Private Re-Randomization for Module LWE and Applications to Quasi-Optimal ZK-SNARKs

Ron Steinfeld[1], Amin Sakzad[1], Muhammed F. Esgin[1,2], and Veronika Kuchta[3]

[1] Faculty of Information Technology, Monash University, Australia
[2] CSIRO's Data61, Australia
[3] Florida Atlantic University, Florida, USA
{ron.steinfeld,amin.sakzad,muhammed.esgin}@monash.edu
and vkuchta@fau.edu

Abstract. We introduce the first candidate lattice-based Designated Verifier (DV) ZK-SNARK protocol with *quasi-optimal proof length* (quasi-linear in the security/privacy parameter), avoiding the use of the exponential smudging technique. Our ZK-SNARK also achieves significant improvements in proof length in practice, with proofs length below 6 KB for 128-bit security/privacy level. Our main technical result is a new regularity theorem 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. To obtain this result, we obtain bounds on the smoothing parameter of an intersection of a random q-ary SIS module lattice, Gadget SIS module lattice, and Gaussian orthogonal module lattice over standard power of 2 cyclotomic rings, and a bound on the minimum of module gadget lattices. We then introduce a new candidate *linear-only* homomorphic encryption scheme called Module Half-GSW (HGSW), which is a variant of the GSW somewhat homomorphic encryption scheme over modules, and apply our regularity theorem to provide smudging-free circuit-private homomorphic linear operations for Module HGSW.

Keywords: Lattice · Zero-Knowledge Proof · Post-Quantum · SNARK

1 Introduction

Zero-knowledge proof (ZKP) systems were introduced by the authors of [36] in 1985 to allow a prover holding some secret witness w for a statement x satisfying some NP relation R, to prove knowledge of w to a verifier (the soundness property), without revealing any information on w to the verifier (the zero-knowledge property) beyond that revealed by the NP statement x known to the verifier. 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 [51] and privacy-preserving cryptocurrencies [9] it is desirable to have *succinct* ZKPs in which the proof (or argument) size is only *polylogarithmic* in the witness size. The first such Zero-Knowledge Succinct Non-interactive ARgument of Knowledge (ZK-SNARK) system for NP languages was proposed by Kilian [40]. While the first ZK-SNARKs were theoretical results and resulted in long proofs in practice, significant practical improvements followed over the last decade [6, 8, 11, 12, 15–17, 24, 25, 30, 37, 51, 57]. The shortest known ZK-SNARK constructions [38] achieve proof lengths in the order of 128 bytes, but rely on quantum-insecure discrete-log assumptions.

Prior work on quantum-safe and lattice-based ZK-SNARKs. ZK-SNARKs based on quantum-safe assumptions from symmetric-key cryptography exist [6, 8, 11, 13] but currently do not achieve proof lengths below around 100KB for typical security parameters. For ZK-SNARKs based on conjectured quantum-safe lattice problems, there are currently two main approaches. The first approach constructs lattice-based ZK-SNARKs that are publicly verifiable but currently yields very long proof sizes in the order of MBs [4][4]. The second approach, which we focus on in this paper, constructs lattice-based *Designated-Verifier* (DV) ZK-SNARKs, 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-SNARKs, proofs can only be verified by the DV holding a secret verification key. DV ZK-SNARKs still suffice for important privacy-preserving applications such as verifiable computation. The first lattice-based DV SNARK following the latter approach was introduced by Boneh et al. [18], and this lattice-based DV SNARK approach was later improved by [31, 39, 50]. The approach in these works constructs a DV SNARK using a cryptographic compiler introduced by Bitansky et al. [17], from two building blocks: (1) a *linear-only* homomorphic vector encryption scheme (i.e. a homomorphic encryption scheme with vector plaintexts where only linear homomorphic operations are computationally feasible) and (2) a Linear Probabilistically Checkable Proof (LPCP) system. Authors of [17] 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-SNARK, i.e. a SNARK satisfying the zero-knowledge privacy property.

The work of [18] 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 [19] and provided a construction for Boolean circuits from a Multi-Prover Interactive Proof (MIP) system.

[4] A very recent concurrent and independent work to ours [14] constructs significantly shorter publicly verifiable lattice-based ZK-SNARKS (LaBRADOR) than prior constructions. However, the LaBRADOR proof lengths reported in [14] for typical security parameters are still an order of magnitude longer than the proof lengths of our designated verifier ZK-SNARKs constructed in this paper for similar security parameters.

The main advantage over the first result in [18], this SNARK 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 SNARKs, but not ZK-SNARKs (i.e. no zero-knowledge property).

Gennaro et al. [31] introduced the first lattice-based construction of SNARKs which also achieved the zero-knowledge property and is built from square span programs (SSP). This paper also provided the first implementation of a lattice-based ZK-SNARK. However, the proof size ranges between 0.5 and 0.9 GB depending on security level. Nitulescu [50] presented a lattice ZK-SNARG from quadratic arithmetic program (QAP), which is defined for arithmetic circuit satisfaction.

The state of the art work on lattice-based DV ZK-SNARKs by authors of [39] (called ISW from hereon) provided a new construction of a shorter ZK-SNARK 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-SNARK 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[5] 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 ZK-SNARK in [39] is a significant reduction of the proof size compared to earlier work in [31].

Lattice ZK-SNARK succinctness problem: smudging-based ZK. However, even with the improvements of the ISW scheme, the resulting ZK-SNARK 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-SNARK proof length in ISW is quadratic in the security parameter λ[6], which is suboptimal, as one could hope to have a proof length quasilinear in λ (i.e. linear up to polylog fac-

[5] 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.

[6] 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.

tors). 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 $20\times$ the ciphertext length of the standard MLWE-based Kyber encryption scheme [20] for typical 128-bit soundness security parameter (the ISW proof length would increase significantly more if we aim for a more standard privacy parameter such as $\kappa = 128$).

A main reason behind the suboptimal proof length of ISW and prior lattice-based ZK-SNARKs is the use of the *exponential smudging* technique to circumvent the difficulty of re-randomizing lattice-based ciphertexts of the underlying linear only encryption scheme E for achieving circuit privacy of the underlying encryption scheme when used inside the Bitansky et al. [17] DV ZK-SNARK compiler. In particular, the first step of the SNARK prover algorithm in ISW and other schemes based on the compiler in [17] 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 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 SNARK 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 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 addition in $e + e'$ is over vectors of integers, so to obtain κ-bit statistical ZK privacy with this smudging method[7], 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-SNARK 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 point of view, can we construct candidate lattice-based ZK-SNARKs with proof length quasilinear in the security parameter $\lambda = \kappa$? From a practical point of view, can we construct candidate ZK-SNARKs 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-SNARK proof length, even by trading off other aspects, such as the size of the common reference string (CRS).

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

Directions and challenges. A first direction towards answering the above open questions was suggested by ISW [39], who asked whether the circuit privacy requirement for the underlying encryption scheme E and its associated smudging technique is really needed for the ZK property of the resulting SNARK constructed with the compiler in [17] from the QAP-based Linear PCP (LPCP) used in [39]. 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 the ISW SNARK. They observed that if this HVZKL property is satisfied for the underlying LPCP, then the ISW SNARK with no smudging achieves ZK. If the latter is true, it would give shorter ZK-SNARK proofs. However, the HVZKL property of the used LPCP was not studied in [39], and it is not clear why it should be true.

A second direction towards the open questions 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 to follow in this direction is to look at circuit privacy techniques developed for lattice-based fully homomorphic encryption (FHE) schemes. The FHE circuit privacy technique devised by Gentry [32] relies on exponential smudging. The later works [21,28] provide FHE circuit privacy without exponential smudging and with small asymptotic overheads. However, trying to adopt the latter circuit privacy techniques to our ZK-SNARK context faces several challenges that require new ideas. The first challenge is that the soundness security of the ZK-SNARK setting crucially requires the underlying encryption scheme to be *linear-only* homomorphic, whereas the circuit-privacy techniques of [21, 28] work for FHE schemes, which would be insecure as linear-only encryption. In particular, the sanitization circuit privacy procedure of [28] crucially relies on the FHE-based bootstrapping procedure. A second challenge is that, to build on the extension-field SNARK succinctness techniques of ISW, we need the underlying encryption scheme to be based on structured MLWE lattices defined over a sufficiently high degree polynomial ring R, whereas the circuit privacy analysis techniques of [21] rely 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 with non-trivial subideals, such as commonly used MLWE rings $R_q = \mathbb{Z}_q[x]/(x^d + 1)$ for a power-of-2 d.

1.1 Our Contributions

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

Negative result: attack on zero-knowledge property of the short SNARK in [39] with no smudging. Our first contribution gives a negative result in the first direction mentioned above. In particular, we present a simple attack (based on natural heuristic assumptions) on the Zero-Knowledge property of

SNAR(G/K) scheme	Base encryption scheme	Quasi-opt. proof size (Yes/No)	Size		ZK property (Yes/No)	ZK technique
			CRS (GB)	Proof (KB)		
BISW17 [18]	LWE-Regev	Yes	-	-	No	N/A
BISW18 [19]	RLWE-Regev	Yes	-	-	No	N/A
GMNO18 [31]	LWE-Regev	No	?*	?*	Yes	exp. smudging
ISW21 [39]	MLWE-Regev	No	$(10, 337)$	33	Yes	exp. smudging
Our work	MLWE-HGSW	Yes	$(32, 410)$	6	Yes	poly. rerandom.

Table 1. Comparison of lattice (ZK-)SNAR(G/K)s for R1CS size of $N_g = 2^{20}$ for both soundness and zero-knowledge (if applicable) security level at 128 bits. Here, we estimated the sizes for the ISW21 protocol [39] for $\kappa = 128$ bit ZK privacy level, by extending the 'Shorter Proofs' parameters in [39] 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. 6.2 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. *We note that [39] pointed out that the suggested parameters in [31] provide only 15 bits of provable soundness. Therefore, in our table, we skip the proof and CRS sizes for [31].

SNARK in [39] with no smudging when it is instantiated with the QAP-based Linear PCP presented in [39] (which in turn is a variant of the LPCPs given in [10, 30]). As mentioned above, the latter ZK property of the SNARK in [39] with no smudging is inherited from the assumed Honest Verifier ZK with Leakage (HVZKL) property of the underlying Linear PCP. Our attack in fact shows that the underlying LPCP introduced in [10, 30] does not in general satisfy the HVZKL property. We do so by exhibiting an R1CS relation $\mathcal{R}$ and a statement x with two distinct witnesses w, w', such that the leakage information provided to the HVZKL distinguisher against the LPCP suffices to distinguish the two witnesses with an $O(1)$ distinguishing advantage (under natural heuristic assumptions). Although our attack applies to a specific relation, it demonstrates that the HVZKL property assumed in [39] does not hold in general for existing LPCPs, and therefore, that with such LPCPs, the exponential smudging technique is in general *necessary* for the ZK property of the ISW ZK-SNARK construction, which leads to non-succinct proofs of length $\Omega(\lambda^2)$. To bypass this issue, our ZK-SNARK does not rely on HVZKL of the LPCP.

Positive result: new candidate lattice-based ZK-SNARK with quasi-optimal succinct proofs. Our main result addresses the second direction discussed above. We construct the first candidate lattice-based ZK-SNARKs with quasi-optimal succinct proofs, namely proof length quasi-linear in the security parameter λ. Our new ZK-SNARK avoids the use of exponential smudging for achieving the ZK property with its inherent inefficient parameters. Following the second direction discussed above, we address the technical challenges of constructing a linear-only 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-SNARK construction. Our encryption scheme E is derived from a suitable modification of a Module LWE variant of the GSW homomorphic encryption scheme [35]. We compare the main properties of our ZK-SNARK to prior lattice-based SNARK constructions in Table 1. Our lattice-based ZK-SNARKs not only achieve the shortest asymptotic proof length to date in theory but also in practice, with substantial savings of more than $5\times$ in proof length versus ISW21 at the same security/privacy level of 128-bit (for lower statistical privacy parameters, e.g. $\kappa = 40$ as used in the concrete parameters suggested in [39], our proof length improvement is smaller but still significant; we refer to Sec. 6.2 for details). As noted above, our work focuses on optimization for the minimum possible proof length, and this comes at the tradeoff of a large CRS size compared to the ISW21 protocol. The main reason for the CRS overhead in our protocol versus ISW21 is our GSW-like base encryption scheme, which typically consists of several tens of Regev ciphertexts used in the ISW21 protocol.

To obtain our main result, we introduce new tools described in the following.

New regularity results for private re-randomization of MLWE samples. Our main technical contribution is a new regularity theorem for 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, which can be applied for circuit-private homomorphic scaling of Module GSW ciphertexts. Our regularity result applies to the following type of 'gadget-based' re-randomization of a set of MLWE samples. Let $\boldsymbol{G}$ denote a 'power of 2' gadget matrix [47]. Take a set of MLWE samples $(\boldsymbol{A}, \boldsymbol{B})$ with $\boldsymbol{B} = \boldsymbol{AS} + \boldsymbol{E}$ over the polynomial ring $R_q := \mathbb{Z}_q[x]/(x^d + 1)$, 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 $(\boldsymbol{u}^T, \boldsymbol{v}^T) := (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T \cdot \boldsymbol{B} + \boldsymbol{y}^T) = (\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T \cdot \boldsymbol{AS} + \boldsymbol{x}^T\boldsymbol{E} + \boldsymbol{y}^T)$, where $\boldsymbol{x}$ is a re-randomization vector sampled from a discrete Gaussian distribution with small 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 $r = \mathsf{poly}(\kappa)$, avoiding the exponential blowup ($r = 2^{\Omega(\kappa)}$) in smudging based re-randomization results as used in [39]. We, therefore, obtain a Module LWE analogue of the regularity theorem for private re-randomization of (unstructured) LWE samples in [21]. The latter LWE-based regularity result over $\mathbb{Z}_q$ uses general leftover hash lemmas over fields and does not extend to MLWE over rings R_q with non-trivial subideals. Technically, our regularity proof for

MLWE requires different and more involved lattice smoothing-based techniques (see technical overview) to deal with this issue.

