = 19$. Selecting the base $\beta = 54919$, we get $m_q = \lceil \log_\beta q \rceil = 4$. The ring dimension $d = 32$, and the number of ring splitting factors mod p, $\ell_p = 2$. So, we have two irreducible factors of $x^d + 1 \bmod p$ with the degree $f = 16$, which are $f_1 = x^{16} + 6x^8 + 18$ and $f_2 = x^{16} + 13x^8 + 18$. These factors are

[6] Our implementation is available at https://github.com/yassimert/LUNA

p	n	d	ℓ_p	$\log q$	$\log q'$	ℓ	ρ	β	L	$\log r$	c	v	proof size	CRS size	CRS size full
37	64	32	2	49.54	20.79	4	2	8	952	26.73	646	56	5.60	2.06	28.37
37	66	32	2	51.35	20.84	4	2	36	970	28.93	2907	97	5.78	1.25	17.80
277	83	32	2	64.70	24.07	4	2	73988	1032	40.02	5973809	258	8.28	0.63	11.12

Table 2: Example parameter settings of LUNA. The proof size is in KB, and CRS sizes in GB. For all these parameter sets, we set the privacy/security parameter $\kappa = 128$, number of ring splitting factors $\ell_q = 2$, initial LWE Gaussian error parameter $s = 2$, sparsification parameter $\tau = 1$, and the number of R1CS constraints $N_g = 2^{16}$ equal to the number of variables N_w (note that $f = d/\ell_p$ and $m_q = \lceil \log_\beta q \rceil$).

embedded into the implementation and the related Φ matrices used in Chinese Remainder Theorem (CRT) encoding/decoding sections are precomputed to reduce some computations. Additionally, we set plaintext dimension $\ell = 6$, secret key dimension $n = 81$, number of ciphertexts $v = 325$, the parameter $c = 5246585$, number of repetitions for knowledge amplification $\rho = 3$, and crypto Gaussian parameter $r = 2^{40.17}$. We do modulus switching in the decryption, with using $\log q' \approx 20.17$. The remaining parameters are specified in Table 2. This parameter setting is aimed at 128-bit security level as in Sec. 5.2 for all $N_g \leq 2^{20}$ and results in a proof length of 6.93 KB (for all $N_g \leq 2^{20}$ as the parameter set is fixed). We only change the number of constraints N_g between 2^{10} and 2^{20} for performing the experiments. We use the AES-256 CTR mode with the AES-NI to implement the pseudorandom generator.

Ring arithmetic. We use the CRT decomposition for $R_q := R_p \times R_{\bar{q}}$ and compute the R_q arithmetic over its R_p and $R_{\bar{q}}$ CRT components. We choose a Number Theoretic Transform (NTT) friendly prime $\bar{q}$ (i.e. $\bar{q} \equiv 1 \pmod{2d}$) and use the NTT for the arithmetic over $R_{\bar{q}}$. Because p is a small NTT-unfriendly prime, we choose a NTT-friendly prime $p' > dp^2$ such that the largest intermediate value in the R_p arithmetic before mod p can be exactly represented over $\mathbb{Z}_{p'}$. We can then use the NTT over $R_{p'}$ for the R_p arithmetic. For our implemented parameters, setting $p' := \bar{q}$ is sufficient. We keep the results in their NTT domain whenever possible, e.g. the ciphertext $\boldsymbol{C}_i$ in HGSW.Encrypt. We also represent the matrix $\boldsymbol{A}$ in the CRT decomposition and generate its $R_{\bar{q}}$ CRT components in the NTT domain directly during HGSW.Setup. For the $\mathbb{Z}_{\bar{q}}$ modular arithmetic, Plantard's modular reduction [63] using `__uint128_t` is adopted. The NTT-based HGSW algorithm description is in Appendix F.

Gaussian sampling. For the discrete Gaussian sampling over lattices, we use the Gaussian sampling algorithm given in [29]. We do the sampling according to the g^{-1}_{rand} algorithm in Def. 3. For discrete Gaussian sampling over integers, we use `GenerateIntegerKarney` function from the PALISADE library [28], which employs Karney's improved sampling method, based on rejection sampling.

6.2 Evaluation of Our HGSW Performance

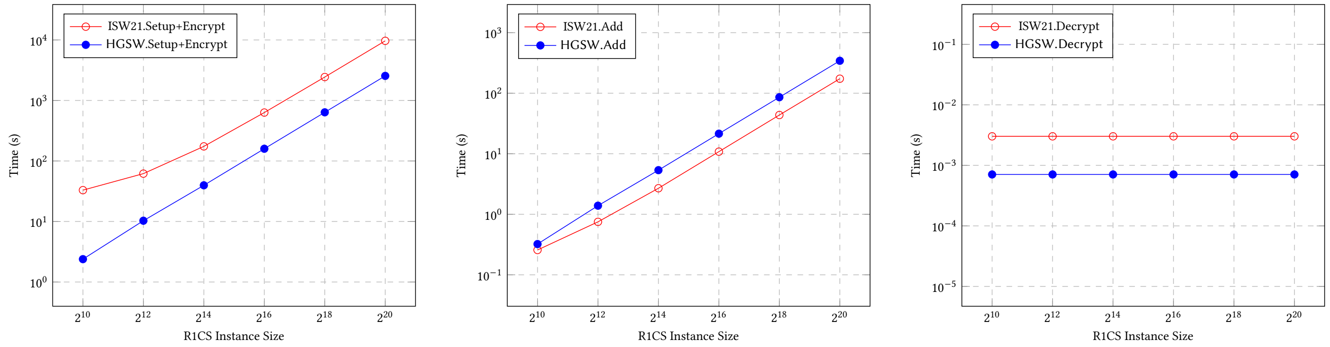
Experimental methodology. In the experiments, we compare the timings of our HGSW-based LO encryption scheme with the Regev-based LO encryption scheme of ISW [47]. For ISW, we measure only the encryption scheme run-time components of ZK-SNARG Setup, Prover and Verifier algorithms, and do *not* include LPCP run-time components (as discussed above). As the setup and encrypt procedures of the LO encryption *collectively* contribute to ZK-SNARG Setup, we measure the sum of the runtimes of these two procedures. In particular, the Setup+Encrypt timings in in this Section are for encryption of *all* $2N_g$ PCP query matrix rows.

While running the implementation of ISW, we used the parameter set "B13C20" ($n = 1815, d = 2, \log q = 98$, etc.), with $p = 2^{13} - 1$. Due to a runtime error (from the underlying libfqfft library) encountered when running the ISW code with $N_g > 2^{14}$, we could only conduct the tests for $N_g = 2^{10}, 2^{12}, 2^{14}$. When measuring the runtimes for ISW, we excluded the LPCP operations and only included the homomorphic Add procedure from Prove, and the Decrypt procedure from Verify. Refer to Table 3 for the timings obtained from these tests. When generating Figure 1 for $N_g = 2^{16}, 2^{18}, 2^{20}$, we extrapolated the runtimes of smaller N_g values due to linear dependence on N_g. As the original ISW implementation is for a lower privacy level of $\kappa = 40$, we consider a slowdown of 2× for Decrypt and 4× for the remaining ISW results obtained from running their original code (on the same machine as ours) when comparing at $\kappa = 128$ privacy level. The reason for these scaling factors is as follows. As explained in 'Parameters of ISW ZK-SNARG at $\kappa = 128$-bit privacy level' in Sec. 5.2, the original $n = 1815$ (for $\kappa = 40$) needs to be increased to $n \approx 3600$ (for $\kappa = 128$). This is almost exactly a 2× increase, yielding 2× more operations in matrix multiplications involved in all components. Moreover, for ring arithmetic mod q, using a 186-bit modulus (for $\kappa = 128$) instead of a 98-bit modulus (for $\kappa = 40$) would result in at least 2× slowdown because ISW implementation uses 128-bit integer arithmetic. Overall, we believe considering a 2× slowdown for Decrypt (where operations are done mod q', which does not change) and 4× slowdown for the rest (where operations are done mod q) over the original ISW runtimes for $\kappa = 40$ are appropriate. In Table 3, we report the scaled ISW runtimes considered at $\kappa = 128$-bit privacy level.

For our encryption scheme, we use the parameters in the 'Implementation Parameters' section above, which provide $\kappa = 128$ bit privacy level. Since we generate the full CRS in our implementation and operate entirely in memory, we were able to perform our tests up to $N_g = 2^{16}$. For larger N_g values, one can simply do matrix multiplications on the fly. Due to the linearity observed in our results, we approximated the timings for larger N_g values in Figure 1, similar to the approach we followed with the ISW results.

Experimental results and comparison. Let us first discuss the communication efficiency (at $\kappa = 128$-bit security level). The parameter setting we use for the implementation achieves a proof length of 6.93 KB, 4.7× reduction compared to ISW21. We also get a bit smaller (about 1-2%) full CRS sizes, and our compressed CRS size is about 30% larger. This is the expected tradeoff between proof and (compressed) CRS sizes observed in Table 1. If one rather prefers a smaller compressed CRS size (at the cost of larger proof sizes), then our scheme can easily support this as given in Table 1.

Looking at the computational efficiency (at $\kappa = 128$-bit security level) from Table 3 and Figure 1, our HGSW Decrypt time achieves a significant speedup of about 4.3×. This is mainly due to our use of smaller system moduli and larger-dimensional polynomial rings, allowing fast NTT-based computation. Similarly in Setup+Encrypt runtimes, our HGSW construction always outperforms ISW and



(a) Comparison of Setup+Encrypt runtimes **(b) Comparison of Add runtimes** **(c) Comparison of Decrypt runtimes**

Figure 1: Runtime comparison between ISW21 [47] and our HGSW.

achieves up to 16× speedup. On the other hand, our HGSW Add runtimes are around 1.2 – 2× slower compared to ISW. This mainly stems from the matrix-vector product, which takes about 50% of its runtimes. In addition, the Gaussian sampling over lattices accounts for around 50% of HGSW.Add runtimes.

Recall that HGSW.Setup+Encrypt runtime roughly corresponds to ZK-SNARG's Setup time, HGSW.Add to ZK-SNARG's Prove time, and HGSW.Decrypt to ZK-SNARG's Verify time. As seen in Table 3, our HGSW Decrypt runtimes are very fast and independent of N_g (as in ISW21). This property is highly desirable for applications where SNARG verification is performed by a lightweight client such as outsourcing computation.

Component	R1CS Instance Size			
	2^{10}	2^{12}	2^{14}	$*2^{16}$
ISW21 Setup+Encrypt	33	62	174	626
HGSW Setup+Encrypt	2	10	40	159
ISW21 Add	0.26	0.75	2.71	10.84
HGSW Add	0.32	1.38	5.36	21.51
ISW21 Decrypt	0.0030	0.0030	0.0030	0.0030
HGSW Decrypt	0.0007	0.0007	0.0007	0.0007

Table 3: Homomorphic encryption runtimes for ISW21 [47] and HGSW (in seconds) at 128-bit security level (refer to Sec. 6.2). *Runtimes of ISW21 for $N_g = 2^{16}$ are extrapolations of those of smaller N_g values.

7 CONCLUSION

We introduced, LUNA, the first lattice-based DV ZK-SNARG protocol with a proof size of around 6 KB for 128-bit security, significantly smaller than the previous 32 KB. This is achieved through a novel re-randomization method of Module LWE samples using discrete Gaussian vectors and establishing smoothing parameter bounds for lattices over cyclotomic rings. Additionally, we have introduced the Module Half-GSW (HGSW) homomorphic encryption scheme. Our implementation results have shown that HGSW provides shorter proofs, quicker CRS generation, and faster encryption and decryption, despite a moderate increase in time for proof generation.

ACKNOWLEDGMENTS

We thank Yang Yu for providing some implementation code of the gadget sampler. This work was supported in part by ARC Discovery Project grants DP180102199 and DP220101234.

REFERENCES

[1] D. Aggarwal and O. Regev. A note on discrete gaussian combinations of lattice vectors. *Chic. J. Theor. Comput. Sci.*, 2016, 2016.

[2] S. Agrawal, C. Gentry, S. Halevi, and A. Sahai. Discrete gaussian leftover hash lemma over infinite domains. In *ASIACRYPT 2013*, volume 8269, pages 97–116, 2013.

[3] S. Agrawal, B. Libert, and D. Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In M. Robshaw and J. Katz, editors, *CRYPTO 2016*, volume 9816, pages 333–362, 2016.

[4] M. R. Albrecht, C. Cid, J. Faugère, R. Fitzpatrick, and L. Perret. Algebraic algorithms for LWE problems. *ACM Commun. Comput. Algebra*, 49(2):62, 2015.

[5] M. R. Albrecht, V. Cini, R. W. F. Lai, G. Malavolta, and S. A. K. Thyagarajan. Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable - (extended abstract). In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 102–132. Springer, 2022. full version at https://eprint.iacr.org/2022/941.

[6] M. R. Albrecht, G. Fenzi, O. Lapiha, and N. K. Nguyen. SLAP: succinct lattice-based polynomial commitments from standard assumptions. *IACR Cryptol. ePrint Arch.*, page 1469, 2023.

[7] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Math. Cryptol.*, 9(3):169–203, 2015.

[8] S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *ACM SIGSAC CCS 2017*, pages 2087–2104, 2017.

[9] S. Atapoor, K. Baghery, H. V. L. Pereira, and J. Spiessens. Verifiable fhe via lattice-based snarks. Cryptology ePrint Archive, Paper 2024/032, 2024. https://eprint.iacr.org/2024/032.

[10] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. In *Mathematische Annalen*, volume 296(4), pages 625—-635, 1993.

[11] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev. Scalable, transparent, and post-quantum secure computational integrity. *IACR Cryptology ePrint Archive*, 2018:46, 2018.

[12] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *IEEE S&P 2014*, pages 459–474. IEEE Computer Society, 2014.

[13] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward. Aurora: Transparent succinct arguments for R1CS. In *EUROCRYPT 2019*, volume 11476, pages 103–128, 2019.

[14] E. Ben-Sasson, L. Goldberg, S. Kopparty, and S. Saraf. DEEP-FRI: sampling outside the box improves soundness. In *ITCS*, volume 151 of *LIPIcs*, pages 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[15] W. Beullens and G. Seiler. Labrador: Compact proofs for R1CS from module-sis. In *CRYPTO (5)*, volume 14085 of *Lecture Notes in Computer Science*, pages 518–548. Springer, 2023.

[16] N. Bitansky, A. Chiesa, Y. Ishai, R. Ostrovsky, and O. Paneth. Succinct non-interactive arguments via linear interactive proofs. In *TCC 2013*, volume 7785, pages 315–333, 2013.

[17] D. Boneh, Y. Ishai, A. Sahai, and D. J. Wu. Lattice-based snargs and their application to more efficient obfuscation. In *EUROCRYPT 2017*, volume 10212, pages 247–277, 2017.

[18] D. Boneh, Y. Ishai, A. Sahai, and D. J. Wu. Quasi-optimal snargs via linear multi-prover interactive proofs. In *EUROCRYPT 2018*, volume 10822, pages 222–255, 2018.

[19] J. Bootle, A. Chiesa, and K. Sotiraki. Lattice-based succinct arguments for NP with polylogarithmic-time verification. In *CRYPTO (2)*, volume 14082 of *Lecture Notes in Computer Science*, pages 227–251. Springer, 2023.

[20] J. Bootle, V. Lyubashevsky, N. K. Nguyen, and G. Seiler. A non-pcp approach to succinct quantum-safe zero-knowledge. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 441–469. Springer, 2020.

[21] J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *IEEE EuroS&P 2018*, pages 353–367, 2018.

[22] F. Bourse and M. Izabachène. Plug-and-play sanitization for tfhe. Cryptology ePrint Archive, Paper 2022/1438, 2022.

[23] F. Bourse, R. D. Pino, M. Minelli, and H. Wee. FHE circuit privacy almost for free. In *CRYPTO*, volume 9815, pages 62–89, 2016.

[24] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3):13:1–13:36, 2014.

[25] L. Brieulle, L. D. Feo, J. Doliskani, J. Flori, and É. Schost. Computing isomorphisms and embeddings of finite fields. *J. Math. Comput.*, 88(317):1391–1426, 2019.

[26] B. Bünz and B. Fisch. Multilinear schwartz-zippel mod N and lattice-based succinct arguments. In *TCC (3)*, volume 14371 of *Lecture Notes in Computer Science*, pages 394–423. Springer, 2023.

[27] V. Cini, R. W. F. Lai, and G. Malavolta. Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In *CRYPTO (2)*, volume 14082 of *Lecture Notes in Computer Science*, pages 72–105. Springer, 2023.

[28] P. Contributors. PALISADE homomorphic addition software library, 2023.

[29] D. B. Cousins, G. Di Crescenzo, K. D. Gür, K. King, Y. Polyakov, K. Rohloff, G. W. Ryan, and E. Savas. Implementing conjunction obfuscation under entropic ring lwe. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 354–371, 2018.

[30] D. Dachman-Soled, H. Gong, M. Kulkarni, and A. Shahverdi. Towards a ring analogue of the leftover hash lemma. *J. of Mathematical Cryptology*, 15(1):87–110, 2021.

[31] T. Debris-Alazard, P. Fallahpour, and D. Stehlé. Quantum oblivious lwe sampling and insecurity of standard model lattice-based snarks. Cryptology ePrint Archive, Paper 2024/030, 2024. https://eprint.iacr.org/2024/030.

[32] Y. Dodis, L. Reyzin, and A. D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT 2004*, volume 3027, pages 523–540, 2004.

[33] L. Ducas and D. Stehlé. Sanitization of FHE ciphertexts. In *EUROCRYPT 2016*, volume 9665, pages 294–310, 2016.

[34] M. F. Esgin, R. Steinfeld, D. Liu, and S. Ruj. Efficient hybrid exact/relaxed lattice proofs and applications to rounding and vrfs. In *CRYPTO (5)*, volume 14085 of *LNCS*, pages 484–517. Springer, 2023.

[35] M. F. Esgin, R. Steinfeld, J. K. Liu, and D. Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In *CRYPTO (1)*, volume 11692 of *LNCS*, pages 115–146. Springer, 2019.

[36] M. F. Esgin, R. Steinfeld, and R. K. Zhao. MatRiCT$^+$: More efficient post-quantum private blockchain payments. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1281–1298. IEEE, 2022. (Full version at ia.cr/2021/545).

[37] G. Fenzi, H. Moghaddas, and N. K. Nguyen. Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. Cryptology ePrint Archive, Paper 2023/846, 2023. https://eprint.iacr.org/2023/846.

[38] N. Genise, D. Micciancio, C. Peikert, and M. Walter. Improved discrete gaussian and subgaussian analysis for lattice cryptography. In A. Kiayias, M. Kohlweiss, P. Wallden, and V. Zikas, editors, *Public-Key Cryptography – PKC 2020*, pages 623–651, Cham, 2020. Springer International Publishing.

[39] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT 2013*, volume 7881, pages 626–645, 2013.

[40] R. Gennaro, M. Minelli, A. Nitulescu, and M. Orrù. Lattice-based zk-snarks from square span programs. In *ACM SIGSAC CCS 2018*, pages 556–573. ACM, 2018.

[41] C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, USA, 2009.

[42] C. Gentry, S. Halevi, and V. Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In *EUROCRYPT (1)*, volume 13275, pages 458–487, 2022.

[43] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *ACM STOC, 2008*, pages 197–206. ACM, 2008.

[44] C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO 2013*, volume 8042, pages 75–92, 2013.

[45] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *ACM STOC 1985*, pages 291–304. ACM, 1985.

[46] J. Groth. On the size of pairing-based non-interactive arguments. In *EUROCRYPT 2016*, volume 9666, pages 305–326, 2016.

[47] Y. Ishai, H. Su, and D. J. Wu. Shorter and faster post-quantum designated-verifier zksnarks from lattices. In *ACM SIGSAC CCS 2021*, CCS '21, page 212–234, 2021.

[48] J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *ACM STOC 1992*, pages 723–732. ACM, 1992.

[49] E. Kirshanova, H. Nguyen, D. Stehlé, and A. Wallet. On the smoothing parameter and last minimum of random orthogonal lattices. *Des. Codes Cryptogr.*, 88(5):931–950, 2020.

[50] K. Kluczniak. Circuit privacy for fhew/tfhe-style fully homomorphic encryption in practice. Cryptology ePrint Archive, Paper 2022/1459, 2022.

[51] A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015.

[52] B. Libert, A. Sakzad, D. Stehlé, and R. Steinfeld. All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from LWE. In *CRYPTO (3)*, volume 10403, pages 332–364, 2017.

[53] V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT 2012*, volume 7237, pages 738–755, 2012.

[54] V. Lyubashevsky, N. K. Nguyen, and M. Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *CRYPTO (2)*, volume 13508 of *LNCS*, pages 71–101. Springer, 2022.

[55] V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-lwe cryptography. In *EUROCRYPT*, volume 7881, pages 35–54, 2013.

[56] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841, pages 465–484, 2011.

[57] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT 2012*, volume 7237, pages 700–718, 2012.

[58] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[59] N. K. Nguyen. On the non-existence of short vectors in random module lattices. In *ASIACRYPT 2019*, volume 11922, pages 121–150, 2019.

[60] A. Nitulescu. Lattice-based zero-knowledge snargs for arithmetic circuits. In *LATINCRYPT 2019*, volume 11774, pages 217–236, 2019.

[61] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *IEEE S&P*, pages 238–252, 2013.

[62] C. Peikert. Limits on the hardness of lattice problems in ℓ_p norms. In *IEEE CCC 2007)*, pages 333–346. IEEE Computer Society, 2007.

[63] T. Plantard. Efficient word size modular arithmetic. *IEEE Trans. Emerg. Top. Comput.*, 9(3):1506–1518, 2021.

[64] M. Rosca, D. Stehlé, and A. Wallet. On the ring-lwe and polynomial-lwe problems. In *EUROCRYPT (1)*, volume 10820, pages 146–173, 2018.

[65] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *EUROCRYPT 2011*, volume 6632, pages 27–47, 2011.

[66] D. Stehlé and R. Steinfeld. Making ntruencrypt and ntrusign as secure as standard worst-case problems over ideal lattices. *IACR Cryptol. ePrint Arch.*, page 4, 2013.

[67] H. Wee and D. J. Wu. Lattice-based functional commitments: Fast verification and cryptanalysis. In *ASIACRYPT (5)*, volume 14442 of *Lecture Notes in Computer Science*, pages 201–235. Springer, 2023.

A A NOTE ON THE FLAVOUR OF LINEAR-ONLY ASSUMPTIONS REQUIRED FOR ZK-SNARGs AND ZK-SNARKs

Flavours of LO. In their paper on a cryptographic compiler for constructing a ZK-SNARG from a LPCP and a linear-only (LO) encryption scheme [16], Bitansky et al. considered various alternative flavours of assumptions on the underlying LO encryption scheme and LPCP ingredients. In particular, they considered a strong *knowledge-based* flavour of LO encryption called Linear-Only Homomorphism (LOH) (See Def 5.4 in [16]), and two weaker *simulation-based* flavours of LO encryption called statistically simultable (resp. computationally simulatable) Linear Targeted Malleability (LTM) (see Remark 6.4 in [16] and Def. 4.2 in [17], resp. Def. 5.8 in [16]).

Strong (Knowledge-Based) LOH Assumption. Informally, the LO requirement means that the only functions $f(\mu_1, \ldots, \mu_m)$ that can be computed efficiently homomorphically by an adversary on a given set of ciphertexts of plaintexts $(\mu_1, \ldots, \mu_m)$, are linear functions $f(\mu_1, \ldots, \mu_m) = \sum_i \pi_i \mu_i$ for some coefficients π_i. The stronger

LOH knowledge-based flavour of this assumption informally requires the existence of an efficient extractor algorithm that can extract from the adversary the knowledge of the linear coefficients π_i used in its homomorphic computation. Assuming the knowledge-based LOH assumption on the encryption scheme, the compiler of [16] was shown (Lemma 6.2 in [16]) to result in a ZK-SNARK (i.e. satisfying an efficient knowledge extractor-based soundness property).

Recent Attacks on LOH Assumption in Lattice Setting. The above knowledge-based LOH assumption was adapted by subsequent work in the lattice-setting by [40, 47, 60]. However, in the lattice-setting, this LOH knowledge assumption is essentially equivalent to the assumption that it is computationally difficult to sample LWE samples without knowing the underlying LWE secret/error. A very recent work [31]showed that the latter LWE knowledge assumption is invalid against efficient quantum attacks. In particular, it gives a polynomial-time *oblivious sampling* quantum algorithm to generate LWE samples without knowledge of the underlying solution (we also remark that a heuristic but *classical* oblivious sampler for a variant of LWE knowledge assumption defined in [5] was recently given in [67]). Therefore, the knowledge-based LOH assumption required from the underlying LWE-based encryption in the ZK-SNARK protocols of [40, 47, 60] does not hold against quantum attacks. Nevertheless, note that, as observed in [31], only the underlying *assumption* flavour is broken by the attack, but the ZK-SNARK protocols may still achieve soundness under a weaker LO assumption flavour.

Our LO Assumption: statistically simulatable LTM. In this paper we base the ZK-SNARG soundness security of LUNA on a weaker flavour of LO assumption on the underlying homomorphic HGSW encryption scheme, namely simulation-based statistically simulatable LTM defined in [16, 17]. Probabilistically, statistically simulatable LTM is captured by the existence of a *computationally unbounded* simulator that can simulate with as $\sum_i \pi_i \mu_i$ the distribution of the decrypted ciphertext output by the adversary, for some distribution on the coefficients π_i chosen by the simulator. It was shown in Theorem 4.6 of [17] (and also observed in Remark 6.4 in [16]) that statistically simulatable LTM suffices for the compiler of [16] to produce a ZK-SNARG with non-adaptive soundness. Moreover the weaker LTM flavour of LO is plausible even in the lattice-based LWE setting. Indeed, breaking the linear targeted malleability assumption of HGSW (or other lattice-based schemes used in the above protocols) requires the attack to actually compute non-linear homomorphisms on the ciphertext, and the quantum oblivious sampler attack algorithm of [31] does not seem to help with this latter problem, as also acknowledged in [31].

Upgrading our construction to ZK-SNARK from computationally simulatable LTM. Under a stronger assumed flavour of simulation based LTM assumption on the HGSW encryption scheme, namely LTM with computationally efficient simulation, and assuming knowledge-based statistical soundness for the underlying LPCP, LUNA is also ZK-SNARK with non-adaptive soundness (i.e. satisfying knowledge-based soundness), rather than only a ZK-SNARG. See Sec. 5.1 for more details. The stronger LTM with computationally efficient simulation assumption is also unaffected by the recent attacks of [31].

B ADDITIONAL PRELIMINARIES

B.1 Table of Notations.

In Table 4, we have summarized all important notations used in this paper.

B.2 Additional Lattice Definitions and Preliminaries.

Definition 5 (Gaussian Function). *For any $r > 0$ the Gaussian function with parameter r*[7] *and for any $\boldsymbol{x} \in \mathbb{R}^n$ is defined as $\rho_r(\boldsymbol{x}) = \exp(-\pi \|\boldsymbol{x}\|^2/r^2)$. Given a lattice $\Lambda \subseteq \mathbb{R}^n$, a parameter r and a vector $\boldsymbol{c} \in \mathbb{R}^n$, the discrete Gaussian distribution with parameter r and support $\Lambda + \boldsymbol{c}$ is defined as*

$$\mathcal{D}_{\Lambda+\boldsymbol{c},r}(\boldsymbol{x}) = \frac{\rho_r(\boldsymbol{x})}{\rho_r(\Lambda + \boldsymbol{c})}, \ \forall \boldsymbol{x} \in \Lambda + \boldsymbol{c},$$

where $\rho_r(\Lambda + \boldsymbol{c}) = \sum_{\boldsymbol{x} \in \Lambda + \boldsymbol{c}} \rho_r(\boldsymbol{x})$.

Lemma 6 ([53], Lemma 4.4). *For $r > 0$, $n \geq 1$ and $k > 1$, we have*

$$\Pr_{\boldsymbol{z} \hookleftarrow \mathcal{D}_{\mathbb{Z}^n,r}}[\|\boldsymbol{z}\| > kr\sqrt{n/(2\pi)}] < k^n \cdot \exp(n/2 \cdot (1 - k^2)).$$

Lemma 7 ([58], Lemma 4.4). *Let Λ be any n-dimensional lattice. Then for any $\epsilon \in (0, 1)$, $r \geq \eta_\epsilon(\Lambda)$, and $\boldsymbol{c} \in \mathbb{R}^n$, we have $\Pr_{\boldsymbol{z} \hookleftarrow \mathcal{D}_{\Lambda+\boldsymbol{c},r}}[\|\boldsymbol{z}\| > r\sqrt{n}] \leq (1 + \epsilon)/(1 - \epsilon) \cdot 2^{-n}$.*

Lemma 8 ([53], Lemma 4.3). *For $n \geq 1$, $\boldsymbol{v} \in \mathbb{R}^n$, $\sigma, r, t > 0$, we have that $\Pr_{\boldsymbol{z} \hookleftarrow \mathcal{D}_{\mathbb{Z}^n,r}}[|\langle \boldsymbol{v}, \boldsymbol{z} \rangle| > t \cdot r\|\boldsymbol{v}\|] \leq 2\exp(-\pi t^2)$.*

Lemma 9 ([43, 58]). *Let Λ be any n-dimensional lattice. Then for any $\epsilon \in (0, 1)$, $r \geq \eta_\epsilon(\Lambda)$, and $\boldsymbol{c} \in \mathbb{R}^n$, we have $\rho_r(\boldsymbol{c} + \Lambda) \in [(1 + \epsilon)/(1 - \epsilon), 1] \cdot \rho_r(\Lambda)$.*

Lemma 10 (Banasczcyk [10]). *For any rank$-n$ lattice $\Lambda \subset \mathbb{R}^m$ and for all $i \in [n]$, we have $1 \leq \lambda_i(\Lambda) \cdot \lambda_{n-i+1}(\Lambda^*) \leq n$.*

Lemma 11 ([62]). *For any n-dimensional lattice Λ and real $\epsilon > 0$, we have*

$$\eta_\epsilon(\Lambda) \leq \frac{\sqrt{\ln(2n(1 + \epsilon^{-1}))/\pi}}{\lambda_1^\infty(\Lambda^*)}.$$

Lemma 12 ([49], Lemma 7). *For any rank-d lattice Λ and $\epsilon \in (0, 1/2)$, we have $\lambda_d(\Lambda)/\sqrt{d} \leq \eta_\epsilon(\Lambda) \leq \lambda_d(\Lambda) \cdot \sqrt{\ln(2d(1 + \epsilon^{-1}))/\pi}$.*

B.3 Lattices over polynomial rings

For the polynomial ring $R := \mathbb{Z}[x]/(x^d + 1)$, where d is a power of 2, and $a, b \in R$, the ring multiplication $c = ab$ corresponds over $\mathbb{Z}$ to a matrix product $\text{rot}(c) = \text{rot}(a) \cdot \text{rot}(b)$, where for a ring element a, $\text{rot}(a) \in \mathbb{Z}^{d \times d}$ denotes the negacyclic matrix whose i'th row consists of the coefficient vector of $x^i a \bmod x^d + 1$, for $i = 0, \ldots, d - 1$. Similarly, for a matrix $\boldsymbol{A} \in R^{l \times n}$, we let $\text{rot}(\boldsymbol{A}) \in \mathbb{Z}^{ld \times nd}$ be the corresponding representation of $\boldsymbol{A}$ over $\mathbb{Z}$, where each ring element of $\boldsymbol{A}$ is replaced by its rot matrix, and we analogously use $\text{rot}(\boldsymbol{A})$ to define the ld-dimensional q-ary lattices $\Lambda_q^\perp(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^{ld} | \boldsymbol{x}^T \text{rot}(\boldsymbol{A}) = \boldsymbol{0} \bmod q\}$ and $\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^{ld} | \boldsymbol{v} = \text{rot}(\boldsymbol{A})\boldsymbol{s} \bmod q, \text{ for some } \boldsymbol{s} \in \mathbb{Z}^{nd}\}$ over $\mathbb{Z}$. The lattice $\Lambda_q(\boldsymbol{A})$ is related to the lattice $\widetilde{\Lambda_q}(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^{ld} | \boldsymbol{v} \text{ is the coeff. vector of } \boldsymbol{A}\boldsymbol{s} \bmod q, \text{ for some } \boldsymbol{s} \in R^n\}$ by a full rank and norm-preserving linear transformation (this is the

[7] Note that the parameter r is related to the standard deviation σ by $r = \sqrt{2\pi} \cdot \sigma$.

Notation	Explanation	Remarks
N_g	the number of constraints	
N_w	the witness length (i.e., $\boldsymbol{w} \in \mathbb{F}^{N_w}$)	$N_w \approx N_g$
n	MLWE secret key dimension over R_q	
k	the number of PCP queries	$k = 4$
m	the PCP query length	$m = 2N_g$
$\boldsymbol{Q} \in \mathbb{F}^{m \times k}$	the PCP query matrix	
$\boldsymbol{\pi} : \mathbb{F}^m \longrightarrow \mathbb{F}$	the linear oracle	
ρ	num. of repetitions for knowledge amplification	
p	the plaintext modulus	
q	the ciphertext modulus before mod switching	
q'	the ciphertext modulus after mod switching	
R	the polynomial ring of dimension d	$R = \mathbb{Z}[x]/(x^d + 1)$
ℓ_p	number of ring splitting factors mod p	
f	degree of the irreducible factors of $x^d + 1$ mod p	$d = f \cdot \ell_p$
ℓ_q	number of ring splitting factors mod q	
ℓ	the plaintext dimension over R_p	$\ell = \lceil 4\rho/\ell_p \rceil$
τ	sparsification parameter	$\tau = \lceil \frac{128}{d \log p} \rceil$
$\ell' = \ell + \tau$	extended plaintext dimension over R_p	
χ	LWE error distribution	
s	Gaussian parameter for initial LWE error	
r	crypto Gaussian parameter for re-randomization	
κ	statistical ZK security parameter	
m_q	number of digits of q base β	$m_q = \lceil \log_\beta q \rceil$
v	num. of ctxts in re-randomization privacy analysis	
L	num. of rows of the uniform re-randomization matrix	$L = v \cdot m_q$

Table 4: List of common parameters and their definitions.

transformation which maps the first column of a rot matrix to its first row, applied to each coefficient vector; for the ring R this is the mapping that maps the coefficient vector $(a_0, a_1, \ldots, a_{d-1})$ to $(a_0, -a_{d-1}, \ldots, -a_1)$). Therefore, the minima of the latter two lattices are the same, and hence (in a slight abuse of notation), we do not distinguish between those two definitions of those lattices in our analysis of their minima, and refer to both as $\Lambda_q(\boldsymbol{A})$.