Half GSW and application to ZK-SNARKs. We present a new candidate linear-only 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 somewhat 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-SNARK 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 $\lambda = \kappa$. Like previous candidate linear-only lattice-based encryption schemes (e.g. [18, 19, 39]), the linear-only property of our HGSW scheme relies on a plausible conjecture that we call 'HGSW linear-only'. Like previous such assumptions, 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 that 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. Thus the HGSW construction itself is not new, and indeed such an encryption scheme has been used in other contexts, e.g. [33]. However, to our knowledge, our work is the first application of such an encryption scheme in the context of *linear-only* encryption.

1.2 Overview of Techniques

Regularity theorem for re-randomization of MLWE samples. The core part of our regularity theorem (given in Lemma 8 in Sec. 4), is a 'leftover hash lemma with leakage' for MLWE re-randomization. Given a uniformly random MLWE matrix $\boldsymbol{A}$ over $R_q := \mathbb{Z}_q[x]/(x^d + 1)$ for d a power of 2, and randomization vectors $\boldsymbol{x}^T$ and $\boldsymbol{y}^T$ sampled from a discrete Gaussian of width parameter r satisfying $\boldsymbol{x}^T \boldsymbol{G} = \boldsymbol{a}^T$ for some scaling vector $a \in R_q$ and a Gaussian distributed error matrix $\boldsymbol{E}$, it shows that the re-randomized vector $\boldsymbol{x}^T \boldsymbol{A} + \boldsymbol{y}^T \bmod q$ is statistically close to a uniform vector over R_q over the randomness of $\boldsymbol{x}, \boldsymbol{y}$, even conditioned on the leakage on $\boldsymbol{x}$ given by $\boldsymbol{x}^T \boldsymbol{E} + \boldsymbol{y}^T$. Due to the splitting structure of the ring R_q, our proof uses a completely different approach to the one used in the proof of the unstructured LHL with leakage over the field $\mathbb{Z}_q$ in [21]. In the latter field case, the authors apply in this core part the 'generalized leftover hash Lemma' of [27] (Lemma 3.5 in [21]), which states that $\boldsymbol{x}^T \boldsymbol{A}$ is statistically close to uniform over $\mathbb{Z}_q$ conditioned on arbitrary bounded leakage on $\boldsymbol{x}^T$, as long as the distribution of $\boldsymbol{x}^T$ has enough min-entropy. However, this general min-entropy based leftover hash fails in general when the field $\mathbb{Z}_q$ is replaced by a ring which has non-trivial subideals, such as our power of 2 ring R_q. Instead,

to derive our LHL with leakage over R_q, we use a different approach based on a rather involved and delicate combination of lattice Discrete Gaussian smoothing arguments that exploits the discrete Gaussian distribution of $\boldsymbol{x}^T$ over a coset $\boldsymbol{c}^T + \Lambda_q^{\perp}(\boldsymbol{G})$ of the Gadget perp lattice $\Lambda_q^{\perp}(\boldsymbol{G}) := \{\boldsymbol{v} : \boldsymbol{v}^T\boldsymbol{G} = \boldsymbol{0}\}$ (where $\boldsymbol{c}$ satisfies $\boldsymbol{c}^T\boldsymbol{G} = \boldsymbol{a}$). In particular, as a core technique underlying our LHL with leakage, which may be of independent interest, we study the smoothing parameter of the intersection of the three underlying perp lattices associated with the matrices $\boldsymbol{A}, \boldsymbol{E}, \boldsymbol{G}$, i.e. the lattice $\Lambda_q^{\perp}(\boldsymbol{A}) \cap \Lambda^{\perp}(\boldsymbol{E}) \cap \Lambda_q^{\perp}(\boldsymbol{G})$, where the $\Lambda^{\perp}(\boldsymbol{E})$ orthogonal lattice is defined over R and not mod q[8]. In bounding the intersection lattice smoothing parameter, 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 previously and may be of independent interest for other cryptographic applications. We remark that although prior work on lattice discrete-Gaussian smoothing-based regularity bounds has studied the distribution of $\boldsymbol{x}^T\boldsymbol{A}$ (e.g. [45,54,55]) or $\boldsymbol{x}^T\boldsymbol{E}$ (e.g. [1,2,41]) in various settings, the leftover hash with *leakage* over modules, which we address here, is to our knowledge a novel result. The work in [43] analyzes the distribution of $\boldsymbol{x}$ conditioned on $(\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{E})$ in the unstructured case, rather than the joint distribution of $(\boldsymbol{x}^T\boldsymbol{A}, \boldsymbol{x}^T\boldsymbol{E})$. The recent work of Dachman-Soled et al. [26] gives three results on different special conditional distributions over rings than we deal with in this paper.

Half GSW linear-only encryption scheme. Our new succinct and circuit-private linear-only encryption scheme Half GSW is a variant of the GSW somewhat homomorphic encryption scheme [35] over modules. We recall that in the (module version of) GSW scheme with secret key $\mathsf{sk} = \boldsymbol{s}$ the ciphertext of a message μ (typically a bit), has the form[9]

$$\begin{bmatrix}\boldsymbol{C}_1\\ \boldsymbol{C}_2\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}' & \boldsymbol{A}'\boldsymbol{s} + \boldsymbol{e}'\\ \boldsymbol{A} & \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}\end{bmatrix} + \mu \begin{bmatrix}\boldsymbol{g} & \boldsymbol{0}_{m_q \times 1}\\ \boldsymbol{0}_{m_q \times 1} & \boldsymbol{g}\end{bmatrix},$$

with uniformly random LWE matrices $(\boldsymbol{A}', \boldsymbol{A}) \hookleftarrow \mathcal{U}(R_q^{m_q \times 1} \times R_q^{m_q \times 1})$, short error vectors $(\boldsymbol{e}', \boldsymbol{e})$, and a power-of-2 gadget vector $\boldsymbol{g}^T = (1, 2, \ldots, 2^{m_q - 1})$, where $m_q = \log_2 q$. The GSW scheme enjoys both additive and multiplicative homomorphisms. It also enjoys circuit-privacy with succinct ciphertexts [21]. However, the multiplicative homomorphism is undesirable for linear-only encryption and its applications. Our first and main observation towards our HGSW scheme is that if we remove the top half ($\boldsymbol{C}_1$) of the GSW ciphertext, the GSW multiplicative homomorphism algorithm no longer applies (and it seems difficult to find a different algorithm for multiplicative homomorphism), but the

[8] We remark that this discussion is a slight simplification; the orthogonal lattice related to $\boldsymbol{E}$ that we study is actually $\Lambda^{\perp}(\bar{\boldsymbol{E}})$, where $\bar{\boldsymbol{E}}$ is an extension of $\boldsymbol{E}$ with an appended identity matrix, to account for the added error vector $\boldsymbol{y}^T$.

[9] We show here for simplicity the case where the MLWE secret $\boldsymbol{s}$ is a single ring element (i.e. GSW based on rank 1 MLWE); for GSW based on rank n MLWE, the top GSW ciphertext part $\boldsymbol{C}_1$ consists of nm_q rows, and in that case our HGSW ciphertext $\boldsymbol{C}_2$ consists of only $1/(n+1)$ of the GSW ciphertext, rather than $1/2$.

linear homomorphism and private homomorphic scaling with succinct ciphertexts property is preserved, allowing us to apply our re-randomization result. Indeed, to privately perform homomorphic scaling by a scaling factor a, we can sample a discrete Gaussian matrix $\boldsymbol{X}$ such that $\boldsymbol{X}\boldsymbol{g} = a\boldsymbol{g}$ and compute $\boldsymbol{C}'_2 = \boldsymbol{X}\boldsymbol{C}_2 = [\boldsymbol{B}, \boldsymbol{B}\boldsymbol{s}+\boldsymbol{e}'+a\mu\boldsymbol{g}]$, where the distribution of $(\boldsymbol{B} = \boldsymbol{X}\boldsymbol{A}, \boldsymbol{e}' = \boldsymbol{X}\boldsymbol{e})$ is independent of a by our re-randomization result. However, this latter linear homomorphic scheme (when considered over the plaintext space R_p for some p smaller than q) still suffers from a circuit privacy problem: a homomorphic sum of m such scaled ciphertexts with messages μ_i and scaling coefficients a_i (for $i \in [m]$) has the form $[\boldsymbol{B}, \boldsymbol{B}\boldsymbol{s} + \boldsymbol{e}' + (\sum_i a_i\mu_i)\boldsymbol{g}]$, and in general leaks information (to an attacker with the secret key $\boldsymbol{s}$) about $\sum_i a_i\mu_i$ in R_q, rather than only revealing the desired value $\sum_i a_i\mu_i \bmod p$ in the plaintext space R_p. To address this problem, we use a modulus q that is a multiple of p, and premultiply the message by $\frac{q}{p}$[10]. We extend the scheme to a vector encryption scheme using the standard randomness sharing technique [53], and reduce scaled ciphertexts to succinct Regev ciphertexts by using just a single row vector $\boldsymbol{x}^T$ to perform homomorphic scaling.

1.3 Roadmap

In section 2, we provide the main preliminaries on lattices. Further definitions and syntax for vector encryptions and linear PCPs are provided in the supplementary material. We present an attack on the zero-knowledge property of ISW construction [39] in Section 3. In Section 4, we give our main result of the paper comprising the new results for a private re-randomization technique of MLWE samples. Our construction of the module-based Half-GSW (HGSW) scheme and its correctness and security analysis are provided in Section 5. In section 6, we show how to apply our new HGSW to construct a lattice-based ZK-SNARK. Finally, in section 7, we analyse and provide optimal and concrete parameters for our HGSW and ZK-SNARK construction. All the missing proofs are given 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 $\{1, \ldots, n\}$ 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 transposition and inversion operations of a matrix $\boldsymbol{M}$ are written as $\boldsymbol{M}^T$ and $\boldsymbol{M}^{-1}$, respectively. For a distribution $\mathcal{D}$, we

[10] We remark that using the rounded value of q/p in case q is not divisible by p does not fix the circuit privacy problem, as the rounding error will cause a circuit privacy leakage.

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.

2.1 Lattice Preliminaries

Lattices. A n-dimensional lattice Λ is a discrete additive subgroup of $\mathbb{R}^n$. For an integer $t \leq n$ 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 $\lambda_i(\Lambda) := \inf\{r : \dim(\mathsf{Span}(\Lambda \cap B(r))) \geq i\}$, where $B(r)$ denotes the closed zero-centered Euclidean ball of radius r.

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

$$\Lambda_q^{\perp}(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^l | \boldsymbol{x}^T\boldsymbol{A} = \boldsymbol{0} \bmod q\},$$
$$\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^l | \boldsymbol{v} = \boldsymbol{As} \bmod q, \text{ for some } \boldsymbol{s} \in \mathbb{Z}^n\}.$$

The two q-ary lattices for a matrix $\boldsymbol{A}$ satisfy the following duality relation: $\Lambda_q^{\perp}(\boldsymbol{A})^* = \frac{1}{q}\Lambda_q(\boldsymbol{A})$. 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 $\mathrm{rot}(c) = \mathrm{rot}(a) \cdot \mathrm{rot}(b)$, where for a ring element a, $\mathrm{rot}(a) \in \mathbb{Z}^{d \times d}$ 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 $\mathrm{rot}(\boldsymbol{A}) \in \mathbb{Z}^{ld \times nd}$ 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 $\mathrm{rot}(\boldsymbol{A})$ to define the ld-dimensional q-ary lattices $\Lambda_q^{\perp}(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{Z}^{ld} | \boldsymbol{x}^T\mathrm{rot}(\boldsymbol{A}) = \boldsymbol{0} \bmod q\}$ and $\Lambda_q(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^{ld} | \boldsymbol{v} = \mathrm{rot}(\boldsymbol{A})\boldsymbol{s} \bmod q, \text{ for some } \boldsymbol{s} \in \mathbb{Z}^{nd}\}$ over $\mathbb{Z}$. The lattice $\Lambda_q(\boldsymbol{A})$ is related to the lattice $\widetilde{\Lambda_q}(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{Z}^{ld} | \boldsymbol{v} \text{ is the coeff. vector of } \boldsymbol{As} \bmod q, \text{ for some } \boldsymbol{s} \in R^n\}$ by a full rank and norm-preserving linear transformation (this is the 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 $\Lambda_q(\boldsymbol{A})$.

Definition 2 (Module Learning With Errors (MLWE) [42]). *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*

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

Definition 3 (Gaussian Function). *For any* $r > 0$ *the Gaussian function with parameter* r[11] *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 $\rho_r(\Lambda+\boldsymbol{c}) = \sum_{\boldsymbol{x}\in\Lambda+\boldsymbol{c}} \rho_r(\boldsymbol{x})$.

Definition 4 (The g_{rand}^{-1} Algorithm [47]). *Let* $\beta, q \in \mathbb{Z}$, *with* $\beta \geq 2$, $\boldsymbol{g}^T = (1, \beta, \beta^2, \ldots, \beta^{m_q-1}) \in R^{1\times m_q}$, *where* $m_q = \lceil \log_\beta q \rceil$. *There is a randomized, efficiently computable function* $\boldsymbol{g}_{\mathsf{rand}}^{-1}(\cdot) : R_q \to R^{1\times m_q}$ *such that* $\boldsymbol{x}^T \leftarrow \boldsymbol{g}_{\mathsf{rand}}^{-1}(a)$ *is sampled from a discrete Gaussian distribution with parameter* r, *such that* $\boldsymbol{x}^T\boldsymbol{g} = a \mod q$ *(i.e.* $\boldsymbol{x} \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\boldsymbol{g})+\boldsymbol{c},r}$, *where* $\boldsymbol{c}$ *is fixed vector satisfying* $\boldsymbol{c}^T\boldsymbol{g} = a \mod q$*). Note that the output of* $\boldsymbol{g}_{\mathsf{rand}}^{-1}$ *is always a row vector. Furthermore, let* $\boldsymbol{G} = \boldsymbol{g} \otimes I_\rho \in R_q^{L\times\rho}$ *with* $\rho := L/m_q$. *For a vector* $\boldsymbol{a} \in R_q^n$, *the* $\boldsymbol{x}^T = \boldsymbol{g}_{\mathsf{rand}}^{-1}(\boldsymbol{a})$ *satisfy* $\boldsymbol{x}^T\boldsymbol{G} = \boldsymbol{a} \mod q$, *where* $\boldsymbol{x} \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\boldsymbol{G})+\boldsymbol{c},r}$ *and* $\boldsymbol{c}$ *is fixed vector satisfying* $\boldsymbol{c}^T\boldsymbol{G} = \boldsymbol{a} \mod q$.

Definition 5 (Smoothing Parameter [48]). *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* $\rho_{1/r}(\Lambda^* \setminus \{\boldsymbol{0}\}) \leq \epsilon$.

Lemma 1 ([44], 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 2 ([48], 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 3 ([44], 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 4 ([34, 48]). *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 5 (Banasczcyk [7]). *For any rank−*n *lattice* $\Lambda \subset \mathbb{R}^m$ *and for all* $i \in [n]$, *we have* $1 \leq \lambda_i(\Lambda)\cdot\lambda_{n-i+1}(\Lambda^*) \leq n$.

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

Lemma 6 ([52]). *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 7 ([41], Lemma 7). *For any rank-d lattice Λ and $\epsilon \in (0, 1/2)$, we have $\lambda_d(\Lambda)/\sqrt{d} \leq \eta_\epsilon(\Lambda) \leq \lambda_d(\Lambda) \cdot \sqrt{\ln(2d(1+\epsilon^{-1}))/\pi}$.*

3 Attack on Zero-Knowledge Property of ISW Construction [39] with no Smudging

For definitions of HVZK and linear PCP and other related syntax, please refer to Appendix A. We now present an attack on the zero-knowledge property of the variant of the ZK-SNARK in [39] which uses no smudging. We first recall the construction of linear-only vector encryption from [39].

Description of attack. Let κ denote the zero-knowledge parameter as it is used in [39]. For $\kappa = 0$, we present an attack against the δ-HVZK with $(\mathcal{D}, q)$ leakage property defined in Definition 14 for the linear PCP Π_{LPCP} used in the lattice-based ZK-SNARK instantiation in [39], where $\mathcal{D}$ is the distribution over matrices over $R := \mathbb{Z}[x]/(x^d + 1)$ defined as follows (see Lemma 3.26 in [39]):

- Sample $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{n \times m_q})$ and $\boldsymbol{E} \hookleftarrow \mathcal{D}_{R,s}^{n \times \ell}$.
- Output the matrix $\boldsymbol{Z} = [\boldsymbol{A}, \boldsymbol{E}] \in R^{n \times (m_q + \ell)}$.

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 $\mathcal{CS}_N$ such that there exists 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 $\boldsymbol{Z} \hookleftarrow \mathcal{D}$, $\boldsymbol{b} = \boldsymbol{Z} \cdot \boldsymbol{\pi} \in R_q^n$, $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(\boldsymbol{x}, \boldsymbol{w})$, and $\boldsymbol{b}' = \boldsymbol{Z} \cdot \boldsymbol{\pi}' \in R_q^n$, $\boldsymbol{\pi}' \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(\boldsymbol{x}, \boldsymbol{w}')$.

We focus here on the bottom ℓ coordinates $\tilde{\boldsymbol{b}} = \boldsymbol{E}\boldsymbol{\pi}, \tilde{\boldsymbol{b}}' = \boldsymbol{E}\boldsymbol{\pi}'$ of $\boldsymbol{b}, \boldsymbol{b}'$ respectively. Due to the correctness of decryption, the parameters are chosen such that $\|\tilde{\boldsymbol{b}}\|_\infty, \|\tilde{\boldsymbol{b}}'\|_\infty < q/2$, so the computation is over R (without any modulus reduction). Notice that according to Definition 9, $\boldsymbol{\pi}$ has the form $\boldsymbol{\pi}^T = (\delta_1, \delta_2, \delta_3, \boldsymbol{h}^T, \bar{\boldsymbol{w}}^T) \in R_p^{4+N_g+N_w-n}$, 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 Definition 9). 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 we can write $\tilde{\boldsymbol{b}} = \boldsymbol{E}_1\boldsymbol{r} + \boldsymbol{E}_2\bar{\boldsymbol{w}}$, where the first term is a random 'error' term, while the second term is a known 'shift' (respectively, $\tilde{\boldsymbol{b}}' = \boldsymbol{E}_1\boldsymbol{r}' + \boldsymbol{E}_2\bar{\boldsymbol{w}}'$). Now, each integer coefficient of the 'error' term $\boldsymbol{E}_1\boldsymbol{r}$ (resp. $\boldsymbol{E}_1\boldsymbol{r}'$) is an inner-product between a (pseudo) uniformly random $\boldsymbol{r}$ with coefficients uniformly random in $(-p/2, p/2)$ and hence of standard deviation $p/\sqrt{12}$, and a row 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 $\sigma_b := sp\sqrt{(N_g+4)/12}$ for both $\tilde{\boldsymbol{b}}$ and $\tilde{\boldsymbol{b}}'$. Let $\left(\boldsymbol{e}_2^{(i)}\right)^T$ denote the i'th row of $\boldsymbol{E}_2$. For $\boldsymbol{b} \in \{\tilde{\boldsymbol{b}}, \tilde{\boldsymbol{b}}'\}$, conditioned on $\boldsymbol{E}$, the distribution of $y^{(i)} := b^{(i)} - \left(\boldsymbol{e}_2^{(i)}\right)^T\bar{\boldsymbol{w}}$ 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 $c_b^{(i)} := \left(\boldsymbol{e}_2^{(i)}\right)^T \cdot (\bar{\boldsymbol{w}}' - \bar{\boldsymbol{w}})$. The i'th coordinate distinguisher, therefore, 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 $\left(\boldsymbol{e}_2^{(i)}\right)^T$ 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

$$\Delta^{(i)} \approx \mathrm{erf}\left(\frac{\bar{c}_b^{(i)}}{2\sqrt{2}\sigma_b}\right) = \mathrm{erf}\left(\frac{\sqrt{6}\|\bar{\boldsymbol{w}}' - \bar{\boldsymbol{w}}\|}{2\pi p\sqrt{N_g+4}}\right). \tag{1}$$

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:

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

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 [30]. 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 $\bar{\boldsymbol{w}}_{\text{R1CS}}^T = ((x,0,\ldots,0),(0,0,\ldots,0),1)$ and $\bar{\boldsymbol{w}}'_{\text{R1CS}}{}^T = ((x,x,\ldots,x),(0,0,\ldots,0),1)$ and hence $\|\bar{\boldsymbol{w}}_{\text{R1CS}} - \bar{\boldsymbol{w}}'_{\text{R1CS}}\| = x\sqrt{N-1}$. Plugging in (1) with $N_g := N \to \infty$ and $x \geq p/c$ for some $c = O(1)$, one can see that the expected distinguishing advantage $\Delta^{(i)} \approx \text{erf}\left(\frac{\sqrt{6}}{2\pi c}\right) = \Omega(1)$ is non-negligible.

4 New Regularity Results for Private Re-randomization of MLWE Samples

Our main regularity result on private re-randomization MLWE samples is the following. Looking ahead, 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 the below Theorem 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 the below Theorem).

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 $L,\nu \in \mathbb{Z}$, $0 < \epsilon \leq 1/2$, $\boldsymbol{G} = \boldsymbol{g}\otimes I_\nu \in R_q^{L\times\nu}$ with $\boldsymbol{g} = (1,\beta,\ldots,\beta^{m_q-1})$, $\boldsymbol{c} \in R_q^L$ be arbitrary, $m_q := \lceil \log_\beta(q)\rceil$ and $\nu := L/m_q \geq 1$. 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*

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

and

$$r \geq \max\left((L+\ell')dc\sqrt{\ln(2Ld(1+\epsilon^{-1}))/\pi}, r_G\right) \text{ and } \ell' d 2^{-Ld} \leq \epsilon, \quad (3)$$

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

$$p_1(c_1) := \sum_{r=0}^{\ell_q-1} \frac{\binom{\ell_q}{r}\bar{q}^{((1-r/\ell_q)n+\nu+\ell')d}\cdot(2\bar{q}^{1-r/\ell_q}/c_1+1)^{Ld}}{\bar{q}^{Ld(1-r/\ell_q)}} \leq \epsilon \quad (4)$$

and

$$c_2 := 2\beta^2, \text{ and } Ld2^{-\ell' d} \leq \epsilon, \tag{5}$$

and

$$c_3 := 4\beta^2 s\sqrt{d\ell'/(2\pi)} \text{ with } 4s^2 d\ell' \geq 2\pi, \tag{6}$$

and

$$r_G := \left(\sqrt{m_q}(\beta-1) + \sqrt{\ell' d}((m_q-1)(\beta-1)+1)E_\infty\right) \cdot \sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}} \tag{7}$$

then

$$\Delta\left((\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}} \bmod q, \boldsymbol{A}, \boldsymbol{E}), (\mathcal{U}(R_q^n), \mathcal{D}_{\mathbb{Z}^{\ell' d}, r\cdot \mathrm{rot}(\bar{\boldsymbol{E}})}, \boldsymbol{A}, \boldsymbol{E}\right) \leq 24\epsilon.$$

The proof of Theorem 1 consists of two parts presented in the following two subsections. The first part (Part 1: LHL over R_q with leakage), given in Lemma 8, shows that the re-randomized matrix $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{A}} \bmod q$ 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 $\bar{\boldsymbol{x}}\bar{\boldsymbol{E}}$. The second part (Part 2: Gaussian LHL), given in Corollary 1, shows that the distribution of the leakage component $\boldsymbol{x}^T\bar{\boldsymbol{E}}$ is statistically close to a skewed discrete Gaussian. The proof is an adaptation of Lemma 3.6 of [21] in the unstructured lattice case (which itself is an adaptation of the result of [1]) to the structured module case, which requires also handling higher dimensional skewed (non-spherical) Gaussian distributions as opposed to the spherical distributions considered in [21].

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-SNARK constructions.

Asymptotic Parameter Setting. We give sample asymptotic parameter settings for Theorem 1 to show how it can be instantiated with parameters $q, r, L = \mathrm{poly}(\kappa)$ 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 (4) 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 (2) 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 $q = \widetilde{O}(\kappa^2)$.

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

The LHL Lemma used in [21] (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 and lemmas 2-7.

Lemma 8 (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 $L, \nu \in \mathbb{Z}$, $0 < \epsilon \leq 1/2$, $\boldsymbol{G} = \boldsymbol{g} \otimes I_\nu \in R_q^{L\times\nu}$ with $\boldsymbol{g} = (1, \beta, \ldots, \beta^{m_q-1})$, $m_q := \lceil \log_\beta(q) \rceil$ and $\nu := L/m_q$. 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*

$$q > \max\left(2(L+\ell')dcp\sqrt{1+4s^2Ld/(2\pi)}, 2r(s\sqrt{Ld}+1)\sqrt{\ln(2\ell' d/\epsilon)/\pi}\right) \tag{8}$$

and

$$r \geq (L+\ell')dc\sqrt{\ln(2Ld(1+\epsilon^{-1}))/\pi} \text{ and } \ell' d 2^{-Ld} \leq \epsilon, \tag{9}$$

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

$$p_1(c_1) := \sum_{r=0}^{\ell_q-1} \frac{\binom{\ell_q}{r}\bar{q}^{((1-r/\ell_q)n+\nu+\ell')d} \cdot (2\bar{q}^{1-r/\ell_q}/c_1+1)^{Ld}}{\bar{q}^{Ld(1-r/\ell_q)}} \leq \epsilon \tag{10}$$

and

$$c_2 := 2\beta^2, \text{ and } Ld2^{-\ell' d} \leq \epsilon, \tag{11}$$

and

$$c_3 := 4\beta^2 s\sqrt{d\ell'/(2\pi)} \text{ with } 4s^2 d\ell' \geq 2\pi \tag{12}$$

then

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

Proof. Let $\bar{\boldsymbol{x}}^T := (\boldsymbol{x}^T, \boldsymbol{y}^T) \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\boldsymbol{G})+\boldsymbol{c},r} \times \mathcal{D}_{R^{\ell'},r} := \mathcal{D}_{\Lambda_q^\perp(\bar{\boldsymbol{G}})+\bar{\boldsymbol{c}},r}$, 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 (2), we have $\|\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}}\|_\infty < q/2$ (i.e. no wraparound mod q in $\bar{\boldsymbol{x}}^T\bar{\boldsymbol{E}} \bmod q$) 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 2, 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 (9), 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 6 and Lemma 11 below, and that $c \geq c_2 \geq \beta^2$. Therefore, by Lemma 3, 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 $\|\boldsymbol{x}^T\boldsymbol{E}\|_\infty \le \sqrt{\ln(2\ell' d/\epsilon)/\pi} sr\sqrt{Ld}$, except with probability $\le 4\epsilon$. Similarly using Lemma 2 and the union bound, we have $\|\boldsymbol{y}\|_\infty \le \sqrt{\ln(2\ell' d/\epsilon)/\pi} r$, except with probability $\le \epsilon$. 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 $\le 5\epsilon$, where the last inequality is due to the second part of (8). The claimed bound of the Lemma therefore follows if we show that