B.4 Linear PCP

Definition 6. *(Linear PCP [47]) Let p be a polynomial and let $\mathcal{CS} = \{\mathcal{CS}_N\}_N$ be the family of R1CS systems over a finite field $\mathbb{F}$, where each system $\mathcal{CS}_N = (n_N, N_{g,N}, N_{w,N}, \{\boldsymbol{a}_{i,N}, \boldsymbol{b}_{i,N}, \boldsymbol{c}_{i,N}\}_{i \in [N_{g,N}]})$ has size at most $|\mathcal{CS}_N| \leq p(N)$.*
In the following, we write $n = n(N)$ to denote a polynomially-bounded function where $n(N) = n_N$ for all $N \in \mathbb{N}$. We define $N_g = N_g(N)$ and $N_w = N_w(N)$ similarly. A k-query input-independent linear PCP for $\mathcal{CS}$ with query length $m = m(N)$ and knowledge error $\epsilon = \epsilon(N)$ is a tuple of algorithms $\Pi_{\mathsf{LPCP}} = (\Pi_{\mathsf{LPCP}}.\mathsf{Query}, \Pi_{\mathsf{LPCP}}.\mathsf{Prove}, \Pi_{\mathsf{LPCP}}.\mathsf{Verify})$ with the following properties:

- $(\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)$*: The query-generation algorithm takes as input the system index $N \in \mathbb{N}$ and outputs a query matrix $\boldsymbol{Q} \in \mathbb{F}^{m \times k}$ and a verification state* st.
- $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$*: On input the system index $N \in \mathbb{N}$, a statement $\boldsymbol{x} \in \mathbb{F}^n$, and a witness $\boldsymbol{w} \in \mathbb{F}^{N_w}$, the prove algorithm outputs a proof $\boldsymbol{\pi} \in \mathbb{F}^m$.*
- $b \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{a})$*: On input the verification state* st*, the statement $\boldsymbol{x} \in \mathbb{F}^n$, and a vector of responses $\boldsymbol{a} \in \mathbb{F}^k$, the verification algorithm outputs a bit $b \in \{0, 1\}$.*

In addition, Π_{LPCP} should satisfy the Completeness, HVZK, and either Statistical Knowledge Soundness or Statistical Soundness properties:

- **Completeness:** For all $N \in \mathbb{N}, \boldsymbol{x} \in \mathbb{F}^n$, $\boldsymbol{w} \in \mathbb{F}^{N_w}$ where $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$,
$$\Pr[\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}^T\boldsymbol{Q}) = 1 | (\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N),$$
$$\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})] = 1.$$
- **Statistical Knowledge Soundness:** There exists an efficient extractor $\mathcal{E}_{\mathsf{LPCP}}$ such that for all $N \in \mathbb{N}, \boldsymbol{x} \in \mathbb{F}^n$, and $\boldsymbol{\pi}^* \in \mathbb{F}^m$, if
$$\Pr[\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, (\boldsymbol{\pi}^*)^T\boldsymbol{Q}) = 1 | (\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)] > \epsilon,$$
then
$$\Pr[\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1 | \boldsymbol{w} \leftarrow \mathcal{E}_{\mathsf{LPCP}}^{\langle \boldsymbol{\pi}^*, \cdot \rangle}(1^N, \boldsymbol{x})] = 1,$$
where ϵ denotes the knowledge error of the linear PCP.
- **Statistical Soundness:** For all $\boldsymbol{x} \in \mathbb{F}^n$ such that $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 0$ for all $\boldsymbol{w} \in \mathbb{F}^{N_w}$, and for all $\pi^* \in \mathbf{F}^m$,
$$\Pr[\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, (\boldsymbol{\pi}^*)^T\boldsymbol{Q}) = 1$$

$$|(\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)] \leq \epsilon,$$

where ϵ denotes the soundness error of the linear PCP.

- **Perfect honest-verifier zero knowledge (HVZK):** There exists an efficient simulator $\mathcal{S}_{\mathsf{LPCP}} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all $N \in \mathbb{N}$ and all instances $(\boldsymbol{x}, \boldsymbol{w})$ where $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$,
$$\{(\mathsf{st}, \boldsymbol{Q}, \boldsymbol{\pi}^T \boldsymbol{Q})\} \equiv \{(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{a}})\},$$
where $(\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)$, $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$, $(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^N)$, and $\tilde{\boldsymbol{a}} \leftarrow \mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, \boldsymbol{x})$.

The Honest-Verifier Zero0Knowledge with Leakage (HVZKL) property of LPCPs was defined as follows [47].

Definition 7 (Honest-Verifier Zero-Knowledge with Leakage [47]). *Let $R = \mathbb{Z}[X]/f(X)$ be a polynomial ring where $\deg(f) = d$. Let p be a prime such that $R_p \cong \mathbb{F}_{p^d}$ is a finite field. Let $\Pi_{\mathsf{LPCP}} = (\mathsf{Query}_{\mathsf{LPCP}}, \mathsf{Prove}_{\mathsf{LPCP}}, \mathsf{Verify}_{\mathsf{LPCP}})$ be a linear PCP for a family of R1CS systems $\mathcal{CS} = \{\mathcal{CS}_\kappa\}_{\kappa \in \mathbb{N}}$ over R_p. Let $\mathcal{D}$ be a distribution on matrices over R and $q > p$ be a modulus. We say that Π_{LPCP} satisfies honest-verifier zero-knowledge with $(\mathcal{D}, q)$-leakage if there exists an efficient simulator $\mathcal{S}_{\mathsf{LPCP}} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all $\kappa \in \mathbb{N}$ and all instances $(\boldsymbol{x}, \boldsymbol{w})$ where $\mathcal{CS}_\kappa(\boldsymbol{x}, \boldsymbol{w}) = 1$,*

$$\{(\mathsf{st}, \boldsymbol{Q}, [\boldsymbol{Q}\boldsymbol{\pi}]_q, Z, [Z\boldsymbol{\pi}]_q)\} \overset{s}{\approx} \{(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{a}}, \tilde{Z}, \tilde{\boldsymbol{b}})\}, \quad (18)$$

where $(\mathsf{st}, \boldsymbol{Q}) \leftarrow \mathsf{Query}_{\mathsf{LPCP}}(1^\kappa)$, $Z \leftarrow \mathcal{D}$, $\boldsymbol{\pi} \leftarrow \mathsf{Prove}_{\mathsf{LPCP}}(1^{\kappa, \boldsymbol{x}, \boldsymbol{w}})$, $(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \tilde{Z}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^\kappa)$ and $(\tilde{\boldsymbol{a}}, \tilde{\boldsymbol{b}}) \leftarrow \mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, \boldsymbol{x})$, and we write $[\boldsymbol{Q}\boldsymbol{\pi}]_q$ and $[Z\boldsymbol{\pi}]_q$ to denote computations over the ring R_q (i.e. the elements of R_q are first lifted to R and the value of the matrix-vector product is then reduced modulo q). When the statistical distance between the two distributions in (18) *is δ, we say that Π_{LPCP} is δ-HVZK with $(\mathcal{D}, q)$-leakage.*

B.5 SNARG and SNARK

Definition 8. *(Succinct Non-Interactive Argument [17]) Let $\mathcal{CS} = \{\mathcal{CS}_N\}_{N \in \mathbb{N}}$ be a family of R1CS systems over a finite field $\mathbb{F}$, where $|\mathcal{CS}_N| \leq s(N)$ for some fixed polynomial $s(\cdot)$. A succinct non-interactive argument in the pre-processing model for $\mathcal{CS}$ is a tuple $\Pi_{\mathsf{SNARG}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ with the following properties:*

- $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$*: On input the security parameter λ and the system index N, the setup algorithm outputs a common reference string* crs *and verification state* st.
- $\boldsymbol{\pi} \leftarrow \mathsf{Prove}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$*: On input a common reference string* crs*, a statement $\boldsymbol{x}$ and a witness $\boldsymbol{w}$, the prove algorithms outputs a proof $\boldsymbol{\pi}$.*
- $b \leftarrow \mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi})$*: On input the verification state* st*, a statement $\boldsymbol{x}$ and a proof $\boldsymbol{\pi}$, the verification algorithm outputs a bit $b \in \{0, 1\}$.*

A secure (non-adaptive) Succinct Non-Interactive Argument of Knowledge (SNARG) Π_{SNARG} should satisfy the following properties:

- ***Completeness**: For all security parameters $\lambda \in \mathbb{N}$, system indices $N \in \mathbb{N}$, and instances $(\boldsymbol{x}, \boldsymbol{w})$ where $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$,*
$$\Pr[\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}) = 1] = 1,$$
where $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$, $\boldsymbol{\pi} \leftarrow \mathsf{Prove}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$.
- ***Non-Adaptive Soundness:** For all polynomial-size provers $\mathcal{P}^*$, and all statements $\boldsymbol{x}$ such that $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) \neq 1$ for all $\boldsymbol{w}$,*
$$\Pr[\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}) = 1] = \mathsf{negl}(\lambda),$$
where $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$*,* $\boldsymbol{\pi} \leftarrow \mathcal{P}^*(\mathsf{crs}, \boldsymbol{x})$.
- ***Efficiency:** There exist a universal polynomial p (independent of $\mathcal{CS}$) such that* Setup *and* Prove *run in time $p(\lambda + |\mathcal{CS}_N|)$, Verify runs in time $p(\lambda + |\boldsymbol{x}| + \log|\mathcal{CS}_N|)$, and the proof size is $p(\lambda + \log|\mathcal{CS}_N|)$.*

Definition 9. *(Succinct Non-Interactive ARgument of Knowledge - adapted from [16]) A triple of algorithms $\Pi_{\mathsf{SNARK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ is a (non-adaptive) succinct non-interactive ARgument of Knowledge (SNARK) in the pre-processing model for R1CS system $\mathcal{CS}$ if Π_{SNARK} is a (non-adaptive) SNARG for $\mathcal{CS}$, where the non-adaptive soundness is replaced by the following stronger requirement:*

- ***Non-Adaptive Knowledge Soundness:** For all polynomial-size provers $\mathcal{P}^*$, there exists a polynomial-size extractor $\mathcal{E}$, such that for all security parameters $\lambda \in \mathbb{N}$, system indices $N \in \mathbb{N}$, statements $\boldsymbol{x} \in \{0, 1\}^{\mathsf{poly}(\lambda)}$, and auxiliary inputs $z \in \{0, 1\}^{\mathsf{poly}(\lambda)}$,*
$$\Pr[\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}) = 1 \land \mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) \neq 1] = \mathsf{negl}(\lambda),$$
where $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$*,* $\boldsymbol{\pi} \leftarrow \mathcal{P}^*(1^*, 1^N, \boldsymbol{x}, \mathsf{crs}; z)$*, and* $\boldsymbol{w} \leftarrow \mathcal{E}(1^\lambda, 1^N, \mathsf{crs}, \mathsf{st}, \boldsymbol{x}; z)$.

Definition 10. *(Zero-Knowledge SNARG (resp. SNARK) A* SNARG *(resp. SNARK)* $\Pi_{\mathsf{SNARG}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *for an R1CS system $\mathcal{CS} = \{\mathcal{CS}_N\}_{N \in \mathbb{N}}$ is statistically zero knowledge, i.e., a (statistical) ZK-SNARG (resp. ZK-SNARK) if there exists an efficient simulator $\mathcal{S}_{\mathsf{SNARG}} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for all $N \in \mathbb{N}$ and all stateful adversaries $\mathcal{A}$, we have that*

$$\Pr[\mathsf{ExptZK}_{\Pi_{\mathsf{SNARG}}, \mathcal{A}, \mathcal{S}_{\mathsf{SNARG}}}(1^\lambda, 1^N) = 1] \leq 1/2 + \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExptZK}_{\Pi_{\mathsf{SNARG}}, \mathcal{A}, \mathcal{S}_{\mathsf{SNARG}}}(1^\lambda, 1^N)$ *is defined as follows:*

1. *The challenger samples $b \hookleftarrow \mathcal{U}(\{0, 1\})$. If $b = 0$, the challenger computes* $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$ *and gives* $(\mathsf{crs}, \mathsf{st})$ *to $\mathcal{A}$. If $b = 1$, the challenger computes* $(\tilde{\mathsf{crs}}, \tilde{\mathsf{st}}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^\lambda, 1^N)$ *and gives* $(\tilde{\mathsf{crs}}, \tilde{\mathsf{st}})$ *to $\mathcal{A}$.*
2. *The adversary $\mathcal{A}$ outputs a statement x and a witness $\boldsymbol{w}$.*
3. *If $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) \neq 1$, then the experiment halts with output* 0*. Otherwise, the challenger proceeds as follows:*
    - *If $b = 0$, the challenger replies with* $\boldsymbol{\pi} \leftarrow \mathsf{Prove}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$.
    - *If $b = 1$, the challenger replies with* $\tilde{\boldsymbol{\pi}} \leftarrow \mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, \boldsymbol{x})$.

   At the end of the experiment, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$. The output of the experiment is 1 *if $b' = b$ and is* 0 *otherwise.*

We say a SNARG (resp. SNARK) is designated verifier if st cannot be efficiently computed from the crs.

We now recall the construction of the LPCP used in the ZK-SNARG of [47].

Definition 11 (LPCP for R1CS construction [47]). *Let $\mathcal{CS} = \{\mathcal{CS}_N\}_{N \in \mathbb{N}}$ be a family of R1CS instances over a finite field $\mathbb{F}$, where $\mathcal{CS}_N = (n_N, N_{g,N}, N_{w,N}, \{\boldsymbol{a}_{i,N}, \boldsymbol{b}_{i,N}, \boldsymbol{c}_{i,N}\}_{i \in [N_{g,N}]}), \boldsymbol{a}_{i,N}, \boldsymbol{b}_{i,N}, \boldsymbol{c}_{i,N} \in \mathbb{F}^{N_{w,N}+1}$. We define $N_g = N_g(N)$, $N_w = N_w(N)$, $\boldsymbol{a}_i = \boldsymbol{a}_i(N)$, $\boldsymbol{b}_i = \boldsymbol{b}_i(N)$, $\boldsymbol{c}_i = \boldsymbol{c}_i(N)$. We additionally define:*

- *$S = \{\alpha_1, \ldots, \alpha_{N_g}\} \subset \mathbb{F}$ be an arbitrary subset of $\mathbb{F}$.*

- *For each* $i \in \{0, \ldots, N_w\}$ *let* $A_i, B_i, C_i : \mathbb{F} \to \mathbb{F}$ *unique polynomials of degree* $N_g - 1$ *and for all* $j \in [N_g]$*:*
$$A_i(\alpha_j) = \boldsymbol{a}_{j,i},\ B_i(\alpha_j) = \boldsymbol{b}_{j,i},\ C_i(\alpha_j) = \boldsymbol{c}_{j,i}$$
- *Let* $Z_S : \mathbb{F} \to \mathbb{F}$ *be the polynomial* $Z_S(z) := \prod_{j \in [N_g]}(z - \alpha_j)$*.*

The 4-query LPCP $\Pi_{\mathsf{LPCP}} = (\Pi_{\mathsf{LPCP}}.\mathsf{Query}, \Pi_{\mathsf{LPCP}}.\mathsf{Prove}, \Pi_{\mathsf{LPCP}}.\mathsf{Verify})$ *for* $\mathcal{CS}$ *is defined as follows:*

$\Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)$**:** *On input* $N \in \mathbb{N}$*, sample* $\tau \hookleftarrow \mathcal{U}(\mathbb{F} \setminus S)$*. Let* $\boldsymbol{a} = (A_1(\tau), \ldots, A_n(\tau)), \boldsymbol{b} = (B_1(\tau), \ldots, B_n(\tau)), \boldsymbol{c} = (C_1(\tau), \ldots, C_n(\tau))$*. Output the state* $\mathsf{st} = (A_0(\tau), B_0(\tau), C_0(\tau), \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, Z_S(\tau))$ *and the query matrix:*

$$\boldsymbol{Q} = \begin{bmatrix} Z_S(\tau) & 0 & 0 & A_{n+1}(\tau) & \cdots & A_{N_w}(\tau) & 0 & 0 & \cdots & 0 \\ 0 & Z_S(\tau) & 0 & B_{n+1}(\tau) & \cdots & B_{N_w}(\tau) & 0 & 0 & \cdots & 0 \\ 0 & 0 & Z_S(\tau) & C_{n+1}(\tau) & \cdots & C_{N_w}(\tau) & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & \tau & \cdots & \tau^{N_g} \end{bmatrix}^T$$
$$\in \mathbb{F}^{(4+N_g+N_w-n)\times 4}$$

$\Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$**:** *On input* $N \in \mathbb{N}$ *and an instance* $(\boldsymbol{x}, \boldsymbol{w})$ *where* $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$*, sample* $\delta_1, \delta_2, \delta_3 \hookleftarrow \mathcal{U}(\mathbb{F})$*. Construct polynomials* $A, B, C : \mathbb{F} \to \mathbb{F}$*, each of degree* N_g*:*

$$A(z) := \delta_1 Z_S(z) + A_0(z) + \sum_{i \in N_w} w_i A_i(z)$$
$$B(z) := \delta_2 Z_S(z) + B_0(z) + \sum_{i \in N_w} w_i B_i(z)$$
$$C(z) := \delta_3 Z_S(z) + C_0(z) + \sum_{i \in N_w} w_i C_i(z)$$

Let $H(z) := (A(z)B(z) - C(z))\ / Z_S(z)$ *and let* $\boldsymbol{h} = (h_0, \ldots, h_{N_g}) \in \mathbb{F}^{N_g+1}$ *be the coefficients of* H*. Parse* $\boldsymbol{w}^T = [\boldsymbol{x}^T | \bar{\boldsymbol{w}}^T]$*. Output the proof vector* $\boldsymbol{\pi} = (\delta_1, \delta_2, \delta_3, \bar{\boldsymbol{w}}, \boldsymbol{h}) \in \mathbb{F}^{4+N_g+N_w-n}$*.*

$\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{a})$**:** *On input* $\mathsf{st} = (a_0, b_0, c_0, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, z), \boldsymbol{x} \in \mathbb{F}^n$ *and* $\boldsymbol{a} \in \mathbb{F}^4$*, the verifier computes* $a'_1 = a_1 + a_0 + \boldsymbol{x}^T\boldsymbol{a}$*,* $a'_2 = a_2 + b_0 + \boldsymbol{x}^T\boldsymbol{b}$*,* $a'_3 = a_3 + c_0 + \boldsymbol{x}^T\boldsymbol{c}$*. It accepts if* $a'_1 a'_2 - a'_3 - a_4 z = 0$

B.6 Vector Encryption

Definition 12 (Linear-Only Vector Encryption (adapted from [17])**.** *Let* $\mathbb{F}$ *be a finite field. A secret-key additively-homomorphic vector encryption scheme over a vector space* $\mathbb{F}^\ell$ *consists of a tuple of algorithms* $\Pi_{\mathsf{Encrypt}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add})$ *which are defined as follows:*

- $(\mathsf{pp}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$*: On input the security parameter* λ *and the plaintext dimension* ℓ*, the setup algorithm outputs public parameters* pp *and a secret key* sk*.*
- $\boldsymbol{C} \leftarrow \mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v})$*: On input the secret key* sk *and a vector* $\boldsymbol{v} \in \mathbb{F}^\ell$*, the encryption algorithm outputs ciphertext* $\boldsymbol{C}$*.*
- $\boldsymbol{v}/\bot \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C})$*: On input the secret key* sk *and a ciphertext* $\boldsymbol{C}$*, the decryption algorithm either outputs a vector* $\boldsymbol{v} \in \mathbb{F}^\ell$ *or a special symbol* $\bot$*.*
- $\boldsymbol{C}^* \leftarrow \mathsf{Add}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}, \{y_i\}_{y\in[m]})$*: On input the public parameters, a collection of ciphertexts* $\{\boldsymbol{C}_i\}_{i\in[m]}$ *and scalars* $\{y_i\} \in \mathbb{F}$*,* $i \in [m]$*, the addition algorithm outputs a new ciphertext* $\boldsymbol{c}^*$*.*

The linear-only vector encryption satisfies the standard properties of CPA security and additive homomorphism (see full version). The circuit privacy peroperty is defined as follows.