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

(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 $\le 3\epsilon$ 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 $\le 8\epsilon$ 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:

$$P_{\hat{\boldsymbol{e}}}(\hat{\boldsymbol{v}}) = \frac{\Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\bar{\boldsymbol{G}})+\bar{\boldsymbol{c}}^T, r}}[\bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{A}} \bmod q, \bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{E}}) = (\hat{\boldsymbol{v}}^T, \hat{\boldsymbol{e}}^T)]}{\Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\bar{\boldsymbol{G}})+\bar{\boldsymbol{c}}^T, r}}[\bar{\boldsymbol{x}}^T \cdot \bar{\boldsymbol{E}} = \hat{\boldsymbol{e}}^T]}. \tag{13}$$

The numerator p_n of (13) has the form

$$p_n := \Pr_{\bar{\boldsymbol{x}}^T \hookleftarrow D_{\Lambda_q^\perp(\bar{\boldsymbol{G}})+\bar{\boldsymbol{c}}^T, r}}[\bar{\boldsymbol{x}}^T \in (\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}_0}^T + \Lambda^\perp(\bar{\boldsymbol{E}}))] \tag{14}$$

$$= \frac{\rho_r((\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda^\perp(\bar{\boldsymbol{E}})) \cap (\bar{\boldsymbol{c}} + \Lambda_q^\perp(\bar{\boldsymbol{G}})))}{\rho_r(\bar{\boldsymbol{c}} + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{15}$$

$$= \frac{\rho_r((\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda^\perp(\bar{\boldsymbol{E}})) \cap (\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}})))}{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{16}$$

$$= \frac{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}}))}{\rho_r(\bar{\boldsymbol{x}}_0^T + \Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{17}$$

$$\in (1 \pm 4\epsilon) \cdot \frac{\rho_r(\Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}}))}{\rho_r(\Lambda_q^\perp(\bar{\boldsymbol{G}}))} \tag{18}$$

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 4 twice, assuming that

$$r \ge \eta', \text{ where } \eta' := \eta_\epsilon(\Lambda_q^\perp(\bar{\boldsymbol{A}}) \cap \Lambda^\perp(\bar{\boldsymbol{E}}) \cap \Lambda_q^\perp(\bar{\boldsymbol{G}})), \tag{19}$$

and 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 6, 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 5 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 $\mathrm{rot}(\bar{\boldsymbol{E}})$; 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 10, 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 10): 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 [49, 55] for lower bounding the minimum of Module SIS lattices. In particular, Lemma 10 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 12): 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 11) is lower bounded by $\bar{q}/\beta^2$. The above arguments are made precise in Lemma 10, which shows that

in this subcase, we get $\|\boldsymbol{w}\| \geq \bar{q}/c_2$ for a 'small' $c_2 = 2\beta^2$, 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 1 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 (8), 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 (18), 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.

To complete the proof of Lemma 8, it remains to prove Lemma 10 for the Subcase 1 and Lemma 12 for the Subcase 2.

Subcase 1 of LHL with Leakage (Lemma 8) We will use the following Lemma lower bounding the minimum of ideal lattices (see Cor. 1 in [29]).

Lemma 9 (Adapted from Cor. 1.2 of [46]). *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 $Z \subseteq [\ell_q]$ with $|Z| = r$. The $\lambda_1(I_{Z,R_{\bar{q}}})$ over all non-zero vectors of the ideal lattice $I_{Z,R_{\bar{q}}} := \{a \in \mathbb{R} : a \bmod (f^{(u)}), \bar{q}) = 0$ for $j \in Z\}$ is lower bounded as*

$$\lambda_1(I_{Z,R_{\bar{q}}}) \geq \bar{q}^{r/\ell_q}.$$

We now present our subcase 1 of LHL with leakage bound.

Lemma 10. *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

$$\|\boldsymbol{w}\|_2 \geq \bar{q}/c_1 \tag{20}$$

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

$$p_1 \leq \sum_{r=0}^{\ell_q - 1} \frac{\binom{\ell_q}{r} \bar{q}^{((1-r/\ell_q)n+\nu+\ell')d} \cdot (2\bar{q}^{1-r/\ell_q}/c_1 + 1)^{Ld}}{\bar{q}^{Ld(1-r/\ell_q)}}. \tag{21}$$

Subcase 2 of LHL with Leakage (Lemma 8) For $\boldsymbol{w} \in \Lambda_{\bar{q}}(\boldsymbol{M})$ but not in the column span of $\boldsymbol{E}$ over $\mathbb{Z}$ in subcase where $\boldsymbol{v}_A = 0 \bmod \bar{q}$ and $\boldsymbol{v}_E \neq 0 \bmod \bar{q}$, 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 11 (Minimum Distance of Gadget Matrix $\boldsymbol{G}$). *Let $\boldsymbol{G} = \boldsymbol{g} \otimes I_\nu \in R_{\bar{q}}^{L \times \nu}$ with $\boldsymbol{g} = (1, \beta, \ldots, \beta^{m_q - 1})$, $m_q \geq \lceil \log_\beta(\bar{q}) \rceil$ and $\nu := L/m_q$. Then we get*

$$\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) \geq \bar{q}/\beta^2. \tag{22}$$

Proof. It suffices to prove the claim with $m_q = \lceil \log_\beta(\bar{q}) \rceil$ since the minimum distance of $\Lambda_{\bar{q}}(\boldsymbol{G})$) 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 $\Lambda_{\bar{q}}(\boldsymbol{G})$ will have components like $v \cdot \beta^i \bmod \bar{q}$ for $0 \leq i \leq m_q - 1$. The goal is to show that

$$\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) = \min_{v \neq 0} \max_{0 \leq i \leq m_q - 1} |v \cdot \beta^i \bmod \bar{q}| \geq \bar{q}/\beta^3.$$

If $\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) < \bar{q}/\beta^3$, we show a contradiction. Assume that $\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G}))$ 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 $|v^* \cdot \beta^{i^*+1}| < \bar{q}/\beta^3$ (due to upper bound on λ_1^∞) and hence $|v^* \cdot \beta^{i^*+1}| = |v^* \cdot \beta^{i^*+1} \bmod \bar{q}| > |v^* \cdot \beta^{i^*} \bmod \bar{q}|$, which is a contradiction. This yields $i^* = m_q - 1$ and therefore $\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) = |v^* \cdot \beta^{m_q - 1} \bmod \bar{q}|$. Let us re-write $|v^* \cdot \beta^{m_q - 1} \bmod \bar{q}| = |v_0^* \cdot \beta^{i_0^*} \bmod \bar{q}|$, for some integer v_0^* and $0 < i_0^* \leq m_q - 1$ such that $\gcd(v_0^*, \beta) = 1$. Since $i^* = m_q - 1$, we get that $i_0^* \neq 0$. Let $v_0^* := \beta v_0 + (v_0 \bmod \beta)$, we also see that

$$\begin{aligned}
|v^* \cdot \beta^{m_q - 1} \bmod \bar{q}| &= |v_0^* \cdot \beta^{i_0^*} \bmod \bar{q}| \geq |v_0^* \cdot \beta^{i_0^* - 1} \bmod \bar{q}| \geq \cdots \\
&\geq |v_0^* \cdot \beta^{-1} \bmod \bar{q}| = |v_0^*| \cdot |\beta^{-1} \bmod \bar{q}| \\
&\geq |\beta v_0 + (v_0 \bmod \beta)| \cdot |\bar{q}/\beta \bmod \bar{q}| \qquad (23) \\
&= |\bar{q} \cdot (v_0 \bmod \beta)/\beta \bmod \bar{q}| \geq |\bar{q}/\beta \bmod \bar{q}|, \qquad (24)
\end{aligned}$$

where (23) is induced from the fact that $|\beta \cdot (\beta^{-1} \bmod \bar{q})| \geq \bar{q}$ and (24) is true because $(v_0 \bmod \beta)/\beta \geq 1/\beta$. It is now clear that (24) is in a contradiction with $\lambda_1^\infty(\Lambda_{\bar{q}}(\boldsymbol{G})) < \bar{q}/\beta^3$ and hence $\lambda_1(\Lambda_{\bar{q}}(\boldsymbol{G})) \geq \bar{q}/\beta^2$. □

We now move to state the main result of Subcase 2.

Lemma 12. *Let $L, d, \bar{q}, \ell', \nu \geq 2$ be integers. Let also $c_2 := 2\beta^2$ for an integer $\beta \geq 2$, $c_3 := 4\beta^2 s\sqrt{d\ell'/(2\pi)}$ with $4s^2 d\ell' \geq 2\pi$, $\bar{\boldsymbol{M}} := (\bar{\boldsymbol{E}}, \bar{\boldsymbol{G}}) \in R_{\bar{q}}^{(L+\ell')\times(\ell'+\nu)}$ 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*

$$\Pr_{\boldsymbol{E}\hookleftarrow\mathcal{D}_{R,s}^{L\times\ell'}}[\exists(\boldsymbol{v}_E, \boldsymbol{v}_G), \|\boldsymbol{v}_E\| \leq \bar{q}/c_3 : \boldsymbol{w} \in \Lambda_{\bar{q}}(\bar{\boldsymbol{M}}) \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{\ell' d}, \|\boldsymbol{w}\| < \bar{q}/c_2] \leq Ld2^{-d\ell'}. \tag{25}$$

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 [41] 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 1 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:

$$\Pr\left[\exists \boldsymbol{v}_E \in \mathbb{Z}^{d\ell'}, \|\boldsymbol{v}_E\| \leq \frac{\bar{q}}{c_3} : \bar{\boldsymbol{E}}\boldsymbol{v}_E \bmod \bar{q} \in \Lambda_{\bar{q}}(\bar{\boldsymbol{E}}) \setminus \bar{\boldsymbol{E}}\mathbb{Z}^{d\ell'}\right] \leq Ld2^{-d\ell'}. \tag{26}$$

We now proceed to the second case ($\boldsymbol{v}_G \neq \boldsymbol{0}$). In this case, 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,

$$\begin{aligned}
\|\boldsymbol{E}\boldsymbol{v}_E + \boldsymbol{G}\boldsymbol{v}_G\| &\geq \|\boldsymbol{G}\boldsymbol{v}_G\| - \|\boldsymbol{E}\boldsymbol{v}_E\| \geq \bar{q}/\beta^2 - \|\boldsymbol{E}\boldsymbol{v}_E\| && (27)\\
&\geq \bar{q}/\beta^2 - \|\boldsymbol{E}\| \cdot \|\boldsymbol{v}_E\| && (28)\\
&\geq \bar{q}/\beta^2 - 2s\sqrt{d\ell'/(2\pi)} \cdot \frac{\bar{q}}{4\beta^2 s\sqrt{d\ell'/(2\pi)}} = \bar{q}/2\beta^2 && (29)
\end{aligned}$$

where (27) is induced from triangle inequality and Lemma 11, in (28) we used Cauchy-Schwartz inequality, and (29) is true by the assumed bounds on $\|\boldsymbol{E}\|$ and $\|\boldsymbol{v}_E\|$.

□

4.2 Private Rerandomization of MLWE - Part 2: Gaussian LHL

We now adapt the Gaussian Leftover Hash Lemma from [21] to our module case.

Lemma 13 (Gaussian LHL over modules, adapted from [21]). *Let $\boldsymbol{G} = \boldsymbol{g}\otimes I_\nu \in R_q^{L\times\nu}$ with $\boldsymbol{g} = (1, \beta, \ldots, \beta^{\ell-1})$ and $\nu := L/\ell$. For a fixed $\boldsymbol{E} \in R^{L\times\ell'}$, Let $\bar{\boldsymbol{E}} := \begin{pmatrix} \boldsymbol{E} \\ \boldsymbol{I}_{\ell'} \end{pmatrix} \in R^{(L+\ell')\times\ell'}$, $E_\infty := \|\mathrm{rot}(\boldsymbol{E}^T)\|_\infty$, $\boldsymbol{x} \hookleftarrow \mathcal{D}_{\Lambda_q^\perp(\boldsymbol{G})+\mathbf{c},r} \times D_{R^{\ell'},r}$ and $\epsilon > 0$. Then, if $r \geq \left(\sqrt{\ell}(\beta-1) + \sqrt{\ell' d}((\ell-1)(\beta-1)+1)E_\infty\right)\cdot\sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}}$, we have*

$$\Delta\left(\boldsymbol{x}^T\bar{\boldsymbol{E}}, \mathcal{D}_{\mathbb{Z}^{\ell' d}, r\cdot\mathrm{rot}(\bar{\boldsymbol{E}})}\right) \leq 2\epsilon.$$

It remains to upper bound the smoothing parameter of the lattice Λ in the proof of Lemma 13.

Lemma 14 (Smoothing parameter of orthogonal module lattice, adapted from [21]). *Let $L = \nu m_q$ with m_q and ν be defined as above. For $\boldsymbol{G}$ and $\boldsymbol{E}$, E_∞ and $\epsilon > 0$ as defined in Lemma 13, and the lattice $\Lambda := \{\boldsymbol{v} \in \Lambda_q^\perp(\boldsymbol{G}) \times R^{\ell'} : \boldsymbol{v}^T \bar{\boldsymbol{E}} = \boldsymbol{0}^T\}$, we have $\eta_\epsilon(\Lambda) \leq \left(\sqrt{m_q}(\beta-1) + \sqrt{\ell' d}((m_q-1)(\beta-1)+1)E_\infty\right) \cdot \sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}}$.*

When $\boldsymbol{E} \in R^{L \times \ell'}$ is chosen from a Gaussian distribution $D_{R^{L\times\ell'},s}$, Lemma 1 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 $\bar{\epsilon} \leq 0.001$. This immediately gives the following.

Corollary 1 (Gaussian LHL over modules with Gaussian $\boldsymbol{E}$). *Let $0 \leq \epsilon < 1/2$, $\boldsymbol{G} = \boldsymbol{g} \otimes I_\nu \in R_q^{L\times\nu}$ with $\boldsymbol{g} = (1, \beta, \ldots, \beta^{m_q-1})$ 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*

$$\Delta\left((\boldsymbol{x}^T\bar{\boldsymbol{E}}, \bar{\boldsymbol{E}}), (\mathcal{D}_{\mathbb{Z}^{\ell' d}, r\cdot\mathrm{rot}(\bar{\boldsymbol{E}})}, \bar{\boldsymbol{E}})\right) \leq 3\epsilon.$$

5 New Half-GSW Candidate Linear-Only Encryption

5.1 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-SNARK 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 $2^0\mu, \ldots, 2^{m_q-1}\mu$. For our ZK-SNARK 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. As a result, the set of valid ciphertexts must be sparse. Therefore, as in [39], 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. It consists of three algorithms HGSW.Setup, HGSW.Encrypt,

HGSW.Decrypt. Let $\boldsymbol{g}^T = (1, \beta^1, \ldots, \beta^{m_q-1}) \in R_q^{m_q}$ and $\ell' = \ell + \tau$, where τ is the sparsification parameter. We remark that the scheme is a stateful deterministic encryption scheme, that takes the encrypted message index i (in practice a counter) as input, all the randomness is in the secret key and generated in the Setup algorithm.

- $\mathsf{HGSW.Setup}(1^\lambda, 1^\ell)$: On input a security parameter 1^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 $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{A}, \boldsymbol{E})$.
- $\mathsf{HGSW.Encrypt}(i, \mathsf{sk}, \boldsymbol{\mu})$: Given the message index i, secret key $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{A}, \boldsymbol{E})$ and a vector of messages $\boldsymbol{\mu}_i^T = (\mu_{i,1}, \ldots, \mu_{i,\ell}) \in R_p^\ell$, computes the vector $\bar{\boldsymbol{\mu}}_i^T = [\boldsymbol{\mu}_i^T | (\boldsymbol{T}\boldsymbol{\mu_i})^T] \in R_p^{\ell'}$. Parse $\bar{\boldsymbol{\mu}}_i^T = (\bar{\mu}_{i,1}, \ldots, \bar{\mu}_{i,\ell'})$. The algorithm then computes the ciphertext:

$$\boldsymbol{C}_i = \left[\boldsymbol{A}_i \ \boldsymbol{A}_i\boldsymbol{S} + \boldsymbol{E}_i\right] + \frac{q}{p} \cdot \boldsymbol{H}_i \in R_q^{m_q\times(n+\ell')},$$

  where:

$$\boldsymbol{H}_i := \left[\boldsymbol{0}^{m_q\times n}, \bar{\mu}_{i,1}\boldsymbol{g}, \ldots, \bar{\mu}_{i,\ell'}\boldsymbol{g}\right] \in R_q^{m_q\times(n+\ell')}.$$

- $\mathsf{HGSW.Add}(\{\boldsymbol{C}_i\}_{i\in[m]}, \{a_i\}_{i\in[m]})$: Let L denote the add block size parameter, where $\nu := L/m_q$ is a positive integer (number of ciphertexts per block). For $i \in [m]$, given the scaling factor $a_i \in R$, sample $\boldsymbol{g}_{\mathsf{rand}}^{-1}(a_i)$ (see Def. 4) and for $j = 0, \ldots, m/\nu - 1$, sample $\boldsymbol{y}_j$ from discrete Gaussian $\mathcal{D}_r^{\ell'}$,

$$\boldsymbol{g}_{\mathsf{rand}}^{-1}(a_i) \cdot \boldsymbol{C}_i = \left[\boldsymbol{b}_i^T \ \boldsymbol{b}_i^T\boldsymbol{S} + \boldsymbol{e}_i^T\right] + \frac{q}{p} \cdot \left[\boldsymbol{0}^n, \bar{\mu}_{i,1}a_i, \ldots, \bar{\mu}_{i,\ell'}a_i\right] \in R_q^{m_q\times(n+\ell')},$$

  where $\boldsymbol{b}_i^T = \boldsymbol{g}_{\mathsf{rand}}^{-1}(a_i) \cdot \boldsymbol{A}$, and $\boldsymbol{e}_i^T = \boldsymbol{g}_{\mathsf{rand}}^{-1}(a_i) \cdot \boldsymbol{E}_i$. Finally one computes

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

  Note that 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.
- $\mathsf{HGSW.Decrypt}(\boldsymbol{S}, \boldsymbol{c}^*)$: given a ciphertext $\boldsymbol{c}^*$ and the secret key $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{A}, \boldsymbol{E})$, 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 $\lceil (p/q) \cdot \bar{\boldsymbol{H}} \rfloor$. Set $\bar{\boldsymbol{\mu}} = [\bar{\boldsymbol{\mu}}_1, \bar{\boldsymbol{\mu}}_2]$, where $\bar{\boldsymbol{\mu}}_1 = \boldsymbol{\mu} \in R_q^\ell$ and $\bar{\boldsymbol{\mu}}_2 \in R_q^\tau$ and return $\bar{\boldsymbol{\mu}}_1$ if $\bar{\boldsymbol{\mu}}_2 = \bar{\boldsymbol{\mu}}_1$ and $\perp$ otherwise. Note that 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 construction of Module HGSW is that it keeps the property of homomorphic scaling and drops the multiplicative homomorphism property. Because of this fact, this scheme can be used in our ZK-SNARK from LPCP construction.

Conjecture 1 (HGSW Linear Only). 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 the construction of HGSW is strictly linear-only.

Remark 2. For a ZK-SNARK from linear-only FHE construction, we only require the properties of homomorphic scaling and homomorphic addition, as in the inner-product homomorphism in ZK-SNARK [39]. Therefore it is sufficient if we compute only one row (one Regev ciphertext) of the m_q rows of $\boldsymbol{g}^{-1}_{\mathsf{rand}}(a\boldsymbol{g})$, and apply the homomorphic additions on that one. It means that the ZK-SNARK 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-SNARK 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 $\boldsymbol{C}_3 = \boldsymbol{g}^{-1}_{\mathsf{rand}}(\boldsymbol{C}_{2,2}) \cdot \boldsymbol{C}_1 = [\boldsymbol{C}_{3,1}, \boldsymbol{C}_{3,2}] = [\boldsymbol{g}^{-1}_{\mathsf{rand}}(\boldsymbol{C}_{2,2})\boldsymbol{A}_1, \boldsymbol{g}^{-1}_{\mathsf{rand}}(\boldsymbol{C}_{2,2})\boldsymbol{A}_1\boldsymbol{S} + \boldsymbol{g}^{-1}_{\mathsf{rand}}(\boldsymbol{C}_{2,2})\boldsymbol{E}_1 + \mu_1\boldsymbol{C}_{2,2}] = [\boldsymbol{C}_{3,1}, \boldsymbol{C}_{3,1}\boldsymbol{S} + \boldsymbol{g}^{-1}_{\mathsf{rand}}(\boldsymbol{C}_{2,2})\boldsymbol{E}_1 + \mu_1\boldsymbol{E}_2 + \mu_1\mu_2\boldsymbol{g} + \mu_1\boldsymbol{A}_2\boldsymbol{S}]$. 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-only 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 [22]. 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 [22] to our Regev ciphertext computed in $\mathsf{HGSW.Add}$.

5.2 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*

$$q > 2ps\sqrt{(r^2(mm_q + m/\nu)d + 1)\ln(2((mm_q + m/\nu)d + 1)/\epsilon)/\pi} \tag{30}$$

then (32) *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. However, Theorem 1 assumes in some places that q is prime (e.g. subcase 1 Lemma).

Theorem 3 (Computationally Circuit Privacy). *Let $\epsilon > 0$ and $p, q, n, m_q, \beta, \ell', \nu, L, s, r$ 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*

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

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.*

The IND-CPA security of our Module HGSW follows directly from [35] and Module-LWE assumption.

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. 4, if choose some $d = \widetilde{\theta}(\lambda)$, $n, \ell', p, s, \nu, \tau = O(1)$, then it is sufficient to set some $\beta = \theta(1)$, $\ell_q, \ell_p = O(1)$, $\alpha = \theta(\log(\lambda m)) = \widetilde{O}(1)$, $L = \alpha \cdot (n + \ell' + \nu) = \widetilde{O}(1)$, $r = \widetilde{O}(\lambda^2)$, 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 $d(n + \ell') \log q = O(\log m) \cdot \widetilde{O}(\lambda)$.

6 Application of HGSW to ZK-SNARK

This section is dedicated to an application of our HGSW scheme to ZK-SNARK. The construction of ZK-SNARK follows directly by applying the cryptographic compiler from [17]. Similar to the ZK-SNARK construction in [39], we provide a construction based on linear PCPs for R1CS systems. The main difference to [39] is the underlying linear-only encryption scheme. In our case, we use HGSW defined in Section 5.1.

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 $\prod_{i=[\ell_p]} \mathbb{Z}_p[x]/(f_i(x))$ for irreducible polynomials $f_i(x)$ which are factors of $x^d+1 \mod p$. As shown in [23],

there is an isomorphism $R_p \cong \prod_{i=[\ell_p]} \mathbb{Z}_p[x]/(f_i(x))$ 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-SNARK, 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 $\mathbb{Z}_p[x]/(f_i(x))$.

6.1 Our ZK-SNARK 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-SNARK 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 $\mathsf{HGSW} = (\mathsf{HGSW.Setup}, \mathsf{HGSW.Encrypt}, \mathsf{HGSW.Decrypt})$ be our additively-homomorphic half-GSW symmetric encryption over $\mathbb{F}^k$.

The designated-verifier ZK-SNARK $\Pi_{\mathsf{SNARK}} = (\Pi_{\mathsf{SNARK}}.\mathsf{Setup}, \Pi_{\mathsf{SNARK}}.\mathsf{Prove}, \Pi_{\mathsf{SNARK}}.\mathsf{Verify})$ is defined as follows:

- $\Pi_{\mathsf{SNARK}}.\mathsf{Setup}(1^\lambda, 1^N)$: 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 $\boldsymbol{C}_i = \mathsf{HGSW.Encrypt}(i, \mathsf{sk}, \boldsymbol{q})$ for each $i \in [m]$. Output $\mathsf{crs} = (N, \{\boldsymbol{C}_i\}_{i \in [m]})$ and the verification key $\mathsf{st} = (\mathsf{st}_{\mathsf{LPCP}}, \boldsymbol{S})$.
- $\Pi_{\mathsf{SNARK}}.\mathsf{Prove}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$: 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 $\boldsymbol{\pi}^* = \boldsymbol{c}^*$.
- $\Pi_{\mathsf{SNARK}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{\pi}^*, \boldsymbol{x})$: 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, i.e. $\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}_{\mathsf{LPCP}}, \boldsymbol{x}, \boldsymbol{a})$ and outputs the corresponding output.

Security Discussion. Completeness of our ZK-SNARK follows from the correctness of the underlying HGSW and from the completeness property of the underlying LPCP protocol. Soundness of ZK-SNARK follows from the linear-only property of our HGSW and the soundness property of the LPCP protocol.

Theorem 5 (ZK-SNARK Security). *If Π_{LPCP} is honest-verifier zero-knowledge and the underlying encryption scheme HGSW is IND-CPA secure and computationally circuit private, then Π_{SNARK} is computationally zero-knowledge.*

6.2 Parameter Setting for Our ZK-SNARK

In our work, we adopt a set of notations similar to those in [39] to make it easy to connect the two works together. The notations are summarized in Table 3 in the appendices. For the parameters and requirements common in both [39] and our work, we employ a similar strategy to choose such parameters. For example, as in [39], 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 $N_g = 2^{20}$, which is used as a common example setting in prior works, including [39]. 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 as $\ell = \lceil 4\rho/\ell_p \rceil$. As in [39], 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 [39] is a field and, therefore, does not split (i.e., $\ell_p = 1$ in [39]), 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 [39] with $\ell_p = 1$.

Ciphertext sparsification. For all parameter settings, the sparsification parameter is set as $\tau = \lceil 128/(d \log p) \rceil$. The reason behind this choice is based on the linear-only encryption conjecture (similar to [39]) adapted to our base encryption scheme. Observe that for $\tau = \lceil 128/(d \log p) \rceil$, we have $p^{-\tau d} \leq 2^{-128}$ as in [39]. The difference in our case is that we work over a *ring* R_q instead of a field. However, the rationale described in [39] 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 [39] 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 [39, Theorem 3.19]. Particularly, since we have the same notations for p, n, d, s, q' as in [39], we can simplify the modulus switching requirement to

$$q' > 12pnds, \tag{31}$$

where we use the same constant $C = 6$ as in [39] for Gaussian "tail-cut" bound.