Additive homomorphism. For all security parameters $\lambda \in \mathbb{N}$, vectors $\{\boldsymbol{v}_i\}_{i\in[m]} \subseteq \mathbb{F}^\ell$, and scalars $\{y_i\}_{i\in[m]} \subseteq \mathbb{F}$ where $m = m(\lambda)$,

$$\Pr[\sum_{i\in[m]} y_i \boldsymbol{v}_i \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C}^*)] = 1 - \mathsf{negl}(\lambda), \tag{19}$$

where $(\mathsf{pp}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$, $\boldsymbol{C}_i \leftarrow \mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v}_i)$ for all $i \in [m]$ and $\boldsymbol{c}^* \leftarrow \mathsf{Add}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}, \{y_i\}_{i\in[m]})$. We say that the linear-only vector encryption is additively homomorphic with respect to a set $S \subseteq R_p^m$ if (19) holds for all $(y_1, \ldots, y_m) \in S$.

Note: The additive homomorphism implies the correctness of the decryption.

Statistical Circuit Privacy. Let $\Pi_{\mathsf{Enc}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add})$ be a secret-key vector encryption scheme over $\mathbb{F}^\ell$. Π_{Enc} is circuit private if for all stateful adversaries $\mathcal{A}$, there exists an efficient simulator $\mathcal{S}$, such that for all security parameters $\lambda \in \mathbb{N}$

$$\Pr[\mathsf{Game}^{\mathsf{circ-priv}}_{\Pi_{\mathsf{Enc}},\mathcal{A},\mathcal{S}}(1^\lambda) = 1] = 1/2 + \mathsf{negl}(\lambda),$$

where $\mathsf{Game}^{\mathsf{circ-priv}}_{\Pi_{\mathsf{Enc}},\mathcal{A},\mathcal{S}}(1^\lambda)$ is defined as follows:

(1) The challenger lets $(pp, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives (pp, sk) to $\mathcal{A}$. $\mathcal{A}$ replies with a collection of vectors $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) \in \mathbb{F}^m$.
(2) The challenger constructs $C_i \leftarrow \mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v}_i)$ for all $i \in [m]$ and gives $\{C_i\}_{i\in[m]}$ to $\mathcal{A}$. The adversary replies with a collection of $\mathbb{F}$ coefficients $\{y_i\}_{i\in[m]}$.
(3) The challenger computes $c_0^* \leftarrow \mathsf{Add}(pp, \{C_i\}_{i\in[m]}, \{y_i\}_{i\in[m]})$ and $c_1^* \leftarrow \mathcal{S}(1^\lambda, pp, \mathsf{sk}, \sum_{i\in[m]} y_i \boldsymbol{v}_i)$. It samples $b \hookleftarrow \{0, 1\}$ and replies to $\mathcal{A}$ with c_b^*.
(4) The adversary outputs a bit $b' \in \{0, 1\}$. The output of the Game is 1 if $b = b'$ and 0 otherwise.

We adopt the following definition of 'linear only' encryption. As opposed to the 'strictly linear only' definition in [47], the definition we use, called 'strictly linear targeted malleability' does not require the existence of a knowledge extractor. It suffices for constructing SNARGs. The 'strict' aspect strengthens the linear targeted malleability definition in [17] by adding the strict linearity requirement, i.e. affine homomorphic operations are also disallowed. It is the simulation-based linear targeted malleability analogue of the knowledge-based strict linear-only requirement in [47]. We remark that, as also observed in [17, 47], the strict linear only can be relaxed to affine if the underlying LPCP soundness notion is strengthened from linear provers to affine provers.

Definition 13 (Strictly linear targeted malleability property (adapted from [16],[17])**.** *A vector encryption scheme* $\Pi_{\mathsf{Encrypt}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add})$ *over* $\mathbb{F}^\ell$ *satisfies* strictly linear targeted malleability *if for all efficient adversaries* $\mathcal{A}$ *and plaintext generators* $\mathcal{M}$*, there is a (possibly computationally unbounded) simulator* $\mathcal{S}$ *such that for all security parameters* $\lambda \in \mathbb{N}$*, auxiliary input* $z \in \{0, 1\}^{\mathsf{poly}(\lambda)}$*, and any efficient distinguisher* $\mathcal{D}$*, we have*

$$\Pr[\mathsf{ExptSLTM}_{\Pi_{\mathsf{Encrypt}},\mathcal{A},\mathcal{M},\mathcal{S},\mathcal{D},z}(1^\lambda, 1^N) = 1] \le 1/2 + \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{ExptSLTM}_{\Pi_{\mathsf{Encrypt}},\mathcal{A},\mathcal{M},\mathcal{S},z}(1^\lambda, 1^N)$ *is defined as follows:*

(1) The challenger samples $(s, \{\boldsymbol{v}_i\}_{i\in[m]}) \leftarrow \mathcal{M}(1^\ell)$*.*
(2) The challenger proceeds as follows:
  - *If* $b = 0$ *('REAL' distribution):*

  - $(\mathsf{pp}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$
  - $C_i \leftarrow \mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v}_i)$ *for each* $i \in [m]$
  - $C' \leftarrow \mathcal{A}(\{C_i\}_{i\in[m]}; z)$, *where* $\boldsymbol{v}'_0 := \mathsf{Decrypt}(\mathsf{sk}, C') \neq \perp$.
  - $\boldsymbol{v}'_0 := \mathsf{Decrypt}(\mathsf{sk}, C')$.
- *If* $b = 1$ *('IDEAL' distribution):*
  - $\boldsymbol{\pi} \leftarrow \mathcal{S}(z)$, *where* $\boldsymbol{\pi} \in \mathbb{F}^\ell$.
  - $\boldsymbol{v}'_1 := [\boldsymbol{v}_1|\boldsymbol{v}_2|\cdots|\boldsymbol{v}_m] \cdot \boldsymbol{\pi}$.

(3) *The challenger runs* $b' \leftarrow \mathcal{D}(\{\boldsymbol{v}_i\}_{i\in[m]}, s, \boldsymbol{v}'_b)$.

(4) *The output of the experiment is* 1 *if* $b' = b$ *and is* 0 *otherwise.*

If the simulator $\mathcal{S}$ *is computationally efficient, then we say that the* Π_{Encrypt} *satisfies strict linear targeted malleability with computationally efficient simulation.*

C FURTHER DISCUSSION ON PRIOR AND CONCURRENT WORK ON LHLL AND RE-RANDOMIZATION

Prior work on lattice discrete Gaussian smoothing-based regularity bounds has studied the distribution of $\boldsymbol{x}^T\boldsymbol{A}$ (e.g. [55, 64, 65]) or $\boldsymbol{x}^T\boldsymbol{E}$ (e.g. [1, 2, 49]). The work [52] analyzes the distribution of $\boldsymbol{x}$ conditioned on $(\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{E})$ in unstructured case [30] analyzes special conditional distributions over rings that are different from those we analyze. Two recent concurrent and independent works to ours [22, 50] introduce a related LHLL variant of the re-randomization result from [23], applied in the context of circuit privacy for the GSW or TFHE Fully Homomorphic Encryption scheme (rather than our 'Half-GSW' ZK-SNARG context). The concurrent independent work [22] gives a polynomial ring variant of [23] re-randomization but for R_q with a non-standard power-of-2 LWE modulus $q = 2^k$ (rather than our prime or 'almost prime' modulus $q = p\bar{q}$ for a large prime $\bar{q}$ and small prime p). The proof approach in [22] also differs to ours; it adapts a variant of the entropy-based leftover hash lemma used in [23] to R_q using an adaptation of the collision-probability arguments of [3, 56], combined with an upper bound on the smoothing parameter of an intersection of the gadget perp lattice and ideals of the ring. The latter is similar to our smoothing bounds, but differs in that we use a lattice smoothing argument over the whole module lattices and study the smoothing parameters of intersections of the module lattices. The second concurrent independent work [50] focuses on generalizing the result of [23] to work over $\mathbb{Z}_q$ with general composite q, but not to R_q.

D ATTACK ON ZERO-KNOWLEDGE PROPERTY OF ISW CONSTRUCTION [47] WITH NO SMUDGING

In this Section, we present an attack on the zero-knowledge property of the variant of the ZK-SNARG in [47] which uses no smudging. The attack applies only to this "no-smudging" variant (introduced in Remark 3.24, Def. 3.25, Le. 3.26 of [47]) and does not invalidate the results of [47] in general. For definitions of HVZK with leakage for the linear PCP and other related syntax, please refer to Appendix B.4.

Description of attack. Let κ denote the zero-knowledge parameter as it is used in [47]. For $\kappa = 0$ and d a power of 2, we present an attack against the δ-HVZK with $(\mathcal{D}, q)$ leakage property defined in Definition 7 for the linear PCP Π_{LPCP} with query length $t := 4 + N_g + N_w - n$ for R1CS family $\mathcal{CS}_N$ over $R_p := \mathbb{Z}_p[x]/(x^d+1)$ with p prime, as used in the lattice-based ZK-SNARG instantiation in [47]. Here, n, N_g and N_w denote the witness size, number of constraints and number of variables for $\mathcal{CS}_N$, respectively, and $\mathcal{D}$ is the distribution over matrices over $R := \mathbb{Z}[x]/(x^d+1)$ defined as follows (see Lemma 3.26 in [47]):

- Sample $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{t\times n})$ and $\boldsymbol{E} \hookleftarrow \mathcal{D}_{R,s}^{t\times\ell'}$.
- Output the matrix $\boldsymbol{Z} = [\boldsymbol{A}, \boldsymbol{E}] \in R^{t\times(n+\ell')}$.

For the attack to break δ-HVZK with $(\mathcal{D}, q)$ leakage property of the linear PCP Π_{LPCP}, it suffices to exhibit an R1CS family $\mathcal{CS}_N$ over R_p and a statement $\boldsymbol{x}$ and two distinct witnesses $\boldsymbol{w}, \boldsymbol{w}'$ such that:

- $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = \mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}') = 1$, but
- the statistical distance Δ between the distribution of $(\boldsymbol{Z}, \boldsymbol{b})$ and $(\boldsymbol{Z}, \boldsymbol{b}')$ is greater than 2δ, where $\boldsymbol{Z} \hookleftarrow \mathcal{D}$, $\boldsymbol{b} = \boldsymbol{Z}^T \cdot \boldsymbol{\pi} \in R_q^n$, $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(\boldsymbol{x}, \boldsymbol{w})$, and $\boldsymbol{b}' = \boldsymbol{Z}^T \cdot \boldsymbol{\pi}' \in R_q^n$, $\boldsymbol{\pi}' \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(\boldsymbol{x}, \boldsymbol{w}')$.

We focus here on the bottom ℓ' coordinates $\tilde{\boldsymbol{b}} = \boldsymbol{E}^T\boldsymbol{\pi}, \tilde{\boldsymbol{b}}' = \boldsymbol{E}^T\boldsymbol{\pi}'$ of $\boldsymbol{b}, \boldsymbol{b}'$ respectively. Due to the correctness of decryption, the parameters in [47] are chosen such that $\|\tilde{\boldsymbol{b}}\|_\infty, \|\tilde{\boldsymbol{b}}'\|_\infty < q/2$, so the computation is over R (without any modulus reduction). By construction of the underlying linear PCP (see Def. 11 in App. B.4),$\boldsymbol{\pi}$ has the form $\boldsymbol{\pi}^T = (\delta_1, \delta_2, \delta_3, \bar{\boldsymbol{w}}^T, \boldsymbol{h}^T) \in R_p^{4+N_g+N_w-n}$, where $(\delta_1, \delta_2, \delta_3)$ is uniformly random in R_p^3, $\bar{\boldsymbol{w}} \in R_p^{N_w-n}$ is the R1CS witness (excluding the statement $\boldsymbol{x} \in R_p^n$, i.e. $\boldsymbol{w}^T = (\boldsymbol{x}^T, \bar{\boldsymbol{w}}^T)$), and $\boldsymbol{h} \in R_p^{N_g+1}$ is the coefficient vector of the polynomial $H(X) := \frac{A(X)B(X)-C(X)}{Z_S(X)}$ constructed from $\boldsymbol{x}, \bar{\boldsymbol{w}}, \delta_1, \delta_2, \delta_3$ (see Def. 11). For our attack analysis, we will make the heuristic assumption that $\boldsymbol{r}^T := (\delta_1, \delta_2, \delta_3, \boldsymbol{h}^T)$ behaves as a (pseudo) uniformly random vector in $R_p^{N_g+4}$ independent of $\bar{\boldsymbol{w}}$ (respectively, $(\boldsymbol{r}')^T := (\delta'_1, \delta'_2, \delta'_3, (\boldsymbol{h}')^T)$ is uniform and independent of $\bar{\boldsymbol{w}}'$ for $\boldsymbol{\pi}'$).

Then, $\tilde{\boldsymbol{b}} = \boldsymbol{E}_1^T\boldsymbol{r} + \boldsymbol{E}_2^T\bar{\boldsymbol{w}}$, where the first term is a random 'error' term, while the second term is a fixed 'shift' (respectively, $\tilde{\boldsymbol{b}}' = \boldsymbol{E}_1^T\boldsymbol{r}' + \boldsymbol{E}_2^T\bar{\boldsymbol{w}}'$), and $\boldsymbol{E}_1^T$ (resp. $\boldsymbol{E}_2^T$) consists of the columns of $\boldsymbol{E}^T$ that multiply $\boldsymbol{r}$ (resp. $\bar{\boldsymbol{w}}$). Now, each integer coefficient of the 'error' term $\boldsymbol{E}_1^T\boldsymbol{r}$ (resp. $\boldsymbol{E}_1^T\boldsymbol{r}'$) is an inner-product between $\boldsymbol{r}$ with coefficients (pseudo) uniformly random in $(-p/2, p/2)$ and hence of standard deviation $p/\sqrt{12}$, and a column of $\boldsymbol{E}_1$ whose expected norm is $\approx s\sqrt{(N_g+4)/(2\pi)}$. Therefore, we heuristically expect the distribution of the error term coordinates (conditioned on $\boldsymbol{E}_1$) to be approximately a discrete Gaussian with standard deviation $\sigma_b := sp\sqrt{(N_g+4)/(24\pi)}$ for both $\tilde{\boldsymbol{b}}$ and $\tilde{\boldsymbol{b}}'$. Let $(\boldsymbol{e}_2^{(i)})^T$ denote the i'th row of $\boldsymbol{E}_2^T$. For $\boldsymbol{b} \in \{\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{b}}'\}$, conditioned on $\boldsymbol{E}$, the distribution of $y^{(i)} := b^{(i)} - (\boldsymbol{e}_2^{(i)})^T\bar{\boldsymbol{w}}$ should therefore be approximately either $\mathcal{D}_{\mathbb{Z},\sqrt{2\pi}\sigma_b}$ (if $\boldsymbol{b} = \tilde{\boldsymbol{b}}$) or $\mathcal{D}_{\mathbb{Z},\sqrt{2\pi}\sigma_b} + c_b^{(i)}$. (if $\boldsymbol{b} = \tilde{\boldsymbol{b}}'$), with centre $c_b^{(i)} := (\boldsymbol{e}_2^{(i)})^T \cdot (\bar{\boldsymbol{w}}' - \bar{\boldsymbol{w}})$. These distributions can be distinguished with high advantage if $\|c_b^{(i)}\|/\sigma_b$ is larger than a constant.

Concrete Analysis of Distinguishing Advantage. The i'th coordinate distinguisher described above achieves a distinguishing advantage approximately equal to the statistical distance between the corresponding continuous Gaussian distributions namely $\Delta^{(i)} \approx$

$2\Phi\left(\frac{|c_b^{(i)}|}{2\sigma_b}\right) - 1 = \mathrm{erf}\left(\frac{|c_b^{(i)}|}{2\sqrt{2}\sigma_b}\right)$, where Φ and erf respectively denote the cumulative distribution function and standard error function for a continuous Gaussian distribution with mean 0 and unit standard deviation. Since $(\boldsymbol{e}_2^{(i)})^T$ has coordinates with standard deviation $s/\sqrt{2\pi}$, the distribution of c_b (wrt the choice of $\boldsymbol{e}_2^{(i)}$) is approximately $\mathcal{D}_{\mathbb{Z},s\|\bar{\boldsymbol{w}}'-\bar{\boldsymbol{w}}\|}$. Therefore, by a continuous Gaussian approximation, the expected value of $|c_b^{(i)}|$ is $\bar{c}_b^{(i)} \approx \frac{s}{\pi}\|\bar{\boldsymbol{w}}' - \bar{\boldsymbol{w}}\|$, and hence we expect to get

$$\Delta^{(i)} \approx \mathrm{erf}\left(\frac{\bar{c}_b^{(i)}}{2\sqrt{2}\sigma_b}\right) = \mathrm{erf}\left(\frac{\sqrt{3}\|\bar{\boldsymbol{w}}' - \bar{\boldsymbol{w}}\|}{\sqrt{\pi}p\sqrt{N_g + 4}}\right). \tag{20}$$

Example. We now give an example R1CS relation for which our attack has a non-negligible distinguishing advantage. For a prime p and a positive integer N, we define the following relation:

$$C(x \in \mathbb{Z}_p, \boldsymbol{w} \in \mathbb{F}_p^N) = 1 \iff \vee_{i\in[N]}(w_i = x) = 1.$$

Let $P_{\boldsymbol{w}}(z) := \prod_{i\in[N]}(z - w_i) \in \mathbb{F}_p^{<N}$. Note that $P_{\boldsymbol{w}}(z)$ is a polynomial in z of degree N with coefficients over $\mathbb{F}_p$ such that $P_{\boldsymbol{w}}(w_j) = 0$ for all $j \in [N]$. Now let us define $A(x, \boldsymbol{w}) := 1 - P_{\boldsymbol{w}}(x)$. It is clear that $A(x, \boldsymbol{w}) = 1$ iff $P_{\boldsymbol{w}}(x) = 0$ iff $C(x, \boldsymbol{w}) = 1$, so A computes C, as required. Now consider the following two valid witnesses for the statement $x \in \mathbb{F}_p \setminus \{0\}$: $\boldsymbol{w} = (x, 0, \ldots, 0) \in \mathbb{F}_p^N$ and $\boldsymbol{w}' = (x, x, \ldots, x) \in \mathbb{F}_p^N$. It is easy to see that $\|\boldsymbol{w} - \boldsymbol{w}'\| = x\sqrt{N-1}$. We convert the natural arithmetic circuit for A (consisting of $N+1$ input wires, $N+1$ addition gates with weighted inputs, $N-1$ multiplication gates, and one output wire) into an R1CS relation $\mathcal{CS}_N$ following the method outlined in Sec. 7.4 of [39]. This gives a number N_g of R1CS constraints equal to the sum of the number of internal multiplication gates plus 1 for the circuit output wire (i.e. a total of $N_g = N$ constraints in our case), and the corresponding R1CS witness vector $\bar{\boldsymbol{w}}_{\mathrm{R1CS}}^T$ has the form $(\bar{\boldsymbol{w}}^T, \boldsymbol{o}_M^T, o_C)$, where $\boldsymbol{o}_M^T$ denotes the vector of internal multiplication gate outputs and o_C denotes the circuit output value. Hence, in our case, the R1CS witness vectors corresponding to $\bar{\boldsymbol{w}}$ and $\bar{\boldsymbol{w}}'$ are $\bar{\boldsymbol{w}}_{\mathrm{R1CS}}^T = ((x, 0, \ldots, 0), (0, 0, \ldots, 0), 1)$ and $(\bar{\boldsymbol{w}}'_{\mathrm{R1CS}})^T = ((x, x, \ldots, x), (0, 0, \ldots, 0), 1)$ and hence $\|\bar{\boldsymbol{w}}_{\mathrm{R1CS}} - \bar{\boldsymbol{w}}'_{\mathrm{R1CS}}\| = x\sqrt{N-1}$. Plugging in (20) with $N_g := N \to \infty$ and $x \geq p/c$ for some $c = O(1)$, one can see that the expected distinguishing advantage $\Delta^{(i)} \approx \mathrm{erf}\left(\frac{\sqrt{3}}{\sqrt{\pi}c}\right) = \Omega(1)$ is non-negligible.