Observation 1 *Note that from the MLWE security perspective the product nd in* (31) *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 error of the PCP is tightly related to the size of the finite field $\mathbb{F}$ over which the PCP is instantiated. Particularly, as in [39], the knowledge error ε is at most $\frac{2N_g}{|\mathbb{F}|-N_g}$, 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 $\varepsilon \leq \frac{2N_g}{p^f - N_g}$ 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-SNARK, Theorem 2, we choose q large enough to ensure that (30) is satisfied. Here, we set $\epsilon = 2^{-128}$ and also have $m \approx 2N_g$ as $\Pi_{\mathsf{SNARK}}.\mathsf{Prove}$ 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 [39], we calculated the "root Hermite factor" δ_{LWE} of the parameter settings in [39] 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 [5] 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 [3] and the number of MLWE samples revealed to the adversary, which can in fact be quite large for the PCP-based SNARK 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 zero-knowledge property of our ZK-SNARK relies on our new private re-randomization results, particularly Theorem 1. These new results of our work (that only impose a $\mathsf{poly}(\kappa)$ condition on the system modulus q for κ-bit zero-knowledge security) are the main reason for the improvements in the SNARK 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 $\kappa = 128$.

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 $m_q(n +$

p	n	d	ℓ_p	f	$\log q$	$\log q'$	ℓ	τ	ρ	β	L	$\log r$	c	ν	proof size (KB)	crs size (GB)	crs size full (GB)
5	31	64	2	32	48.00	17.86	6	1	3	2	288	24.78	271	6	5.30	252.00	1368.00
5	32	64	2	32	49.66	17.91	6	1	3	4	325	26.94	1081	13	5.46	135.79	756.54
5	33	64	2	32	51.55	17.95	6	1	3	9	289	29.12	5472	17	5.61	95.85	547.72
29	33	64	2	32	51.54	20.49	2	1	1	4	312	26.26	708	12	5.76	62.81	753.77
5	34	64	2	32	53.12	17.99	6	1	3	17	252	30.83	20480	18	5.76	75.53	442.39
29	34	64	2	32	53.06	20.53	2	1	1	8	270	28.05	2830	15	5.93	44.77	552.16
5	35	64	2	32	54.65	18.04	6	1	3	32	242	32.53	69173	22	5.92	65.75	394.50
29	35	64	2	32	54.56	20.57	2	1	1	15	238	29.72	10240	17	6.11	35.80	453.53
29	36	64	2	32	56.04	20.61	2	1	1	27	228	31.32	32239	19	6.28	31.52	409.79
53	73	32	2	16	56.12	21.50	4	1	2	26	444	31.04	27290	37	6.55	26.31	410.38

Table 2. Example parameter setting of our ZK-SNARK proposal. We have $\kappa = 128$, $\ell_q = 2$, $s = 2$ and $N_g = 2^{20}$ always.

$\ell')d \log q \cdot 2N_g$ bits. We note that by choosing $s = 1$, the proof sizes in Table 2 can be reduced by 6-8%.

As our main motivation in this work is reducing the proof size, we build on the "shorter proof" parameters of [39] 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 [39], 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 [39], the modulus q' after mod switching also doubles and so $\log q' = 36$. The other parameters are kept the same as in [39], i.e., $p = 2^{13} - 1$, $\rho = 26$, $\ell = 104$ and $\tau = 5$. This produces a proof of 33 KB, a compressed CRS of 10 GB, and a full CRS of 337 GB as given in Table 1.

Compared to [39], our proposal can reduce the proof size up to $6\times$ (with much larger CRS). Alternatively, along with about $5\times$ proof reduction, our full CRS size becomes close to that of [39] while having about $3\times$ larger compressed CRS. The reason behind 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 scheme.

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.

Acknowledgements. This research 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. 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.
4. 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. *IACR Cryptol. ePrint Arch.*, page 941, 2022.
5. M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Math. Cryptol.*, 9(3):169–203, 2015.
6. 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.
7. W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. In *Mathematische Annalen*, volume 296(4), pages 625—-635, 1993.
8. 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.
9. 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.
10. E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza. Snarks for C: verifying program executions succinctly and in zero knowledge. In *CRYPTO 2013*, volume 8043, pages 90–108, 2013.
11. 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.
12. E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Scalable zero knowledge via cycles of elliptic curves. In *CRYPTO 2014*, volume 8617, pages 276–294, 2014.
13. 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.
14. W. Beullens and G. Seiler. Labrador: Compact proofs for R1CS from module-sis. *IACR Cryptol. ePrint Arch.*, page 1341, 2022.
15. N. Bitansky, R. Canetti, A. Chiesa, S. Goldwasser, H. Lin, A. Rubinstein, and E. Tromer. The hunting of the SNARK. *J. Cryptol.*, 30(4):989–1066, 2017.
16. N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *ITCS 2012*, pages 326–349. ACM, 2012.
17. 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.
18. 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.

19. 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.
20. 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.
21. F. Bourse, R. D. Pino, M. Minelli, and H. Wee. FHE circuit privacy almost for free. In *CRYPTO*, volume 9815, pages 62–89, 2016.
22. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3):13:1–13:36, 2014.
23. 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.
24. B. Bünz, B. Fisch, and A. Szepieniec. Transparent snarks from DARK compilers. In *EUROCRYPT 2020*, volume 12105, pages 677–706, 2020.
25. A. Chiesa and E. Yogev. Subquadratic snargs in the random oracle model. In *CRYPTO 2021*, volume 12825, pages 711–741, 2021.
26. 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.
27. 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.
28. L. Ducas and D. Stehlé. Sanitization of FHE ciphertexts. In *EUROCRYPT 2016*, volume 9665, pages 294–310, 2016.
29. M. F. Esgin, R. Steinfeld, and R. K. Zhao. Efficient verifiable partially-decryptable commitments from lattices and applications. In *PKC (1)*, volume 13177, pages 317–348, 2022.
30. 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.
31. 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.
32. C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, USA, 2009.
33. C. Gentry, S. Halevi, and V. Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In *EUROCRYPT (1)*, volume 13275 of *Lecture Notes in Computer Science*, pages 458–487. Springer, 2022.
34. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *ACM STOC, 2008*, pages 197–206. ACM, 2008.
35. 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.
36. 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.
37. J. Groth. Short pairing-based non-interactive zero-knowledge arguments. In *ASIACRYPT 2010*, volume 6477, pages 321–340, 2010.

38. J. Groth. On the size of pairing-based non-interactive arguments. In *EUROCRYPT 2016*, volume 9666, pages 305–326, 2016.
39. 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.
40. J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *ACM STOC 1992*, pages 723–732. ACM, 1992.
41. 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.
42. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015.
43. 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 of *Lecture Notes in Computer Science*, pages 332–364. Springer, 2017.
44. V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT 2012*, volume 7237, pages 738–755, 2012.
45. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-lwe cryptography. In *EUROCRYPT*, volume 7881, pages 35–54, 2013.
46. V. Lyubashevsky and G. Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In *EUROCRYPT 2018*, volume 10820, pages 204–224, 2018.
47. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT 2012*, volume 7237, pages 700–718, 2012.
48. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.
49. N. K. Nguyen. On the non-existence of short vectors in random module lattices. In *ASIACRYPT 2019*, volume 11922, pages 121–150, 2019.
50. A. Nitulescu. Lattice-based zero-knowledge snargs for arithmetic circuits. In *LATINCRYPT 2019*, volume 11774, pages 217–236, 2019.
51. B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *IEEE S&P*, pages 238–252, 2013.
52. C. Peikert. Limits on the hardness of lattice problems in ℓ_p norms. In *IEEE CCC 2007)*, pages 333–346. IEEE Computer Society, 2007.
53. C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO 2008*, volume 5157, pages 554–571, 2008.
54. M. Rosca, D. Stehlé, and A. Wallet. On the ring-lwe and polynomial-lwe problems. In *EUROCRYPT (1)*, volume 10820, pages 146–173, 2018.
55. 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.
56. 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.
57. T. Xie, J. Zhang, Y. Zhang, C. Papamanthou, and D. Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In *CRYPTO 2019*, volume 11694, pages 733–764, 2019.

A Additional Preliminaries and Definitions

A.1 Table of Notations.

In Table 3, we have summarized all important notations used in this paper.

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	
$\boldsymbol{\pi} : \mathbb{F}^m \to \mathbb{F}$	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	$m_q = \lceil \log_\beta q \rceil$
ν	num. of ctxts in re-randomization privacy analysis	
L	num. of rows of the uniform re-randomization matrix	$L = \nu \cdot m_q$

Table 3. List of common parameters and their definitions.

A.2 Field Extensions.

A degree-d field-extension $\mathbb{F}_{p^d}$ of $\mathbb{F}_p$ is a d-dimensional vector space over $\mathbb{F}_p$. For an element $s \in \mathbb{F}_{p^d}$, let $\boldsymbol{v}_s \in \mathbb{F}_p^d$ denote its representation in $\mathbb{F}_p^d$. There is an efficient isomorphism $\phi : s \mapsto \boldsymbol{v}_s$, i.e. for all $s, t \in \mathbb{F}_p^d$, $\boldsymbol{v}_s + \boldsymbol{v}_t = \boldsymbol{v}_{s+t} \in \mathbb{F}_p^d$. Let $\boldsymbol{M}_s \in \mathbb{F}_p^{d \times d}$ denote the linear transformation over $\mathbb{F}_p^d$ corresponding to scalar multiplication by s over $\mathbb{F}_{p^d}$.

A.3 Linear PCP Preliminaries

Definition 6. *(Linear PCP [39]) 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 following properties:

- ***Completeness:*** *For all $N \in \mathbb{N}, \boldsymbol{x} \in \mathbb{F}^n$, $\boldsymbol{w} \in \mathbb{F}^{N_w}$ where $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$,*

$$\Pr[\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}^T\boldsymbol{Q}) = 1 | (\mathsf{st}, \boldsymbol{Q}) \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N), \boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})] = 1.$$

- ***Knowledge:*** *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.
- ***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 $\tilde{\boldsymbol{a}} \leftarrow \mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, \boldsymbol{x})$.

Definition 7. *(Succinct Non-Interactive Argument) 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 $\Pi_{\mathsf{SNARK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ 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 Π_{SNARK} *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 $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N), \boldsymbol{\pi} \leftarrow \mathsf{Prove}(\mathsf{crs}, \boldsymbol{x}, \boldsymbol{w})$*.*
- ***Knowledge:*** *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* $N \in \mathbb{N}$*, and auxiliary inputs* $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$*,*

$$\Pr[\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{\pi}) = 1 \wedge \mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) \neq 1] = \mathsf{negl}(\lambda),$$

  where $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$*,* $(\boldsymbol{x}, \boldsymbol{\pi}) \leftarrow \mathcal{P}^*(1^*, 1^N, \mathsf{crs}; \boldsymbol{z})$*, and* $\boldsymbol{w} \leftarrow \mathcal{E}(1^\lambda, 1^N, \mathsf{crs}, \mathsf{st}, \boldsymbol{x}; \boldsymbol{z})$*.*
- ***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 8. *(Zero-Knowledge) A* SNARK $\Pi_{\mathsf{SNARK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *for an* $R1CS$ *system* $\mathcal{CS} = \{\mathcal{CS}_N\}_{N \in \mathbb{N}}$ *is computational zero knowledge (i.e., a ZK-SNARK) if there exists an efficient simulator* $\mathcal{S}_{\mathsf{SNARK}} = (\mathcal{S}_1, \mathcal{S}_2)$ *such that for all* $N \in \mathbb{N}$ *and all efficient and stateful adversaries* $\mathcal{A}$*, we have that*

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

where the experiment $\mathsf{ExptZK}_{\Pi_{\mathsf{SNARK}}, \mathcal{A}, \mathcal{S}_{\mathsf{SNARK}}}(1^\lambda, 1^N)$ *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 SNARK is a designated verifier if st cannot be efficiently computed from the crs.

Definition 9 (LPCP for R1CS construction [39]). *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 $\tau \leftarrow\!\!\$\ \mathcal{U}(\mathbb{F}\backslash S)$. 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:*

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

$\Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^{N}, \boldsymbol{x}, \boldsymbol{w})$: *On input $N \in \mathbb{N}$ and an instance $(\boldsymbol{x}, \boldsymbol{w})$ where $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$, sample $\delta_1, \delta_2, \delta_3 \leftarrow\!\!\$\ \mathcal{U}(\mathbb{F})$. 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 $a'_1 a'_2 - a'_3 - a_4 z = 0$*

Definition 10 ($\mathbb{F}_{p^d}$ LPCP to $\mathbb{F}_p$ LPCP [39]). *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 a family of R1CS systems $\mathcal{CS} = \{\mathcal{CS}_N\}_{N\in\mathbb{N}}$ over an extension field $\mathbb{F}_{p^d}$ with query length m. A (dk)-query linear PCP $\Pi_{\mathsf{LPCP}} = (\Pi_{\mathsf{LPCP}}.\mathsf{Query}, \Pi_{\mathsf{LPCP}}.\mathsf{Prove}, \Pi_{\mathsf{LPCP}}.\mathsf{Verify})$ for $\mathcal{CS}$ with query length $d\ell$ over the base over the base field $\mathbb{F}_p$:*

- *$\Pi_{\mathsf{LPCP}}.\mathsf{Query}(1^N)$: Run $(\mathsf{st}, \boldsymbol{Q}') \leftarrow \Pi'_{\mathsf{LPCP}}.\mathsf{Query}(1^\lambda)$ where $\boldsymbol{Q}' \in \mathbb{F}_{p^d}^{m\times k}$. Let $\boldsymbol{Q} \in \mathbb{F}_p^{d\ell\times dk}$ be the matrix formed by taking each component $q'_{i,j} \in \mathbb{F}_{p^d}$ in $\boldsymbol{Q}'$ and replacing it with the transpose of the "multiplication-by-$q'_{i,j}$" matrix $\boldsymbol{M}^T_{q'_{i,j}} \in \mathbb{F}_p^{d\times d}$. It holds:*

$$\boldsymbol{Q}' = \begin{bmatrix} q'_{1,1} & \cdots & q'_{1,k} \\ \vdots & \ddots & \vdots \\ q'_{m,1} & \cdots & q'_{m,k} \end{bmatrix}, \quad \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{M}^T_{q'_1,1} & \cdots & \boldsymbol{M}^T_{q'_1,k} \\ \vdots & \ddots & \vdots \\ \boldsymbol{M}^T_{q'_m,1} & \cdots & \boldsymbol{M}^T_{q'_m,k} \end{bmatrix},$$

- *$\Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$: Compute $\boldsymbol{\pi}' \leftarrow \Pi'_{\mathsf{LPCP}}.(1^N, \boldsymbol{x}, \boldsymbol{w}) \in \mathbb{F}_{p^d}^m$. Let $\boldsymbol{\pi} \in \mathbb{F}_p^{dm}$ be the vector formed by taking each component $\pi'_i \in \mathbb{F}_{p^d}$ in $\boldsymbol{\pi}'$ and replacing it with the vector $\boldsymbol{v}_{\pi'_i} \in \mathbb{F}_p^d$ representing π_i. Output the proof vector $\boldsymbol{\pi}$.*
- *$\Pi_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{a})$: Parse $\boldsymbol{a} \in \mathbb{F}_p^{dk}$ as $[\boldsymbol{v}_{a'_1}, \ldots, \boldsymbol{v}_{a'_k}]$ for some $\boldsymbol{a}' = (a'_1, \ldots, a'_k) \in \mathbb{F}_{p^d}^k$. Output $\Pi'_{\mathsf{LPCP}}.\mathsf{Verify}(\mathsf{st}, \boldsymbol{x}, \boldsymbol{a}')$.*

Π_{LPCP} is complete, perfect HVZK and has knowledge error ϵ if the underlying Π'_{LPCP} has the same properties. (see proof of Theorem 3.2 in [39]).