E ADDITIONAL PROOFS

E.1 Proof of Lemma 1

Proof. Let $\bar{\boldsymbol{x}}^T := (\boldsymbol{x}^T, \boldsymbol{y}^T) \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(G)+\boldsymbol{c},r} \times \mathcal{D}_{R^{\ell'},r} := \mathcal{D}_{\Lambda_q^\perp(\bar{G})+\bar{\boldsymbol{c}},r}$, where $\bar{\boldsymbol{G}} := \begin{pmatrix} \boldsymbol{G} \\ \boldsymbol{0}_{\ell'\times\nu} \end{pmatrix} \in R^{(L+\ell')\times\nu}$ and $\bar{\boldsymbol{c}} := \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{0}_{\ell'} \end{pmatrix} \in R^{L+\ell'}$. We first observe that thanks to (1), we have $\|\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}\|_\infty < q/2$ (i.e. no wraparound mod q in $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}$ mod q) except with probability $\leq 4\epsilon$. Indeed, we have $\|\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}\|_\infty \leq \|\boldsymbol{x}^T\boldsymbol{E}\|_\infty + \|\boldsymbol{y}\|_\infty$. Now, by Lemma 7, we have $\|\boldsymbol{x}\| \leq r\sqrt{Ld}$ except with probability $\leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-Ld} \leq 3\epsilon$ using $\epsilon \leq 1/2$ and the choice of r in (8), where we have used the fact that $\eta_\epsilon(\Lambda_q^\perp(\boldsymbol{G})) \leq \beta^2\sqrt{\ln(2Ld(1+\epsilon^{-1}))/\pi}$ by Lemma 11 and Lemma 3 below, and that $c \geq c_2 \geq \beta^2$. Therefore, by Lemma 8, a fixed integer coefficient of $\boldsymbol{x}^T\boldsymbol{E}$ has absolute value $\leq \sqrt{\ln(2/\epsilon')/\pi}s\|\boldsymbol{x}\| \leq \sqrt{\ln(2/\epsilon')/\pi}sr\sqrt{Ld}$ except with probability $\leq \epsilon'$ and therefore, by a union bound over the $\ell' d$ integer coordinates of $\boldsymbol{x}^T\boldsymbol{E}$, setting $\epsilon' := \epsilon/(\ell' d)$, we conclude that $\|\boldsymbol{x}^T\boldsymbol{E}\|_\infty \leq \sqrt{\ln(2\ell' d/\epsilon)/\pi}sr\sqrt{Ld}$, except with probability $\leq 4\epsilon$.

Similarly using Lemma 2 and the union bound, we have $\|\boldsymbol{y}\|_\infty \leq \sqrt{\ln(2\ell' d/\epsilon)/\pi}r$, except with probability $\leq \epsilon$. Overall, we have $\|\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}\|_\infty < r(s\sqrt{Ld}+1)\sqrt{\ln(2\ell' d/\epsilon)/\pi} < q/2$, except with probability $\leq 5\epsilon$, where the last inequality is due to the second part of (7). The claimed bound of the Lemma, therefore, follows if we show that

$$\Delta\left((\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}, \boldsymbol{A}, \boldsymbol{E}), (\mathcal{U}(R_q^n), \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}, \boldsymbol{A}, \boldsymbol{E})\right) \leq 11\epsilon.$$

(i.e. with no mod q reduction on $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}$). To show that latter bound we use a smoothing-based approach. Namely, it is enough to show that, except with negligible probability $\leq 3\epsilon$ over the choice of $\boldsymbol{A}, \boldsymbol{E}$, the conditional distribution of $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}}$ mod q conditioned on $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}$ over the choice of $\bar{\boldsymbol{x}}^T$ is within neg. statistical distance $\leq 8\epsilon$ to $\mathcal{U}(R_q^n)$. For fixed $\bar{\boldsymbol{A}}, \bar{\boldsymbol{E}}$ and $\hat{\boldsymbol{e}}$ and $\hat{\boldsymbol{v}}$, let $P_{\hat{\boldsymbol{e}}}(\hat{\boldsymbol{v}}) := \Pr_{\bar{\boldsymbol{x}}}[\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q = \hat{\boldsymbol{v}}^T | \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}} \bmod q = \hat{\boldsymbol{e}}^T]$. Then, for $\hat{\boldsymbol{v}}$ in the support of $P_{\hat{\boldsymbol{e}}}$, and $\hat{\boldsymbol{e}}^T$ in the support of $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}$, there exists $\bar{\boldsymbol{x}}_0^T \in \Lambda_q^\perp(\bar{\boldsymbol{G}}) + \bar{\boldsymbol{c}}^T$ such that $\bar{\boldsymbol{x}}_0^T \cdot (\bar{\boldsymbol{A}}, \bar{\boldsymbol{E}}) = (\hat{\boldsymbol{v}}^T, \hat{\boldsymbol{e}}^T) \bmod q$. Then:

$$P_{\hat{\boldsymbol{e}}}(\hat{\boldsymbol{v}}) = \frac{\Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\bar{G})+\bar{\boldsymbol{c}}^T,r}}[\bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{E}}) = (\hat{\boldsymbol{v}}^T, \hat{\boldsymbol{e}}^T)]}{\Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\bar{G})+\bar{\boldsymbol{c}}^T,r}}[\bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{E}} = \hat{\boldsymbol{e}}^T]}. \tag{21}$$

The numerator p_n of (21) has the form

$$p_n := \Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow D_{\Lambda_q^\perp(\bar{G})+\bar{\boldsymbol{c}}^T,r}}[\bar{\boldsymbol{x}}^T \in (\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}_0}^T + \Lambda^\perp(\bar{\boldsymbol{E}}))] \tag{22}$$

$$= \frac{\rho_r((\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda^\perp(\bar{\boldsymbol{E}})) \cap (\bar{\boldsymbol{c}} + \Lambda_q^\perp(\bar{\boldsymbol{G}})))}{\rho_r(\bar{\boldsymbol{c}} + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{23}$$

$$= \frac{\rho_r((\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda^\perp(\bar{\boldsymbol{E}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}})))}{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{24}$$

$$= \frac{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}}))}{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{25}$$

$$\in (1 \pm 4\epsilon) \cdot \frac{\rho_r(\Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}}))}{\rho_r(\Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{26}$$

where the second equality uses the fact that both $\bar{\boldsymbol{c}}^T$ and $\bar{\boldsymbol{x}}_0^T$ are in same coset of $\Lambda_q^\perp(\bar{\boldsymbol{G}})$, and the last equation uses smoothing Lemma 9 twice, if the smoothing condition $r \geq \eta'$ holds, where $\eta' := \eta_\epsilon(\Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}}))$ is the smoothing parameter of the intersection of the lattices $\Lambda_q^\perp(\bar{\boldsymbol{A}})$, $\Lambda^\perp(\bar{\boldsymbol{E}})$ and $\Lambda_q^\perp(\bar{\boldsymbol{G}})$. Note that the orthogonality relation defining the lattice $\Lambda^\perp(\bar{\boldsymbol{E}}) := \{\boldsymbol{v} \in R^{L+\ell'} : \boldsymbol{v}^T \cdot \bar{\boldsymbol{E}} = 0\}$ is over R, not just R_q. Let $\bar{\Lambda}' := \Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}})$. Now, notice that due to the $\ell' \times \ell'$ identity matrix at the bottom ℓ' rows of $\bar{E}$, the rank of lattice $\bar{\Lambda}'$ and $\Lambda^\perp(\bar{\boldsymbol{E}})$ over $\mathbb{R}$ is Ld (rather than the rank $(L+\ell')d$ of $\Lambda_q^\perp(\bar{\boldsymbol{G}})$). To upper bound η', by Lemma 11, it suffices to get an upper bound on $\lambda_{Ld}(\bar{\Lambda}')$, namely $\eta' \leq \sqrt{\ln(2Ld(1+\epsilon^{-1}))/\pi} \cdot \lambda_{Ld}(\bar{\Lambda}')$. To upper bound $\lambda_{Ld}(\bar{\Lambda}')$, we use a transference bound argument which can be viewed as a generalization of the bound on $\eta_\epsilon(\Lambda^\perp(\boldsymbol{E}))$ in Corollary 3 of [2]. The idea is to do it in two steps, where the first step involves finding an upper bound on the Ld'th minimum of the q-ary lattice

$\bar{\Lambda} := \Lambda_q^{\perp}(\bar{\boldsymbol{A}}) \cap \Lambda_q^{\perp}(\bar{\boldsymbol{E}}) \cap \Lambda_q^{\perp}(\bar{\boldsymbol{G}})$ and then the second step is to show that this bound also applies to the *non* q-ary lattice $\bar{\Lambda}'$, as follows:

- **Step 1**: Use the transference bound in Lemma 10 to transform the problem of upper bounding $\lambda_{Ld}(\bar{\Lambda})$ to the problem of lower bounding the $(\ell' d + 1)$'th minimum $\lambda_{\ell' d+1}(\bar{\Lambda}^*)$ of the dual lattice $\bar{\Lambda}^*$:

$$\lambda_{Ld}(\bar{\Lambda}) \leq \frac{(L+\ell')d}{\lambda_{\ell' d+1}(\bar{\Lambda}^*)} = \frac{(L+\ell')d}{\frac{1}{q}\lambda_{\ell' d+1}(\Lambda_q(\bar{\boldsymbol{M}}))},$$

  where $\bar{\boldsymbol{M}} := (\bar{\boldsymbol{A}}, \bar{\boldsymbol{E}}, \bar{\boldsymbol{G}}) \in R_q^{(L+\ell')\times(n+\ell'+\nu)}$. In this step, we give a lower bound $\lambda_{\ell' d+1}(\Lambda_q(\bar{\boldsymbol{M}}))$ of the form q/c for some 'small' c. We first observe that $\lambda_{\ell' d+1}(\Lambda_q(\bar{\boldsymbol{M}}))$ is lower bounded by the norm of the shortest vector $\boldsymbol{w}$ in the lattice $\Lambda_q(\bar{\boldsymbol{M}})$ excluding those lattice vectors in the integer column span of $\text{rot}(\bar{\boldsymbol{E}})$; therefore it suffices to lower bound the latter minimum norm which we denote by $\lambda_1(\Lambda_q(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}))$. This is because, if $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{\ell' d+1}$ are $\ell' d + 1$ linearly independent vectors in $\Lambda_q(\bar{\boldsymbol{M}})$ all of norm at most $\lambda_{\ell' d+1}$, one of them must not be in the span of $\bar{\boldsymbol{E}}$ since the latter has dimension less than $\ell' d$, so that vector has norm $\lambda_{\ell' d+1} \geq \lambda_1(\Lambda_q(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}))$. Next, to lower bound $\lambda_1(\Lambda_q(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}))$, we proceed as follows. First, to simplify the following analysis, we focus on the prime modulus ring $R_{\bar{q}}$, using the observation that, since $\bar{q}$ divides q, we have that $\Lambda_q(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d})$ is a subset of $\lambda_1(\Lambda_{\bar{q}}(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}))$, and hence $\lambda_1(\Lambda_q(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d})) \geq \lambda_1(\Lambda_{\bar{q}}(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}))$. Now, for any vector $\boldsymbol{w}$ in $\Lambda_{\bar{q}}(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d})$, write $\boldsymbol{w} = \bar{\boldsymbol{A}}\boldsymbol{v}_A + \bar{\boldsymbol{E}}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G \bmod q$. We can divide this problem into three sub cases, whose results we summarise below and provide the detailed analysis of subcases 1 and 2 in Lemma 2, Lemma 13 in the following pages. Namely, we show that $\|\boldsymbol{w}\| \geq \bar{q}/c$ with $c := \max(c_1, c_2, c_3)$ for some 'small' c_1, c_2, c_3:
  - *Subcase 1* ($\boldsymbol{v}_A \neq 0 \bmod \bar{q}$, Lemma 2): Here, we use a probabilistic approach to lower bound $\|\boldsymbol{w}\|$ over the randomness of $\boldsymbol{A}$ and using a union bound over $\boldsymbol{v}_E, \boldsymbol{v}_G$ by extending the approach from [59, 65] for lower bounding the minimum of Module SIS lattices.

    In particular, Lemma 2 shows that $\|\boldsymbol{w}\| \geq \bar{q}/c_1$ for some 'small' c_1, except with negligible probability $p_1(c_1) \leq \epsilon$, for the assumed choice of parameters.
  - *Subcase 2* ($\boldsymbol{v}_A = 0 \bmod \bar{q}$ and $\|\boldsymbol{v}_E\| \leq \bar{q}/c_3$ for a 'small' c_3, Lemma 4): Here, if $\boldsymbol{v}_G = \boldsymbol{0}$, the smallness of $\|\boldsymbol{v}_E\|$ and $\|\boldsymbol{E}\|$ implies that $\boldsymbol{w} = \bar{\boldsymbol{E}}\boldsymbol{v}_E$ is $< \bar{q}/2$ over the integers and hence does not wrap around mod $\bar{q}$ and is in $\Lambda_{\bar{q}}(\bar{\boldsymbol{M}} \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d})$) with negligible probability over the choice of $\boldsymbol{E}$. On the other hand, if $\boldsymbol{v}_G \neq \boldsymbol{0}$, the length of $\boldsymbol{w}$ is lower bounded up to a 'small' additive norm $\|\bar{\boldsymbol{E}}\boldsymbol{v}_E\|$ from the minimum of the Gadget lattice $\Lambda_{\bar{q}}(\boldsymbol{G})$, which we show in turn (in Lemma 3) is lower bounded by $\bar{q}/(2\beta)$. The above arguments are made precise in Lemma 4, which shows that in this subcase, we get $\|\boldsymbol{w}\| \geq \bar{q}/c_2$ for a 'small' $c_2 = 4\beta$, except with negligible probability $Ld2^{-d\ell'} \leq \epsilon$, for the assumed choice of parameters.
  - *Subcase 3* ($\boldsymbol{v}_A = 0 \bmod \bar{q}$ and $\|\boldsymbol{v}_E\| > \bar{q}/c_3$): Here, we use the observation that $\|\boldsymbol{w}\| \geq \|\boldsymbol{v}_E\| > \bar{q}/c_3$, since the bottom $\ell' d$ $\mathbb{Z}$ coordinates of $\boldsymbol{w}$ consists of $\boldsymbol{v}_E$, thanks to the identity matrix in the bottom rows of $\bar{\boldsymbol{E}}$.
- **Step 2**: We observe that any Ld $\mathbb{R}$-linearly independent vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{Ld}$ in $\bar{\Lambda}$ all of norm $\leq \lambda_{Ld}(\bar{\Lambda})$ are also in $\bar{\Lambda}'$ (i.e. orthogonal to $\bar{\boldsymbol{E}}$ over R and not just R_q), thanks to short norm of those vectors and the shortness of $\bar{\boldsymbol{E}}$ compared to q, except with negligible probability $\leq \epsilon$ over the choice of $\boldsymbol{E}$. This shows that $\lambda_{Ld}(\bar{\Lambda}') \leq \lambda_{Ld}(\bar{\Lambda})$ except with negligible probability $\leq \epsilon$ over choice of $\boldsymbol{E}$. Indeed, by Cauchy-Schwartz inequality, we have $\|\boldsymbol{w}_i^T \cdot \bar{\boldsymbol{E}}\| \leq \|\boldsymbol{w}_i\| \cdot \|\bar{\boldsymbol{E}}^T\|$. Using the Step 1 bound we have $\|\boldsymbol{w}_i\| \leq (L+\ell')dcp$ and using Lemma 6 and a union bound over the $\ell' d$ columns of $\boldsymbol{E}$, we have the bound $\|\bar{\boldsymbol{E}}^T\|^2 \leq 1 + (2s\sqrt{Ld}/\sqrt{2\pi})^2$ except with probability $\leq \ell' d 2^{-Ld} \leq \epsilon$ over the choice of $\boldsymbol{E}$. According to (7), we get $\|\boldsymbol{w}_i^T \cdot \bar{\boldsymbol{E}}\| < (L+\ell')dcp\sqrt{1 + (2s\sqrt{Ld}/\sqrt{2\pi})^2} < q/2$ and therefore $\lambda_{Ld}(\bar{\Lambda}') \leq \lambda_{Ld}(\bar{\Lambda})$, except with negligible probability $\leq \epsilon$ as required.

Putting together Steps 1 and Steps 2, we get the upper bound $\lambda_{Ld}(\bar{\Lambda}') \leq \lambda_{Ld}(\bar{\Lambda}) \leq (L+\ell')dpc$ and hence the smoothing parameter bound $\eta' \leq (L+\ell')dpc\sqrt{\ln(2Ld(1+\epsilon^{-1}))/\pi}$, except with probability $\leq 3\epsilon$ over the choice of $\boldsymbol{A}, \boldsymbol{E}$. The assumed bound on r therefore implies that $r \geq \eta'$ except with probability $\leq 3\epsilon$ and hence from (26), we conclude that except with negligible probability $\leq 3\epsilon$ over the choice of $\boldsymbol{A}, \boldsymbol{E}$, the conditional distribution of $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q$ conditioned on $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}$ over the choice of $\bar{\boldsymbol{x}}^T$ is within neg. statistical distance 8ϵ to $\mathcal{U}(R_q^n)$, as required. □

E.2 Proof of Lemma 2

Proof. We use a union bound argument similar to that used in Lemma 3.2 of [66] (see also [59]) to lower bound the minimum of random q-ary module lattices. For $\beta \in \mathbb{R}$, we denote by $S_{2,\beta}$ the set of elements of R of Euclidean norm less than β, i.e. $S_{2,\beta} := \{\boldsymbol{w} \in R : \|\boldsymbol{w}\|_2 < \beta\}$. By a union bound, we have:

$$p_1 \leq \sum_{\substack{(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) \\ \in R_{\bar{q}}^n \setminus \boldsymbol{0} \times R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^{\nu} \times S_{2,\beta}^L}} \Pr_{A^T \hookleftarrow \mathcal{U}(R_{\bar{q}}^{L\times n})} [\boldsymbol{A}\boldsymbol{v}_A = \boldsymbol{t} - \boldsymbol{E}\boldsymbol{v}_E - \boldsymbol{G}\boldsymbol{v}_G]. \tag{27}$$

Let $p(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) := \Pr_{\boldsymbol{A} \hookleftarrow \mathcal{U}(R_{\bar{q}}^{L\times n})}[\boldsymbol{A}\boldsymbol{v}_A = \boldsymbol{t} - \boldsymbol{E}\boldsymbol{v}_E - \boldsymbol{G}\boldsymbol{v}_G]$. Since the L rows of $\boldsymbol{A}$ are sampled independently and uniformly random in $R_{\bar{q}}^n$, we have

$$p(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) = \prod_{i \in [L]} \frac{|A_i(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})|}{\bar{q}^{dn}}, \tag{28}$$

where $A_i(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) := \{\boldsymbol{a}_i^T \in R_{\bar{q}}^n : \boldsymbol{a}_i^T\boldsymbol{v}_A = t_i - \boldsymbol{e}_i^T\boldsymbol{v}_E - \boldsymbol{g}_i^T\boldsymbol{v}_E\}$ for $i \in [L]$. The d-dimensional (over $\mathbb{Z}_{\bar{q}}$) ring $R_{\bar{q}}$ is isomorphic by the Chinese Remainder Theorem (CRT) to the cross-product of the (d/ℓ_q)-dimensional fields $R_{\bar{q}}^{(u)} := \mathbb{Z}_{\bar{q}}[x]/(f_u(x))$ for $u \in [\ell_q]$. For $z \in R_{\bar{q}}$, we denote by $z^{(u)} := z \bmod f_u(x) \in R_{\bar{q}}^{(u)}$ its reduction mod $f_u(x)$ (and analogously for vectors and matrices over $R_{\bar{q}}$). Let $\boldsymbol{a}_i^T$, $\boldsymbol{e}_i^T$, and $\boldsymbol{g}_i^T$, be the i'th row of $\boldsymbol{A}$, $\boldsymbol{E}$, and $\boldsymbol{G}$, respectively. We then have

$$p(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) = \prod_{i \in [L]} \frac{\prod_{u \in [\ell_q]} |A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})|}{\bar{q}^{dn}}, \tag{29}$$

where for $i \in [L]$ and $u \in [\ell_q]$, we define $A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})$ as below:

$$\{\boldsymbol{a}_i^{(u)} \in (R_{\bar{q}}^{(u)})^n : (\boldsymbol{a}_i^{(u)})^T \boldsymbol{v}_A^{(u)} = t_i^{(u)} - (\boldsymbol{e}_i^{(u)})^T \boldsymbol{v}_E^{(u)} - (\boldsymbol{g}_i^{(u)})^T \boldsymbol{v}_G^{(u)}\}.$$

Now, for each $u \in [\ell_q]$ and fixed $\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}$, since $R_{\bar{q}}^{(u)}$ is a field of size $\bar{q}^{d/\ell_q}$, there are two possible cases for $|A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})|$, depending on the value of $\boldsymbol{v}_A^{(u)} \in (R_{\bar{q}}^{(u)})^n$:

- Case 1: $\boldsymbol{v}_A^{(u)} \neq 0$. In this case, there exists $\ell' \in [n]$ such that $v_{A,\ell'}^{(u)} \neq 0$ and hence is invertible in the field $R_{\bar{q}}^{(u)}$. This implies that for any possible choice of $\{a_{i,\ell''}^{(u)}\}_{\ell'' \neq \ell'} \in (R_{\bar{q}}^{(u)})^{n-1}$, there exists a unique value for $a_{i,\ell'}^{(u)} \in R_{\bar{q}}^{(u)}$ satisfying $(\boldsymbol{a}_i^{(u)})^T \boldsymbol{v}_A^{(u)} = t_i^{(u)} - (\boldsymbol{e}_i^{(u)})^T \boldsymbol{v}_E^{(u)} - (\boldsymbol{g}_i^{(u)})^T \boldsymbol{v}_G^{(u)}$. Hence, in this case, we have $|A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})| = |(R_{\bar{q}}^{(u)})^{n-1}| = \bar{q}^{d/\ell_q(n-1)}$.
- Case 2: $\boldsymbol{v}_A^{(u)} = 0$. In this case, since $(\boldsymbol{a}_i^{(u)})^T \boldsymbol{v}_A^{(u)} = 0$ regardless of the choice of $(\boldsymbol{a}_i^{(u)})^T$, we have $|A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})| = |(R_{\bar{q}}^{(u)})^n| = \bar{q}^{d/\ell_q n}$ if $t_i^{(u)} - (\boldsymbol{e}_i^{(u)})^T \boldsymbol{v}_E^{(u)} - (\boldsymbol{g}_i^{(u)})^T \boldsymbol{v}_G^{(u)} = 0$ and $|A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})| = 0$ otherwise.

For a vector $\boldsymbol{v} \in R_{\bar{q}}^m$ and any $m \geq 1$, let us denote by $Z(\boldsymbol{v}) \subseteq [\ell_q]$ the set of $u \in [\ell_q]$ such that $\boldsymbol{v}^{(u)} = \boldsymbol{0} \in (R_{\bar{q}}^{(u)})^m$ (i.e. the set of CRT slots that are zero for all m coordinates of $\boldsymbol{v}$ over $R_{\bar{q}}$).

We conclude from the above that for any fixed $\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}$, we have

$$\prod_{u \in [\ell_q]} |A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})| =$$
$$\begin{cases} \bar{q}^{d(n-1)+d/\ell_q |Z(\boldsymbol{v}_A)|}, & \text{if } Z(\boldsymbol{v}_A) \subseteq Z(t_i - \boldsymbol{e}_i^T \boldsymbol{v}_E - \boldsymbol{g}_i^T \boldsymbol{v}_G) \\ 0, & \text{otherwise.} \end{cases}$$

For $r \in [\ell_q - 1]$, let V_r denote the set of $(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) \in R_{\bar{q}}^n \setminus \boldsymbol{0} \times R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^{\nu} \times S_{2,\beta}^L$ such that $|Z(\boldsymbol{v}_A)| = r$ and $Z(\boldsymbol{v}_A) \subseteq Z(t_i - \boldsymbol{e}_i^T \boldsymbol{v}_E - \boldsymbol{g}_i^T \boldsymbol{v}_G)$ for all $i \in [L]$.

Summarising, the above discussion shows that

$$p_1 \leq \sum_{r \in [\ell_q - 1]} \frac{|V_r|}{\bar{q}^{Ld(1-r/\ell_q)}} \tag{30}$$

and it remains to upper bound $|V_r|$ for $r \in [\ell_q - 1]$. For each possible choice of $\boldsymbol{v}_A \in R_{\bar{q}}^n \setminus \boldsymbol{0}$ with $Z(\boldsymbol{v}_A) := Z$ and $|Z| = r$ and $(\boldsymbol{v}_E, \boldsymbol{v}_G) \in R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^{\nu}$, let $T(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G)$ denote the set of $\boldsymbol{t} \in S_{2,\beta}^L$ such that $(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) \in V_r$. We denote by $I_{Z,R_{\bar{q}}}$ the ideal lattice of elements w in $R_{\bar{q}}$ with $Z \subseteq Z(w)$, i.e having zero CRT slots in Z, i.e. $I_{Z,R_{\bar{q}}} := \{w \in R : w^{(u)} = 0 \bmod \bar{q} \text{ for all } u \in Z\}$. Notice that $\boldsymbol{t} \in T(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G)$ if and only if

$$t_i \in I_{Z,R_{\bar{q}}} + c_i$$

where $c_i := \boldsymbol{e}_i^T \boldsymbol{v}_E + \boldsymbol{g}_i^T \boldsymbol{v}_G$ for $i \in [L]$. For each $i \in [L]$, let $N_i(c_i)$ denote the number of t_i in $(I_{Z,R_{\bar{q}}} + c_i) \cap S_{2,\beta}$, i.e. the number of points in the coset containing c_i of the lattice $I_{Z,R_{\bar{q}}}$ that are inside the Euclidean ball $S_{2,\beta}$ of radius β. We upper bound $N_i(c_i)$ by a volume argument. Namely, let λ denote the minimum of $I_{Z,R_{\bar{q}}}$, i.e. the Euclidean norm of the shortest non-zero vector in $I_{Z,R_{\bar{q}}}$. We consider the enlarged ball $S_{2,\beta+\lambda/2}$ of radius $\beta + \lambda/2$, which contains the union of $N_i(c_i)$ non-intersecting balls of radius $\lambda/2$ centered on the points of $(I_{Z,R_{\bar{q}}} + c_i) \cap S_{2,\beta}$. It follows that $N_i(c_i) \leq \frac{\mathrm{vol}(S_{2,\beta+\lambda/2})}{\mathrm{vol}(\lambda/2)} = (2\beta/\lambda + 1)^d$ for all $i \in [L]$. By Lemma [36, Lemma 2], we have $\lambda \geq \bar{q}^{r/\ell_q}/\sqrt{d}$, and we conclude that, setting $\beta := \bar{q}/c$, we have

$$N_i(c_i) \leq (2\sqrt{d}\bar{q}^{1-r/\ell_q}/c + 1)^d \text{ for } i \in [L].$$

It follows that, for each $r \in [\ell_q - 1]$

$$|V_r| \leq N_Z N_A N_E N_G \prod_{i \in L} N_i(c_i) \leq \bar{q}^{(1-r/\ell_q)n+\ell'+\nu)d} \cdot (2\sqrt{d}\bar{q}^{1-r/\ell_q}/c + 1)^{Ld}, \tag{31}$$

where $N_Z = \binom{\ell_q}{r}$ is the number of possible $Z \subset [\ell_q]$ with $|Z| = r$, $N_A \leq \bar{q}^{(1-r/\ell_q)nd}$ is the number of possible $\boldsymbol{v}_A \in R_{\bar{q}}^n \setminus \boldsymbol{0}$ with $Z(\boldsymbol{v}_A) = Z$ and $|Z| = r$, $N_E \leq \bar{q}^{\ell' d}$ is the number of possible $\boldsymbol{v}_E$ in $R_{\bar{q}}^{\ell'}$, and $N_G \leq \bar{q}^{\nu d}$ is the number of possible $\boldsymbol{v}_G$ in $R_{\bar{q}}^{\nu}$. Plugging (31) into (30) give (13) and completes the proof. □

E.3 Proof of Lemma 3

Proof. It suffices to prove the claim with $m_q = \lceil \log_\beta(\bar{q}) \rceil$ since the minimum distance of $\Lambda_{\bar{q}}(G))$ cannot decrease as m_q increases. Let $\bar{q} = \beta^{m_q-1} + \bar{q}_{m_q-2}\beta^{m_q-2} + \cdots + \bar{q}_0\beta^0$, where $0 \leq \bar{q}_i < \beta$, for $0 \leq i \leq m_q - 1$. Every nonzero lattice vector in $\Lambda_{\bar{q}}(G)$ will have components of the form $v \cdot \beta^i \bmod \bar{q}$ for $0 \leq i \leq m_q - 1$. Notice that, since β^i are integers (constant polynomials) in $R_{\bar{q}}$, then using the coefficient embedding, for any $v = v_0 + v_1 x + \cdots + v_{d-1}x^{d-1} \in R_{\bar{q}}$ with $v_j \in \mathbb{Z}_{\bar{q}}$, we have that $\|v \cdot (1, \beta, \ldots, \beta^{m_q-1}) \bmod \bar{q}\|_\infty = \max_{j \in [d-1]} \|v_j \cdot (1, \beta, \ldots, \beta^{m_q-1}) \bmod \bar{q}\|_\infty$. Therefore, the minimum $\lambda_1^\infty(\Lambda_{\bar{q}}(G))$ of $\|v \cdot (1, \beta, \ldots, \beta^{m_q-1}) \bmod \bar{q}\|_\infty$ over $v \in R_{\bar{q}} \setminus \{0\}$ is equal to the minimum of $\|v_0 \cdot (1, \beta, \ldots, \beta^{m_q-1}) \bmod \bar{q}\|_\infty$ over $v_0 \in \mathbb{Z}_{\bar{q}} \setminus \{0\}$. Indeed, we will show that

$$\lambda_1^\infty(\Lambda_{\bar{q}}(G)) = \min_{v \in \mathbb{Z}_{\bar{q}} \setminus \{0\}} \max_{0 \leq i \leq m_q - 1} |v \cdot \beta^i \bmod \bar{q}| \geq \bar{q}/(2\beta).$$

If $\lambda_1^\infty(\Lambda_{\bar{q}}(G)) < \bar{q}/(2\beta)$, we show a contradiction. Assume that $\lambda_1^\infty(\Lambda_{\bar{q}}(G))$ is achieved for a non-zero v^*. Let $i^* := \arg\max_{0 \leq i \leq m_q-1} |v^* \cdot \beta^i \bmod \bar{q}|$. We now claim that $i^* = m_q - 1$, otherwise we have that $|v^* \cdot \beta^{i^*} \bmod \bar{q}| < \bar{q}/(2\beta)$ (due to upper bound on λ_1^∞) and hence $|v^* \cdot \beta^{i^*+1} \bmod \bar{q}| = \beta \cdot |v^* \cdot \beta^{i^*} \bmod \bar{q}| > |v^* \cdot \beta^{i^*} \bmod \bar{q}|$ since $\beta > 1$, which is a contradiction. This yields $i^* = m_q - 1$ and therefore $\lambda_1^\infty(\Lambda_{\bar{q}}(G)) = |v^* \cdot \beta^{m_q-1} \bmod \bar{q}|$. Let us re-write $|v^* \cdot \beta^{m_q-1} \bmod \bar{q}| = |v_0^* \cdot \beta^{i_0^*}|$ where the latter is a factorization over $\mathbb{Z}$, for some integer $v_0^* \neq 0$ and $0 \leq i_0^* \leq m_q - 1$ such that $\gcd(v_0^*, \beta) = 1$. We claim that $i_0^* < m_q - 1$. Otherwise, if $i_0^* = m_q - 1$, then $|v_0^* \beta^{m_q-1} \bmod \bar{q}| = |v_0^* \beta^{m_q-1}| \geq \beta^{m_q-1} \geq \beta^{m_q}/\beta \geq \bar{q}/\beta$, contradicting with the upper bound $\bar{q}/(2\beta)$. Let us divide v_0^* by β over $\mathbb{Z}$ to get a quotient v_0 and remainder $r_0 \neq 0$ such that $v_0^* := \beta v_0 + r_0$, with $|r_0| \leq \beta/2$ and $|v_0| \leq |\lceil v_0^*/\beta \rceil| < \bar{q}/(2\beta^2) + 1$ (since $\lambda_1^\infty(\Lambda_{\bar{q}}(G)) = |v_0^* \cdot \beta^{i_0^*}| < \bar{q}/(2\beta)$). Then we have by definition of i^* that

$$\begin{aligned} |v^* \cdot \beta^{m_q-1} \bmod \bar{q}| &\geq |v^* \cdot \beta^{m_q-(i_0^*+2)} \bmod \bar{q}| && (32)\\ &= |v_0^* \cdot \beta^{-1} \bmod \bar{q}| && (33)\\ &= |(\beta v_0 + r_0) \cdot \beta^{-1} \bmod \bar{q}| && (34)\\ &= |v_0 + r_0 \cdot \beta^{-1} \bmod \bar{q}| \\ &\geq |r_0 \cdot \beta^{-1} \bmod \bar{q}| - |v_0| \\ &> (\bar{q}/\beta - 1/2) - (\bar{q}/(2\beta^2) + 1), && (35) \end{aligned}$$

where (32) uses $i_0^* < m_q - 1$ and hence $i := m_q - (i_0^* + 2) \geq 0$, (33) uses the equality $|v^* \cdot \beta^{m_q-1} \bmod \bar{q}| = |v_0^* \cdot \beta^{i_0^*}|$, (34) uses the representation of v_0^* in terms of v_0 and r, and (35) is true using the above upper bound on $|v_0|$ and the lower bound $|r_0 \cdot \beta^{-1} \bmod \bar{q}| \geq \bar{q}/\beta - 1/2$. To show the latter lower bound, write $\beta^{-1} \bmod \bar{q} := \frac{k\bar{q}+1}{\beta}$, for some $k \in \mathbb{Z}$ with $\gcd(k, \beta) = 1$. Then,

$$\begin{aligned} |r_0 \cdot \beta^{-1} \bmod \bar{q}| &= \left| r_0 \cdot \frac{k\bar{q}+1}{\beta} \bmod \bar{q} \right| \\ &= \left| (r_0 \cdot k \bmod \beta) \cdot \frac{\bar{q}}{\beta} + \frac{r_0}{\beta} \bmod \bar{q} \right| \\ &\geq \left| (r_0 \cdot k \bmod \beta) \cdot \frac{\bar{q}}{\beta} \bmod \bar{q} \right| - \left| \frac{r_0}{\beta} \right| \\ &\geq \bar{q}/\beta - 1/2, \end{aligned}$$

as claimed, where the last inequality uses the fact that $(r_0 \cdot k \bmod \beta) \neq 0$ (because $\gcd(k, \beta) = 1$ and $r_0 \neq 0 \bmod \beta$ since $\gcd(v_0^*, \beta) = 1$) and that $|\frac{r_0}{\beta}| \leq 1/2$. We now have that (35) is a contradiction with $\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) < \bar{q}/(2\beta)$ if $\bar{q}/\beta - \bar{q}/(2\beta^2) - 3/2 \geq \bar{q}/(2\beta)$, which is equivalent to the assumed condition $\bar{q} \geq 3\beta^2/(\beta - 1)$.