A.4 Vector Encryption Preliminaries

Definition 11 (Linear-Only Vector Encryption (adapted from [18]). *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 $\boldsymbol{C}$.*
- *$\boldsymbol{v}/\perp \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C})$: 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 $\perp$.*
- *$\boldsymbol{C}^* \leftarrow \mathsf{Add}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}, \{y_i\}_{y\in[m]})$: 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 property of CPA security stated above, and the properties of additive homomorphism and circuit privacy are defined as follows:

IND-CPA Security. For all security parameters $\lambda \in \mathbb{N}$ and all efficient adversaries $\mathcal{A}$ their advantage of winning the security game $\mathsf{Game}^{\mathsf{ind-cpa}}_{\mathcal{A},\Pi_{\mathsf{Encrypt}}}$ is given by the following probability:

$$\Pr[\mathsf{Game}^{\mathsf{ind-cpa}}_{\mathcal{A},\Pi_{\mathsf{Encrypt}}}(1^\lambda) = 1] = 1/2 + \mathsf{negl}(\lambda),$$

where $\mathsf{Game}^{\mathsf{ind-cpa}}_{\mathcal{A},\Pi_{\mathsf{Encrypt}}}$ is defined as follows:

1. $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda)$
2. $\mu_0, \mu_1 \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Encrypt}(\mathsf{sk},\cdot)}(1^\lambda)$
3. Select $b \hookleftarrow \mathcal{U}(\{0,1\})$, and compute $c = \mathsf{Encrypt}(sk, \mu_b)$
4. $b' \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Encrypt}(\mathsf{sk},\cdot)}(c)$

If $b' = b$ and c has not been queried to the encryption oracle, output 1, otherwise output 0.

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 $m = m(\lambda)$,

$$\Pr[\sum_{i\in[m]} y_i \boldsymbol{v}_i \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C}^*)] = 1 - \mathsf{negl}(\lambda), \tag{32}$$

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 (32) holds for all $(y_1, \ldots, y_m) \in S$.
Note: The additive homomorphism implies the correctness of the decryption.

Circuit Privacy [32] 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 efficient and 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.

Definition 12 (Linear-only property (adapted from [17]). *A vector encryption scheme* $\Pi_{\mathsf{Encrypt}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add})$ *over* $\mathbb{F}^\ell$ *is strictly linear-only if for all polynomial-size adversaries* $\mathcal{A}$*, there is a polynomial-size extractor* $\mathcal{E}$ *such that for all security parameters* $\lambda \in \mathbb{N}$*, auxiliary inputs* $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$*, and any efficient plaintext generator* $\mathcal{M}$*,*

$$\Pr[\mathsf{ExptLinearExt}_{\Pi_{\mathsf{Encrypt}},\mathcal{A},\mathcal{M},\mathcal{E},z}(1^\lambda) = 1] = \mathsf{negl}(\lambda),$$

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

- *The challenger samples* $(\mathsf{pp}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ *and* $\{\boldsymbol{v}_i\}_{i\in[m]} \leftarrow \mathcal{M}(1^\lambda, \mathsf{pp})$*. It computes* $\boldsymbol{C}_i \leftarrow \mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v}_i)$ *for each* $i \in [m]$ *and runs* $\mathcal{A}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}; z)$ *to obtain a tuple* $(\boldsymbol{C}'_1, \ldots, \boldsymbol{C}'_k)$
- *The challenger computes* $\Pi \leftarrow \mathcal{E}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}; z)$ *and* $\boldsymbol{V}' \leftarrow \Pi \cdot [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m]^T$*, where* $\Pi \in \mathbb{F}^{k\times m}$ *and* $\boldsymbol{V}' \in \mathbb{F}^{k\times\ell}$*. The experiment outputs* 1 *if there exists an index* $i \in [k]$ *such that* $\mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C}'_i) \neq \perp$ *and* $\mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C}'_i) \neq \boldsymbol{v}'_i$*, where* $\boldsymbol{v}'_i \in \mathbb{F}^\ell$ *is the* i*-th row of* $\boldsymbol{V}'$*. Otherwise, the experiment outputs* 0*.*

A.5 ISW Definitions and Constructions

Definition 13 (Vector Encryption [39]). *Let* $d = d(\lambda)$ *be a power of two and let* $R = \mathbb{Z}[x]/(x^d + 1)$*. Fix lattice parameters* $p = p(\lambda), q = q(\lambda), n = n(\lambda)$ *and an error distribution* $\chi = \chi(\lambda)$ *over* R_q*. Let* ℓ *denote the plaintext parameter,* τ *denote the sparsification parameter and* B *denote the noise smudging bound. Set* $\ell' = \ell + \tau$*. A secret-key vector encryption scheme is defined by the following four algorithms* $\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Add}$*:*

- $\mathsf{Setup}(1^\lambda, 1^\ell)$*: Sample matrices* $\boldsymbol{A} \hookleftarrow \mathcal{U}(R_q^{n\times n}), \boldsymbol{S} \hookleftarrow \chi^{n\times\ell'}, \boldsymbol{T} \hookleftarrow R_q^{r\times\ell}$ *and* $\boldsymbol{E} \hookleftarrow \chi^{n\times\ell'}$*. Compute* $\boldsymbol{D} = \boldsymbol{S}^T\boldsymbol{A} + p\boldsymbol{E} \in R_q^{\ell'\times n}$*. Output the secret key* $\boldsymbol{sk} = (\boldsymbol{S}, \boldsymbol{T})$ *and the public parameters* $\mathsf{pp} = (\boldsymbol{A}, \boldsymbol{D})$*.*
- $\mathsf{Encrypt}(\mathsf{sk}, \boldsymbol{v})$*: On input the secret key* $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T})$ *and a vector* $\boldsymbol{v} \in R_q^\ell$ *construct the concatenated vector* $\boldsymbol{u}^T = [\boldsymbol{v}^T, (\boldsymbol{T}\boldsymbol{v})^T] \in R_p^{\ell'}$*. Sample* $\boldsymbol{a} \hookleftarrow \mathcal{U}(R_q^n), \boldsymbol{e} \hookleftarrow \chi^{\ell'}$ *and compute* $\boldsymbol{c} = \boldsymbol{S}^T\boldsymbol{a} + p\boldsymbol{e} + \boldsymbol{u} \in R_q^{\ell'}$*. Output the ciphertext* $\boldsymbol{C} = (\boldsymbol{a}, \boldsymbol{c})$*.*
- $\mathsf{Add}(\mathsf{pp}, \{\boldsymbol{C}_i\}_{i\in[m]}, \{y_i\}_{i\in[m]})$*: On input the public parameters* $\mathsf{pp} = (\boldsymbol{A}, \boldsymbol{D})$*, ciphertexts* $\boldsymbol{C}_i = (\boldsymbol{a}_i, \boldsymbol{c}_i)$ *for* $i \in [m]$*, and scalars* $y_i \in R_p$*, sample* $\boldsymbol{r} \hookleftarrow \chi^n$*,* $\boldsymbol{e}_a \hookleftarrow \chi^n, \boldsymbol{e}_c \hookleftarrow \mathcal{U}([-B, B]^{d\ell'})$ *and output the ciphertext*

$$\boldsymbol{C}^* = \left(\sum_{i\in[m]} y_i\boldsymbol{a}_i + \boldsymbol{A}\boldsymbol{r} + p\boldsymbol{e}_a, \sum_{i\in[m]} y_i\boldsymbol{c}_i + \boldsymbol{D}\boldsymbol{r} + p\boldsymbol{e}_c \right). \tag{33}$$

- $\mathsf{Decrypt}(\mathsf{sk}, \boldsymbol{C})$*: On input the secret key* $\mathsf{sk} = (\boldsymbol{S}, \boldsymbol{T})$ *and a ciphertext* $\boldsymbol{C} = (\boldsymbol{a}, \boldsymbol{c})$*, compute* $\boldsymbol{z} = \boldsymbol{c} - \boldsymbol{S}^T\boldsymbol{a} \in R_q^{\ell'}$*. Compute* $\boldsymbol{u} = \boldsymbol{z} \mod p$ *and parse* $\boldsymbol{u}^T = [\boldsymbol{v}_1^T, \boldsymbol{v}_2^T]$ *where* $\boldsymbol{v}_1 \in R_p^\ell$ *and* $\boldsymbol{v}_2 \in R_p^r$*. Output* $\boldsymbol{v}_1$ *if* $\boldsymbol{v}_2 = \boldsymbol{T}\boldsymbol{v}_1 \in R_p^r$ *and* $\perp$ *otherwise.*

Definition 14 (Honest-Verifier Zero-Knowledge with Leakage [39]). *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 $\mathcal{CS}_\kappa(\boldsymbol{x}, \boldsymbol{w}) = 1$,*

$$\{(\mathsf{st}, \boldsymbol{Q}, [\boldsymbol{Q}\boldsymbol{\pi}]_q, \boldsymbol{Z}, [\boldsymbol{Z}\boldsymbol{\pi}]_q)\} \overset{s}{\approx} \{(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{Z}}, \tilde{\boldsymbol{b}})\}, \tag{34}$$

where $(\mathsf{st}, \boldsymbol{Q}) \leftarrow \mathsf{Query}_{\mathsf{LPCP}}(1^\kappa), \boldsymbol{Z} \leftarrow \mathcal{D}, \boldsymbol{\pi} \leftarrow \mathsf{Prove}_{\mathsf{LPCP}}(1^{\kappa,\boldsymbol{x},\boldsymbol{w}})$, $(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{Z}}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^\kappa)$ 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 $[\boldsymbol{Z}\boldsymbol{\pi}]_q$ to denote computations over the ring R_q (i.e. teh elements of R_q are first lifted to R and teh value of the matrix-vector product is then reduced modulo q). When the statistical distance between the two distributions in (34) is δ, we say that Π_{LPCP} is δ-HVZK with $(\mathcal{D}, q)$-leakage.