□

E.4 Proof of Lemma 4

Proof. We distinguish between two cases: $\boldsymbol{v}_G = \boldsymbol{0}$ and $\boldsymbol{v}_G \neq \boldsymbol{0}$. The first case ($\boldsymbol{v}_G = \boldsymbol{0}$) is similar to Lemma 11 of [49] and follows from a probabilistic upper bound on $\|\bar{\boldsymbol{E}}\boldsymbol{v}_E\|$. Indeed, by the Cauchy-Schwartz inequality, $\|\bar{\boldsymbol{E}}\boldsymbol{v}_E\| \leq \|\bar{\boldsymbol{E}}\| \cdot \|\boldsymbol{v}_E\|$. Since each row of $\boldsymbol{E}$ has norm less than $2s\sqrt{d\ell'/(2\pi)}$ except with probability $\leq 2^{-d\ell'}$ by Lemma 6 with $k = 2$, we get by a union bound over the Ld rows of $\boldsymbol{E}$ that $\|\boldsymbol{E}\| \leq 2s\sqrt{d\ell'/(2\pi)}$ except with probability $\leq Ld2^{-d\ell'}$. The same bound holds for $\|\bar{\boldsymbol{E}}\|$ since $2s\sqrt{d\ell'/(2\pi)} \geq 1$. Since $c_3 \geq c_2 \cdot 2s\sqrt{d\ell'/(2\pi)}$ and $c_2 > 2$, we therefore get for the first case:

$$\Pr\left[\exists \boldsymbol{v}_E \in \mathbb{Z}^{d\ell'}, \|\boldsymbol{v}_E\| \leq \frac{\bar{q}}{c_3} : \bar{\boldsymbol{E}}\boldsymbol{v}_E \bmod \bar{q} \in \Lambda_{\bar{q}}(\bar{\boldsymbol{E}}) \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{d\ell'}\right] \leq Ld2^{-d\ell'}. \qquad (36)$$

We now assume $\boldsymbol{v}_G \neq \boldsymbol{0}$. Here, it suffices to show that, if $\|\boldsymbol{E}\| \leq 2s\sqrt{d\ell'/(2\pi)}$ and $\|\boldsymbol{v}_E\| \leq q/c_3$, then $\|\bar{\boldsymbol{E}}\boldsymbol{v}_E + \bar{\boldsymbol{G}}\boldsymbol{v}_G\| \geq \|\boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G\| \geq \bar{q}/c_2$. Indeed,

$$\begin{aligned} \|\boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G\| \geq \|\boldsymbol{G}\boldsymbol{v}_G\| - \|\boldsymbol{E}\boldsymbol{v}_E\| &\geq \bar{q}/(2\beta) - \|\boldsymbol{E}\boldsymbol{v}_E\| && (37)\\ &\geq \bar{q}/(2\beta) - \|\boldsymbol{E}\| \cdot \|\boldsymbol{v}_E\| && (38)\\ \geq \bar{q}/(2\beta) - 2s\sqrt{d\ell'/(2\pi)} \cdot \frac{\bar{q}}{8\beta s\sqrt{d\ell'/(2\pi)}} &= \bar{q}/(4\beta) && (39) \end{aligned}$$

where (37) is induced from triangle inequality and Lemma 3, (38) uses Cauchy-Schwartz inequality, and (39) holds by assumed bounds on $\|\boldsymbol{E}\|$ and $\|\boldsymbol{v}_E\|$.

□

E.5 Proof of Lemma 5

We first state a result from [38].

Lemma 13 (Adapted from Theorem 3.1 of [38]). *Let $\varepsilon \in [0, 1)$, $\boldsymbol{S}$ be a full column rank matrix, Λ_0 be a lattice with a coset $A = \Lambda_0 + a \subseteq span(\boldsymbol{S})$, $\boldsymbol{T}$ be a matrix such that ker($\boldsymbol{T}$) is a Λ_0-subspace and $\eta_\varepsilon(\Lambda_0 \cap ker(\boldsymbol{T})) \leq \boldsymbol{S}$. Then, we have that $\Delta(\boldsymbol{T} \cdot \mathcal{D}_{A,\boldsymbol{S}}, \mathcal{D}_{\boldsymbol{T}A,\boldsymbol{T}\boldsymbol{S}}) \leq \frac{2\varepsilon}{1-\varepsilon}$.*

Let $\mathbf{S} = r \cdot \mathbf{I}$, $\Lambda_0 = \Lambda_q^\perp(\mathbf{G}) \times R^{\ell'}$, $a = \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{0} \end{pmatrix} \in R^L \times R^{\ell'}$, and $\boldsymbol{T} = \bar{\boldsymbol{E}}^T$We first note that $\boldsymbol{S} = r \cdot \mathbf{I}$ has full column rank. Then we have $\boldsymbol{T}A = \bar{\boldsymbol{E}}^T A = \boldsymbol{E}^T(\Lambda_q^\perp(\mathbf{G}) + \mathbf{c}) + R^{\ell'} = R^{\ell'}$ thanks to the identity matrix at the bottom of $\bar{\boldsymbol{E}}^T$, and $\boldsymbol{T}\boldsymbol{S} = \bar{\boldsymbol{E}}^T r$. Now, we calculate a bound on $\eta_\varepsilon(\Lambda_0 \cap \ker(\boldsymbol{T}))$, where $\Lambda_0 \cap \ker(\boldsymbol{T}) := \Lambda = \{\boldsymbol{v} \in \Lambda_q^\perp(\mathbf{G}) \times R^{\ell'} : \boldsymbol{v}^T \bar{\boldsymbol{E}} = \boldsymbol{0}^T\}$.

Lemma 14 (Smoothing parameter of orthogonal module lattice, adapted from [23]). *Let $L = \nu m_q$ with m_q and ν be defined as above. For $\mathbf{G}$ and $\mathbf{E}$, E_∞ and $\epsilon > 0$ as defined in Lemma 5, and the lattice $\Lambda := \{\boldsymbol{v} \in \Lambda_q^\perp(\mathbf{G}) \times R^{\ell'} : \boldsymbol{v}^T\bar{\mathbf{E}} = \mathbf{0}^T\}$, we have $\eta_\epsilon(\Lambda) \leq \left(\sqrt{m_q}(\beta - 1) + \sqrt{\ell' d}((m_q - 1)(\beta - 1) + 1)E_\infty\right) \cdot \sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}}$.*

Proof. Recall that that Λ has a $\mathbb{Z}$-rank $L \cdot d$. By Lemma 12, it suffices to show that the last minimum $\lambda_{Ld}(\Lambda)$ of Λ is upper bounded by $\gamma := \sqrt{m_q} \cdot \left(1 + \sqrt{\ell' d} \cdot E_\infty\right)$. Namely we exhibit Ld $\mathbb{R}$-linearly independent vectors $\boldsymbol{u}_i$ ($i \in [Ld]$) in Λ whose Euclidean norm is upper bounded by γ. Let $\bar{\boldsymbol{B}} \in \mathbb{Z}^{m_q d \times m_q d}$ denote a column $\mathbb{Z}$-basis for $\Lambda_q(\mathrm{rot}(\boldsymbol{g}))$. We take $\bar{\boldsymbol{B}} = I_d \otimes \bar{\boldsymbol{B}}'$ with $\bar{\boldsymbol{B}}' \in \mathbb{Z}^{m_q \times m_q}$ having its j'th column of the form $\boldsymbol{b}'_j = \beta\boldsymbol{e}_j - \boldsymbol{e}_{j+1}$ for $j \in [m_q - 1]$ (with $\boldsymbol{e}_j$ denoting the jth unit vector having 1 in coordinate j and zeroes elsewhere) and $\boldsymbol{b}'_{m_q} = (q_0, q_1, \ldots, q_{m_q-1})^T$, here $q_i \in \{0, \ldots, \beta - 1\}$ is the i'th digit in the β-ary representation of q (i.e. $q = \sum_{j=0}^{m_q-1} q_j\beta^j$).

Let $\boldsymbol{B} \in \mathbb{Z}^{(L+\ell')d \times (L+\ell')d}$ denote a column $\mathbb{Z}$-basis for $\Lambda_q(\mathrm{rot}(\boldsymbol{G}))$ (namely, we take for $\boldsymbol{B}$ the matrix whose first Ld rows consist of $(I_\nu \otimes \bar{\boldsymbol{B}}, \boldsymbol{0}^{Ld \times \ell' d})$ and whose last $\ell' d$ rows consist of $(\boldsymbol{0}^{\ell' d \times Ld}, \boldsymbol{I}_{\ell' d})$). Let $\boldsymbol{B}_2 \in \mathbb{Z}^{(L+\ell')d \times \ell' d}$ denote the last $\ell' d$ columns of $\boldsymbol{B}$. Note that with $\mathrm{rot}(\bar{\boldsymbol{E}})^T = (\mathrm{rot}(\boldsymbol{E})^T, \boldsymbol{I}_{\ell' d})$, we get $\mathrm{rot}(\bar{\boldsymbol{E}})^T \cdot \boldsymbol{B}_2 = \boldsymbol{I}_{\ell' d}$. Now, for $i \in [Ld]$, we let $\boldsymbol{b}_i$ denote the ith column of $\boldsymbol{B}$, and define $\boldsymbol{u}_i := \boldsymbol{b}_i - \boldsymbol{B}_2 \cdot \mathrm{rot}(\bar{\boldsymbol{E}})^T \cdot \boldsymbol{b}_i = \boldsymbol{K} \cdot \boldsymbol{b}_i$, with $\boldsymbol{K} := \boldsymbol{I}_{(L+\ell')d} - \boldsymbol{B}_2 \cdot \mathrm{rot}(\bar{\boldsymbol{E}})^T$. The vectors $(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{Ld})$ are linearly independent over $\mathbb{R}$ since the top Ld rows of $\boldsymbol{K}$ is a full-rank Ld matrix $(\boldsymbol{I}_{Ld}, \boldsymbol{0}^{Ld \times \ell' d})$. Moreover, from $\mathrm{rot}(\bar{\boldsymbol{E}})^T \cdot \boldsymbol{B}_2 = \boldsymbol{I}_{\ell' d}$ and the definition of $\boldsymbol{b}_i$ we have $\boldsymbol{u}_i^T \mathrm{rot}(\bar{\boldsymbol{E}}) = \boldsymbol{0}$ so $\boldsymbol{u}_i \in \Lambda$ for $i \in [Ld]$ as required. It remains to bound the norm of the $\boldsymbol{u}_i$'s. Note that each entry $y \in \mathbb{Z}$ of $\mathrm{rot}(\bar{\boldsymbol{E}})^T \cdot \boldsymbol{b}_i$ is an inner product between a row of $\mathrm{rot}(\bar{\boldsymbol{E}})^T$ of infinity norm $\leq E_\infty$ and a vector $\boldsymbol{b}_i$ having a 1-norm $\|\boldsymbol{b}_i\|_1 \leq \max(\beta + 1, (m_q - 1)(\beta - 1) + 1) = (m_q - 1)(\beta - 1) + 1$ (where we have used the form of $\boldsymbol{B}'$ defined above and $m_q \geq 3$), so $|y| \leq ((m_q - 1)(\beta - 1) + 1) \cdot E_\infty$. Since there are $\ell' d$ coordinates in $\boldsymbol{u}_i$, we get $\|\boldsymbol{u}_i\| \leq \|\boldsymbol{b}_i\| + \sqrt{\ell' d} \cdot ((m_q - 1)(\beta - 1) + 1) \cdot E_\infty \leq \sqrt{m_q}(\beta - 1) + \sqrt{\ell' d}((m_q - 1)(\beta - 1) + 1)E_\infty$.

□

When $\boldsymbol{E} \in R^{L \times \ell'}$ is chosen from a Gaussian distribution $D_{R^{L \times \ell'}, s}$, Lemma 6 and a union bound over the $\ell' Ld$ integer coefficients of the entries of $\boldsymbol{E}$ implies that the maximal absolute value E_∞ is upper bounded by $ks/\sqrt{2\pi}$ except with probability $\leq \epsilon$ if $k^2 - 2\ln(k) + 1 \geq 2\ln(1/\bar{\epsilon})$ where $\bar{\epsilon} := \epsilon/(L\ell' d)$. We observe that the latter inequality is satisfied with $k := \sqrt{2\ln(1/\bar{\epsilon}) + \ln\ln(1/\bar{\epsilon})}$ if $\bar{\epsilon} \leq 0.001$. Combining this with Lemma 13 and 14, and noting that $2\epsilon/(1-\epsilon) \leq 4\epsilon$ for $\epsilon \leq 1/2$, we complete the proof of Lemma 5.

E.6 Proof of Theorem 2

Proof. Let $(\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{A}, \boldsymbol{E}) \leftarrow \mathsf{HGSW.Setup}(1^\lambda, 1^\ell)$ and $\boldsymbol{C}_k \leftarrow \mathsf{HGSW.Encrypt}(k, \boldsymbol{S}, \boldsymbol{\mu})$ for $k \in [m]$. For $\boldsymbol{a} = (a_1, \ldots, a_m) \in S := \mathcal{D}_r^m$, we have that

$$\boldsymbol{c}^* = \sum_{j=0}^{m/\nu-1} \left(\sum_{i=1}^{\nu} \boldsymbol{g}_{\text{rand}}^{-1}(a_{j\nu+i}) \cdot \boldsymbol{C}_{j\nu+i} + [\boldsymbol{0}^n, \boldsymbol{y}_j^T] \right)$$

Let us now find the $\mathsf{HGSW.Decrypt}(\boldsymbol{c}^*, \boldsymbol{S})$ by calculating $\boldsymbol{H}^* = \lceil (p/q) \cdot \langle \boldsymbol{c}^*, \bar{\boldsymbol{S}} \rangle \rfloor$, where $\bar{\boldsymbol{S}}^T = \begin{bmatrix} -\boldsymbol{S}^T & \boldsymbol{I}_{\ell'} \end{bmatrix}$. Replacing $\boldsymbol{g}_{\text{rand}}^{-T}(a_{j\nu+i}) = \boldsymbol{x}_{j,i}^T$ and $\boldsymbol{C}_i$ from the above and their definitions, we get that:

$$\begin{aligned} \boldsymbol{H}^* &= \left\lceil \frac{p}{q} \sum_{j=0}^{m/\nu-1} \left(\sum_{i=1}^{\nu} \boldsymbol{x}_{j\nu+i}^T \left(\boldsymbol{C}_{j\nu+i} + \frac{q}{p} \boldsymbol{H}_{j\nu+i} + [\boldsymbol{0}, \boldsymbol{y}_j^T] \right) \cdot \begin{bmatrix} -\boldsymbol{S} \\ \boldsymbol{I}_{\ell'} \end{bmatrix} \right) \right\rfloor \\ &= \left\lceil \frac{p}{q} \sum_{j=0}^{m/\nu-1} \left(\sum_{i=1}^{\nu} \left(\boldsymbol{x}_{j\nu+i}^T \boldsymbol{E}_{j\nu+i} + \frac{q}{p} a_{j\nu+i} \bar{\boldsymbol{\mu}}_{j\nu+i} \right) + \boldsymbol{y}_j^T \right) \right\rfloor \\ &= \left\lceil \frac{p}{q} \left(\sum_{k=0}^{m-1} \frac{q}{p} a_k \bar{\boldsymbol{\mu}}_k + \sum_{k=0}^{m-1} \boldsymbol{x}_k^T \boldsymbol{E}_k + \sum_{j=0}^{m/\nu-1} \boldsymbol{y}_j^T \right) \right\rfloor \end{aligned}$$

Letting $\bar{\boldsymbol{X}}^T := \begin{bmatrix} \boldsymbol{x}_1^T, & \ldots & , \boldsymbol{x}_m^T & , \boldsymbol{y}_0^T & , \ldots & , \boldsymbol{y}_{m/\nu-1}^T \end{bmatrix}$ and $\bar{\boldsymbol{E}} := \begin{bmatrix} \boldsymbol{E}_1, & \ldots & , \boldsymbol{E}_m & , \boldsymbol{I}_{\ell'} & , \ldots & , \boldsymbol{I}_{\ell'} \end{bmatrix}$, we now observe that thanks to (16), we have $\|\bar{\boldsymbol{X}}^T \bar{\boldsymbol{E}}\|_\infty < q/(2p)$ (i.e. no wraparound mod q) except with probability $\leq \epsilon$. Indeed, by Lemma 7, we have $\|\bar{\boldsymbol{X}}\| \leq r\sqrt{(mm_q + m\ell'/\nu)d}$ except with probability $\leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-(mm_q + m\ell'/\nu)d} \leq 2^{-(mm_q+m\ell'/\nu)d+2} \leq 4\epsilon$ using $\epsilon \leq 1/2$ and the choice of

$$r \geq (mm_q + \ell')dc\sqrt{\ln(2(mm_q + m\ell'/\nu)d(1+\epsilon^{-1}))/\pi}$$

and $2^{-(mm_q+m\ell'/\nu)d} \leq \epsilon$, where we have used the fact that $\eta_\epsilon(\Lambda_q^\perp(\boldsymbol{G})^T) \leq \beta^2\sqrt{\ln(2(mm_q + m\ell'/\nu)d(1+\epsilon^{-1}))/\pi}$ by Lemma 11 and Lemma 3, and that $c \geq c_2 \geq \beta^2$. Therefore, by Lemma 8, each integer coefficient of $\bar{\boldsymbol{X}}^T\bar{\boldsymbol{E}}$ has absolute value $\leq s\sqrt{(r^2(mm_q + m\ell'/\nu)d + 1)\ln(2((mm_q + m\ell'/\nu)d + 1)/\epsilon)/\pi}$ $< q/(2p)$ except with probability ϵ over the choice of $\boldsymbol{E}_i$ and $\boldsymbol{y}_i$ for all $i \in [m]$. □

E.7 Proof of Theorem 3

Proof. To prove the circuit privacy of our HGSW, we need to build a simulator $\mathcal{S}$. On input the security parameter λ, the secret key $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{A}, \boldsymbol{E})$ and a message vector $\boldsymbol{\mu} \in R_p^\ell$, the simulator computes the following:

- For $j = 0, \ldots, m/\nu - 1$, sample $\boldsymbol{b}_j^T \leftarrow \mathcal{U}(R_q^n)$ and $\boldsymbol{e}_j^T \leftarrow \mathcal{D}_{\mathbb{Z}^{\ell' d}, r\cdot\text{rot}(\bar{\boldsymbol{E}}_j)}$, where $\bar{\boldsymbol{E}}_j^T := [\boldsymbol{E}_j^T, \boldsymbol{I}_{\ell'}] \in R^{(L+\ell')\times\ell'}$.
- Compute the sum $(\boldsymbol{b}^T, \boldsymbol{e}^T) := \sum_{j=0}^{m/L-1}(\boldsymbol{b}_j^T, \boldsymbol{e}_j^T) \in R_q^n \times R^{\ell'}$.
- Compute $\bar{\boldsymbol{\mu}}^T = [\boldsymbol{\mu}^T | (\boldsymbol{T}\boldsymbol{\mu})^T] \in R_p^{\ell'}$ and output the simulated ciphertext $\boldsymbol{c}_1^* := (\boldsymbol{b}^T, \boldsymbol{b}^T\boldsymbol{S} + \boldsymbol{e}^T + \frac{q}{p}\bar{\boldsymbol{\mu}}^T) \in R_q^n \times R_q^{\ell'}$.

We show that the output $\boldsymbol{c}_1^*$ of the simulator is statistically indistinguishable from the ciphertext $\boldsymbol{c}_0^*$ computed by the challenger with the original Add algorithm. For this, observe that the latter is computed as

$$\boldsymbol{c}_0^* := \sum_{j=0}^{m/\nu-1} \left(\sum_{i=1}^{\nu} \boldsymbol{g}_{\text{rand}}^{-1}(a_{j\nu+i}) \cdot \boldsymbol{C}_{j\nu+i} + [\boldsymbol{0}^n, \boldsymbol{y}_j] \right) \tag{40}$$

$$= \left(\sum_{j=0}^{m/\nu-1} \boldsymbol{b}_j^T, (\sum_{j=0}^{m/\nu-1} \boldsymbol{b}_j^T)\boldsymbol{S} + \sum_{j=0}^{m/\nu-1} \boldsymbol{e}_j^T + \frac{q}{p}(\sum_{j=0}^{m/\nu-1} \bar{\boldsymbol{\mu}}_j^T \bmod p) \right), \tag{41}$$

where $\boldsymbol{b}_j^T := \sum_{i=1}^{\nu} \boldsymbol{x}_{j\nu+i}^T \boldsymbol{A}_{j\nu+i}$, $\boldsymbol{e}_j^T := \sum_{i=0}^{\nu-1} \boldsymbol{x}_{j\nu+i}^T \boldsymbol{E}_{j\nu+i} + \boldsymbol{y}_j$ and $\boldsymbol{x}_{j\nu+i}^T := \boldsymbol{g}_{\text{rand}}^{-1}(a_{j\nu+i})$, and $\bar{\boldsymbol{\mu}}_j^T := \sum_{i=1}^{\nu} \boldsymbol{x}_{j\nu+i}^T \frac{q}{p} \boldsymbol{H}(\bar{\boldsymbol{\mu}}_i) = \frac{q}{p}\left(\sum_{i=1}^{\nu} a_{j\nu+i}\bar{\boldsymbol{\mu}}_{j\nu+i} \bmod p\right)$. We have that $\sum_{j=0}^{m/\nu-1} \bar{\boldsymbol{\mu}}_j^T \bmod p = \sum_{i=1}^{m} a_i\bar{\boldsymbol{\mu}}_i \bmod p$, equal to the sum message vector $\bar{\boldsymbol{\mu}}$ computed by the simulator. Furthermore, for each $j = 0, \ldots, m/\nu - 1$ we apply Theorem 1, to conclude that in the Add algorithm, the distribution of $(\boldsymbol{b}_j^T, \boldsymbol{e}_j^T)$ is within statistical distance $O(\epsilon)$ from the distribution $D_j := \mathcal{U}(R_q^n) \times \mathcal{D}_{\mathbb{Z}^{\ell' d}, r\cdot\text{rot}(\bar{\boldsymbol{E}}_j)}$ used by the simulator to sample $(\boldsymbol{b}_j^T, \boldsymbol{e}_j^T)$. It follows that the distribution of $(\boldsymbol{b}^T, \boldsymbol{e}^T) := \sum_{j=0}^{m/\nu-1}(\boldsymbol{b}_j^T, \boldsymbol{e}_j^T)$ in the Add algorithm is within statistical distance $18(m/\nu)\cdot\epsilon$ of its distribution in the simulation, and the same bound therefore applies for the statistical distance between the distributions of $\boldsymbol{c}_1^*$ and $\boldsymbol{c}_0^*$. □

F HGSW ALGORITHMS WITH THE NTT

In this Section, we provide full details of the NTT-based variants of our HGSW algorithms, that were used in our implementation. They are given in Alg. 1– 4.

Notations. Recall that $R_p = \mathbb{Z}_p[x]/(x^d+1)$ and $x^d + 1$ splits into a product of ℓ_p irreducible factors mod p, each of degree f. Therefore, the plaintext ring R_p is isomorphic to $\mathbb{F}^{\ell_p}$ via the Chinese Remainder Theorem (CRT) over R_p, where $\mathbb{F}$ denotes the extension field of $\mathbb{Z}_p$ of degree f and is the underlying LPCP field. Using CRT over R_p, our encryption algorithm $\mathsf{HGSW.Encrypt}$ encodes a plaintext vector $\boldsymbol{q}_i \in \mathbb{F}^{4\rho}$ into a vector $\mu_i \in R_p^\ell$ of $\ell = \lceil 4\rho/\ell_p \rceil$ elements of R_p, where each R_p element holds in its ℓ_p CRT slots over R_p a block of ℓ_p plaintext components in $\mathbb{F}$. We denote this plaintext encoding map by $\mathsf{CRTEncode}$ and its inverse by $\mathsf{CRTDecode}$. Let $\mathsf{CRT}(a) := (a_p, a_{\bar{q}}) \in R_p \times R_{\bar{q}}$ be the (coefficient-wise) CRT decomposition of $a \in R_q$. Let $\mathsf{CRT}^{-1}(a_p, a_{\bar{q}}) := a \in R_q$ be the (coefficient-wise) inverse CRT of $(a_p, a_{\bar{q}}) \in R_p \times R_{\bar{q}}$. Let $\mathsf{NTT}_{R'}(a)$ be the Number Theoretic Transform (NTT) of $a \in R'$ over a ring R'. Let $\mathsf{NTT}_{R'}^{-1}(\tilde{a})$ be the inverse NTT of $\tilde{a}$ over R'. Assume $\mathsf{CRT}(a) = (a_p, a_{\bar{q}}) \in R_p \times R_{\bar{q}}$ for $a \in R_q$. Let $\mathsf{NTT}(a) := (\tilde{a}_{p'}, \tilde{a}_{\bar{q}})$ where $\tilde{a}_{\bar{q}} = \mathsf{NTT}_{R_{\bar{q}}}(a_{\bar{q}}), \tilde{a}_{p'} = \mathsf{NTT}_{R_{p'}}(a_p)$. Let $\mathsf{NTT}^{-1}(\tilde{a}_{p'}, \tilde{a}_{\bar{q}}) := \mathsf{CRT}^{-1}(\mathsf{NTT}_{R_{p'}}^{-1}(\tilde{a}_{p'}) \bmod p, \mathsf{NTT}_{R_{\bar{q}}}^{-1}(\tilde{a}_{\bar{q}}))$. Also, let $\mathsf{NTT}_{R_{q'}}(a)$ be a similar construction to $\mathsf{NTT}(a)$ for $a \in R_{q'}$ such as, $\mathsf{NTT}_{R_{q'}}(a) = (\tilde{a}_{p'}, \tilde{a}_{\bar{q}'})$ where $\tilde{a}_{\bar{q}'} = \mathsf{NTT}_{R_{\bar{q}'}}(a_{\bar{q}'})$. Here, we let $q' = p\cdot\bar{q}'$, where $\bar{q}'$ is an NTT-friendly prime. Let $\mathsf{NTT}^{-1}(\tilde{a}_{p'}, \tilde{a}_{\bar{q}'}) := \mathsf{CRT}^{-1}(\mathsf{NTT}_{R_{p'}}^{-1}(\tilde{a}_{p'}) \bmod p, \mathsf{NTT}_{R_{\bar{q}'}}^{-1}(\tilde{a}_{\bar{q}'}))$. We write $\tilde{a} := \mathsf{NTT}(a)$ as the NTT of $a \in R_q$. Given $\tilde{a} = (\tilde{a}_{p'}, \tilde{a}_{\bar{q}})$, $\tilde{b} = (\tilde{b}_{p'}, \tilde{b}_{\bar{q}})$, let $\tilde{a} \pm \tilde{b} := (\tilde{a}_{p'} \pm \tilde{b}_{p'}, \tilde{a}_{\bar{q}} \pm \tilde{b}_{\bar{q}})$. Let $\tilde{a} \odot \tilde{b} := (\tilde{a}_{p'} \circ \tilde{b}_{p'}, \tilde{a}_{\bar{q}} \circ \tilde{b}_{\bar{q}})$, where $\circ$ is the pointwise multiplication. The $\odot$ operation between vectors and/or matrices in the NTT domain is computed similarly to the product between vectors and/or matrices in their original domain, except the ring multiplications are replaced with the ⊙ operations between NTT elements.

Algorithm 1 HGSW.Setup(N_g)

```
Require: $N_g$
Ensure: $\mathsf{sk}_i = (\boldsymbol{S}, \boldsymbol{T}, \tilde{\boldsymbol{A}}_i, \boldsymbol{E}_i)$ for $i \in [m]$
$m \leftarrow 2N_g$
for $i \in [m]$ do
    Let $\tilde{\boldsymbol{A}}_i := ((\tilde{\boldsymbol{A}}_i)_{p'}, (\tilde{\boldsymbol{A}}_i)_{\bar{q}})$, where $(\tilde{\boldsymbol{A}}_i)_{p'} \leftarrow \mathsf{NTT}_{R_{p'}}(\mathcal{U}(R_p^{m_q \times n})), (\tilde{\boldsymbol{A}}_i)_{\bar{q}} \leftarrow \mathcal{U}(R_{\bar{q}}^{m_q \times n})$
    $\boldsymbol{E}_i \leftarrow \mathcal{D}_{R,s}^{m_q \times \ell'}$
    Let $\tilde{\boldsymbol{E}}_i := \mathsf{NTT}(\boldsymbol{E}_i)$
end for
$\boldsymbol{S} \leftarrow \mathcal{D}_{R,s}^{n \times \ell'}$
Let $\tilde{\boldsymbol{S}} := \mathsf{NTT}(\boldsymbol{S})$
$\boldsymbol{T} \leftarrow \mathcal{U}(R_p^{\tau \times \ell})$
return $\mathsf{sk}_i = (\tilde{\boldsymbol{S}}, \boldsymbol{T}, \tilde{\boldsymbol{A}}_i, \tilde{\boldsymbol{E}}_i)$ for $i \in [m]$
```

Algorithm 2 HGSW.Encrypt($i, \mathsf{sk}_i = (\tilde{\boldsymbol{S}}, \boldsymbol{T}, \tilde{\boldsymbol{A}}_i, \tilde{\boldsymbol{E}}_i), \boldsymbol{q}_i \in \mathbb{F}^{4\rho}$)

```
Require: $i, \mathsf{sk}_i = (\tilde{\boldsymbol{S}}, \boldsymbol{T}, \tilde{\boldsymbol{A}}_i, \tilde{\boldsymbol{E}}_i), \boldsymbol{q}_i \in \mathbb{F}^{4\rho}$
Ensure: $\tilde{C}_i$
$\boldsymbol{\mu}_i \leftarrow \mathsf{CRTEncode}(\boldsymbol{q}_i)$
$\bar{\boldsymbol{\mu}}_i^T = [\boldsymbol{\mu}_i^T | (\boldsymbol{T}\boldsymbol{\mu}_{\boldsymbol{i}})^T] \in R_p^{\ell'}$
Let $\tilde{\boldsymbol{\mu}}_i^T := \mathsf{NTT}_{R_{p'}}(\bar{\boldsymbol{\mu}}_i^T)$
Parse $\tilde{\boldsymbol{\mu}}_i^T := (\tilde{\mu}_{i,1}, \ldots, \tilde{\mu}_{i,\ell'})$
Assume $\mathsf{CRT}(\boldsymbol{g}) = (\boldsymbol{g}_p, \boldsymbol{g}_{\bar{q}}), \tilde{\boldsymbol{g}}_{p'} = \mathsf{NTT}_{R_{p'}}(\boldsymbol{g}_p)$
Let $\boldsymbol{H}_i' := \begin{bmatrix} \boldsymbol{0}^{m_q \times n}, & \tilde{\mu}_{i,1}\tilde{\boldsymbol{g}}_{p'}, & \ldots, & \tilde{\mu}_{i,\ell'}\tilde{\boldsymbol{g}}_{p'} \end{bmatrix}$
Let $\tilde{\boldsymbol{H}}_i' := ((\bar{q} \bmod p) \cdot \boldsymbol{H}_i', \boldsymbol{0})$
$\tilde{C}_i := \begin{bmatrix} \tilde{\boldsymbol{A}}_i & \tilde{\boldsymbol{A}}_i \odot \tilde{\boldsymbol{S}} + \tilde{\boldsymbol{E}}_i \end{bmatrix} + \tilde{\boldsymbol{H}}_i'$
return $\tilde{C}_i$
```

Algorithm 3 HGSW.Add($\{\tilde{C}_i\}_{i \in [m]}, \{a_i\}_{i \in [m]}$)

```
Require: $\{\tilde{C}_i\}_{i \in [m]}, \{a_i\}_{i \in [m]} \in \mathbb{F}^m$
Ensure: $\boldsymbol{c}^{*'} \in R_{q'}^{n+\ell'}$
$\tilde{\boldsymbol{c}}^* := (0, 0)^{n+\ell'}$
for $j \in [m/v]$ do
    for $i \in [v]$ do
        $\hat{a}_i \leftarrow \mathsf{CRTEncode}(a_{jv+i}, a_{jv+i}, \ldots, a_{jv+i})$
        $\hat{a}_{i,rand}^T \leftarrow \boldsymbol{g}_{\mathrm{rand}}^{-1}(\hat{a}_i)$
        Let $\tilde{a}_{i,rand}^T := \mathsf{NTT}(\hat{a}_{i,rand}^T)$
        $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}^* + \tilde{a}_{i,rand}^T \odot \tilde{C}_{jv+i}$
    end for
end for
Let $\boldsymbol{c}^* := \mathsf{NTT}^{-1}(\tilde{\boldsymbol{c}}^*)$
for $j \in [m/v]$ do
    $\boldsymbol{y}_j \leftarrow \mathcal{D}_r^{\ell'}$
    $\boldsymbol{c}^* := \boldsymbol{c}^* + [\boldsymbol{0}^n, \boldsymbol{y}_j^T]$
end for
$\boldsymbol{c}^{*'} := \mathsf{ModSwitch}(\boldsymbol{c}^*, q, q')$
return $\boldsymbol{c}^{*'} \in R_{q'}^{n+\ell'}$
```

Algorithm 4 HGSW.Decrypt($\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{c}^{*'}$)

```
Require: $\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{c}^{*'} \in R_{q'}^{n+\ell'}$
Ensure: $\boldsymbol{q} \in \mathbb{F}^{4\rho}$
$\bar{\boldsymbol{S}}^T := \begin{bmatrix} -\boldsymbol{S}^T, \boldsymbol{I}_{\ell'}^T \end{bmatrix}$
Let $\tilde{\boldsymbol{c}^*} := \mathsf{NTT}_{R_{q'}}(\boldsymbol{c}^{*'})$
Let $\tilde{\bar{\boldsymbol{S}}} := \mathsf{NTT}_{R_{q'}}(\bar{\boldsymbol{S}})$
$\tilde{\boldsymbol{H}} := \tilde{\boldsymbol{c}^*} \odot \tilde{\bar{\boldsymbol{S}}}$
Let $\bar{\boldsymbol{H}} := \mathsf{NTT}_{R_{q'}}^{-1}(\tilde{\boldsymbol{H}})$
$\bar{\boldsymbol{\mu}} := \lceil (p/q') \cdot \bar{\boldsymbol{H}} \rfloor \in R_p^{\ell'}$
Parse $\bar{\boldsymbol{\mu}} = [\bar{\boldsymbol{\mu}}_1, \bar{\boldsymbol{\mu}}_2]$, where $\bar{\boldsymbol{\mu}}_1 = \boldsymbol{\mu} \in R_p^{\ell}$ and $\bar{\boldsymbol{\mu}}_2 \in R_p^{\tau}$
if $\bar{\boldsymbol{\mu}}_2 \neq \boldsymbol{T}\bar{\boldsymbol{\mu}}_1$ then return $\perp$.
$\boldsymbol{q} \leftarrow \mathsf{CRTDecode}(\bar{\boldsymbol{\mu}}_1, 4\rho)$
return $\boldsymbol{q} \in \mathbb{F}^{4\rho}$
```