B Proofs of Subsection 4.1

B.1 Proof of Lemma 10

Proof. We use a union bound argument similar to that used in Lemma 3.2 of [56] (see also [49]) 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:

$$p_1 \leq \sum_{\substack{(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) \\ \in R_{\bar{q}}^n \setminus \boldsymbol{0} \times R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^{\nu} \times S_{2,\beta}^L}} \Pr_{\boldsymbol{A}^T \hookleftarrow \mathcal{U}(R_{\bar{q}}^{L\times n})} [\boldsymbol{A}\boldsymbol{v}_A = \boldsymbol{t} - \boldsymbol{E}\boldsymbol{v}_E - \boldsymbol{G}\boldsymbol{v}_G]. \tag{35}$$

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

$$p(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) = \prod_{i\in[L]} \frac{|A_i(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})|}{\bar{q}^{dn}}, \tag{36}$$

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

$$p(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) = \prod_{i\in[L]} \frac{\prod_{u\in[\ell_q]} |A_i^{(u)}(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t})|}{\bar{q}^{dn}}, \tag{37}$$

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:

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

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 $\{a_{i,\ell''}^{(u)}\}_{\ell'' \neq \ell'} \in \left(R_{\bar{q}}^{(u)}\right)^{n-1}$, there exists a unique value for $a_{i,\ell'}^{(u)} \in R_{\bar{q}}^{(u)}$ satisfying $\left(\boldsymbol{a}_i^{(u)}\right)^T \boldsymbol{v}_A^{(u)} = t_i^{(u)} - \left(\boldsymbol{e}_i^{(u)}\right)^T \boldsymbol{v}_E^{(u)} - \left(\boldsymbol{g}_i^{(u)}\right)^T \boldsymbol{v}_G^{(u)}$. 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 $\left(\boldsymbol{a}_i^{(u)}\right)^T \boldsymbol{v}_A^{(u)} = 0$ regardless of the choice of $\left(\boldsymbol{a}_i^{(u)}\right)^T$, 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 $t_i^{(u)} - \left(\boldsymbol{e}_i^{(u)}\right)^T \boldsymbol{v}_E^{(u)} - \left(\boldsymbol{g}_i^{(u)}\right)^T \boldsymbol{v}_G^{(u)} = 0$ 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 $\boldsymbol{v}^{(u)} = \boldsymbol{0} \in \left(R_{\bar{q}}^{(u)}\right)^m$ (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 $(\boldsymbol{v}_A, \boldsymbol{v}_E, \boldsymbol{v}_G, \boldsymbol{t}) \in R_{\bar{q}}^n \backslash \boldsymbol{0} \times R_{\bar{q}}^{\ell'} \times R_{\bar{q}}^{\nu} \times S_{2,\beta}^L$ 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

$$p_1 \leq \sum_{r \in [\ell_q - 1]} \frac{|V_r|}{\bar{q}^{Ld(1-r/\ell_q)}} \tag{38}$$

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 $N_i(c_i) \leq \frac{\mathsf{vol}(S_{2,\beta+\lambda/2})}{\mathsf{vol}(\lambda/2)} = (2\beta/\lambda + 1)^d$ for all $i \in [L]$. By Lemma 9, we have $\lambda \geq \bar{q}^{r/\ell_q}$, and we conclude that, setting $\beta := \bar{q}/c$, we have

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

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

$$|V_r| \leq N_Z N_A N_E N_G \prod_{i \in L} N_i(c_i) \leq \bar{q}^{(1-r/\ell_q)n+\ell'+\nu)d} \cdot (2\bar{q}^{1-r/\ell_q}/c + 1)^{Ld}, \quad (39)$$

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 (39) into (38) give (21) and completes the proof. □

C Proofs of Subsection 4.2

Proof of Lemma 13

Proof. The proof proceeds similarly to [21]. The support of the distribution D of $\boldsymbol{x}^T \bar{\boldsymbol{E}}$ is $R^{\ell'} = \mathbb{Z}^{\ell' d}$ due to the submatrix $\boldsymbol{I}_{\ell'}$ in $\bar{\boldsymbol{E}}$ and the fact that the last ℓ' coordinates of $\boldsymbol{x}^T$ over R are independently sampled over support $R^{\ell'}$. Now let $\boldsymbol{z} \in R^{\ell'}$. We have

$$D(\boldsymbol{z}) = D_{\Lambda_q^\perp(\boldsymbol{G}) \times R^{\ell'} + (\boldsymbol{c},0),r}(\Lambda_{\boldsymbol{z}}) = \frac{\rho_r(\Lambda_{\boldsymbol{z}})}{\rho_r(\Lambda_q^\perp(\boldsymbol{G}) \times R^{\ell'} + (\boldsymbol{c},0))},$$

where $\Lambda_{\boldsymbol{z}} := \{\boldsymbol{v} \in \Lambda_q^\perp(\boldsymbol{G}) \times R^{\ell'} : \boldsymbol{v}^T \bar{\boldsymbol{E}} = \boldsymbol{z}^T\}$. Note that $\Lambda_{\boldsymbol{z}}$ is a coset of a $\mathbb{Z}$-rank $L \cdot d$ lattice $\Lambda := \{\boldsymbol{v} \in \Lambda_q^\perp(\boldsymbol{G}) \times R^{\ell'} : \boldsymbol{v}^T \cdot \bar{\boldsymbol{E}} = \boldsymbol{0}^T\}$, so we can write $\Lambda_{\boldsymbol{z}} = \Lambda + \boldsymbol{w}_{\boldsymbol{z}}$, where $\boldsymbol{w}_{\boldsymbol{z}} \in R^{L+\ell'}$ is any solution to $\boldsymbol{w}_{\boldsymbol{z}}^T \cdot \bar{\boldsymbol{E}} = \boldsymbol{z}^T$. Now let $\boldsymbol{u}_{\boldsymbol{z}}$ (resp. $\boldsymbol{u}_{\boldsymbol{z}}^\perp$) denote the projection of $\boldsymbol{w}_{\boldsymbol{z}}$ along (resp. orthogonally to) the rank $\ell' \cdot d$ subspace $V_{\bar{\boldsymbol{E}}}$ of $\mathbb{R}^{(L+\ell')d}$ spanned by the rows of $\bar{\boldsymbol{E}}$. Then, writing $\boldsymbol{w}_{\boldsymbol{z}} = \boldsymbol{u}_{\boldsymbol{z}}^\perp + \boldsymbol{u}_{\boldsymbol{z}}$, we have

$$\rho_r(\Lambda_{\boldsymbol{z}}) = \rho_r(\Lambda + \boldsymbol{u}_{\boldsymbol{z}}^\perp + \boldsymbol{u}_{\boldsymbol{z}}) = \rho_r(\Lambda + \boldsymbol{u}_{\boldsymbol{z}}^\perp) \cdot \rho_r(\boldsymbol{u}_{\boldsymbol{z}}),$$

where in the last equality we used the orthogonality of $\boldsymbol{u}_{\boldsymbol{z}}$ to $\Lambda + \boldsymbol{u}_{\boldsymbol{z}}^\perp$. We show in Lemma 14 below that $r \geq \eta_\epsilon(\Lambda)$. Using this and the fact that $\boldsymbol{u}_{\boldsymbol{z}}^\perp$ is in the span of Λ, Lemma 4 implies that $\rho_r(\Lambda + \boldsymbol{u}_{\boldsymbol{z}}^\perp) \in [\frac{1-\epsilon}{1+\epsilon}, 1] \cdot \rho_r(\Lambda)$. Plugging in the expression for $\rho_r(\Lambda_{\boldsymbol{z}})$ and then into the expression for $D(\boldsymbol{z})$

shows that $D(\boldsymbol{z})$ is within statistical distance $\leq 2\epsilon$ of $D_{R^{\ell'},r}(\boldsymbol{u_z})$. We claim that $D_{R^{\ell'},r}(\boldsymbol{u_z}) = D_{\mathbb{Z}^{d\ell'},r\cdot S}(\boldsymbol{z})$, where $S := \mathrm{rot}(\bar{\boldsymbol{E}})$. Indeed, note that the projection $\boldsymbol{u_z}$ of $\boldsymbol{w_z}$ on the row span of $\mathrm{rot}(\bar{\boldsymbol{E}})$ (over $\mathbb{R}$) can be written as $\boldsymbol{u}_{\boldsymbol{z}}^T = \boldsymbol{w}_{\boldsymbol{z}}^T \cdot (\mathrm{rot}(\bar{\boldsymbol{E}})^*)^T \cdot \mathrm{rot}(\bar{\boldsymbol{E}})^*$, where $\mathrm{rot}(\bar{\boldsymbol{E}})^* = L_E^{-T} \cdot \mathrm{rot}(\bar{\boldsymbol{E}})$ has along its rows the normalized Gram-Schmidt Orthognalization (GSO) of the rows of $\mathrm{rot}(\bar{\boldsymbol{E}})$, and L_E denotes the upper diagonal GSO coefficient matrix. Therefore, $\boldsymbol{u}_{\boldsymbol{z}}^T = \boldsymbol{w}_{\boldsymbol{z}}^T \cdot \mathrm{rot}(\bar{\boldsymbol{E}})^T L_E^{-1} \mathrm{rot}(\bar{\boldsymbol{E}})^* = \boldsymbol{z}^T \cdot L_E^{-1} \mathrm{rot}(\bar{\boldsymbol{E}})^*$, using the fact that $\boldsymbol{w}_{\boldsymbol{z}}^T \cdot \mathrm{rot}(\bar{\boldsymbol{E}})^T = \boldsymbol{z}^T$. It follows that $\boldsymbol{u}_{\boldsymbol{z}}^T \boldsymbol{u_z} = \boldsymbol{z}^T \cdot L_E^{-1} \mathrm{rot}(\bar{\boldsymbol{E}})^* (\mathrm{rot}(\bar{\boldsymbol{E}})^*)^T L_E^{-T} \boldsymbol{z} = \boldsymbol{z}^T \cdot (L_E^T L_E)^{-1} \boldsymbol{z} = \boldsymbol{z}^T \cdot (S^T S)^{-1} \boldsymbol{z}$ with $S := \mathrm{rot}(\bar{\boldsymbol{E}})$, as required. □

C.1 Proof of Lemma 14

Proof. Recall that that Λ has a $\mathbb{Z}$-rank $L \cdot d$. By Lemma 7, 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 $\|\boldsymbol{u}_i\| \leq \|\boldsymbol{b}_i\| + \sqrt{\ell' d} \cdot ((m_q - 1)(\beta - 1) + 1) \cdot E_\infty \leq \sqrt{m_q}(\beta - 1) + \sqrt{\ell' d}((m_q - 1)(\beta - 1) + 1) E_\infty$. □

D Proofs of Section 5

D.1 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

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

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 $\bar{\boldsymbol{S}}^T = \left[-\boldsymbol{S}^T \ \boldsymbol{I}_{\ell'} \right]$. Replacing $\boldsymbol{g}_{\mathrm{rand}}^{-T}(a_{j\nu+i}) = \boldsymbol{x}_{j,i}^T$ 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 $\bar{\boldsymbol{X}}^T := \left[\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_m^T, \boldsymbol{y}_0^T, \ldots, \boldsymbol{y}_{m/\nu-1}^T \right]$ and $\bar{\boldsymbol{E}} := \left[\boldsymbol{E}_1, \ldots, \boldsymbol{E}_m, \boldsymbol{I}_{\ell'}, \ldots, \boldsymbol{I}_{\ell'} \right]$, we now observe that thanks to (30), 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 2, 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

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

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 6 and Lemma 11, and that $c \geq c_2 \geq \beta^2$. Therefore, by Lemma 3, 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 $i \in [m]$.

D.2 Proof of Theorem 4

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 $\boldsymbol{e}_j^T \leftarrow \mathcal{D}_{\mathbb{Z}^{\ell' d}, r \cdot \mathrm{rot}(\bar{\boldsymbol{E}}_j)}$, 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

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

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

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 $\boldsymbol{x}_{j\nu+i}^T := \boldsymbol{g}_{\mathsf{rand}}^{-1}(a_{j\nu+i})$, 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 $D_j := \mathcal{U}(R_q^n) \times \mathcal{D}_{\mathbb{Z}^{\ell' d}, r \cdot \mathrm{rot}(\bar{\boldsymbol{E}}_j)}$ 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 $\boldsymbol{c}_0^*$. □

E Proof of Theorem 5

Proof. Bitansky et al. [17] showed that the zero-knowledge property of a ZK-SNARK construction from linear-only encryption and LPCP follows from the re-randomization (circuit privacy) property of the underlying linear-only encryption and from the honest-verifier zero-knowledge property of the underlying LPCP protocol. In Section 4 we proved the rerandomization property of our module full GSW and half GSW schemes.

Let $\mathcal{S}_{\mathsf{LPCP}} = (\mathcal{S}_{\mathsf{LPCP},1}, \mathcal{S}_{\mathsf{LPCP},2})$ be a simulator of linear PCP $\mathit{\Pi}_{\mathsf{LPCP}}$ and $\mathcal{S}_{\mathsf{HGSW}}$ be the simulator for circuit privacy of the underlying HGSW scheme. We build a simulator $\mathcal{S}_{\mathsf{SNARK}} = (\mathcal{S}_{\mathsf{SNARK},1}, \mathcal{S}_{\mathsf{SNARK},2})$ of our ZK-SNARK construction from the simulators of the underlying building blocks, LPCP and HGSW:

- $\mathcal{S}_{\mathsf{SNARK},1}(1^\lambda, 1^N)$: Take as input a security parameter λ and the system parameter N of $\{\mathcal{CS}_N\}$ and run the LPCP simulator $\mathcal{S}_{\mathsf{LPCP}}$, i.e. $\left(\tilde{\mathsf{st}}, \tilde{\boldsymbol{Q}}, \mathsf{st}_{\mathcal{S}}\right) \leftarrow \mathcal{S}_{\mathsf{LPCP},1}$, for a $\tilde{\boldsymbol{Q}} \in \mathbb{F}^{m \times k}$ is a query matrix. Let $\tilde{S} \leftarrow \mathsf{HGSW.Setup}(1^\lambda)$ and compute $\boldsymbol{C}_i = \mathsf{HGSW.Encrypt}(\boldsymbol{S}, \boldsymbol{q}_i^T)$ for each $i \in [m]$. Output $\tilde{\mathsf{crs}} = (N, \tilde{\boldsymbol{C}}_1, \ldots, \tilde{\boldsymbol{C}}_m)$ and the verification key $\tilde{\mathsf{st}} = (\tilde{\mathsf{st}}_{\mathsf{LPCP}}, \tilde{\boldsymbol{S}})$ and the simulation state $\mathsf{st}_{\mathcal{S}} = (\mathsf{st}_{\mathsf{LPCP}}, \tilde{\boldsymbol{S}})$.
- $\mathcal{S}_{\mathsf{SNARK},2}(\mathsf{st}_{\mathcal{S}}, \boldsymbol{x})$: Take as input $\mathsf{st}_{\mathcal{S}} = (\mathsf{st}_{\mathsf{LPCP}}, \tilde{\boldsymbol{S}})$ and the statement $\boldsymbol{x}$, compute $\tilde{\boldsymbol{a}} \leftarrow \mathcal{S}_{\mathsf{LPCP},2}(\mathsf{st}_{f,\boldsymbol{x}})$.
  Output the simulated proof $\tilde{\pi} = \mathcal{S}_{\mathsf{HGSW.Encrypt}}(1^\lambda, \tilde{\boldsymbol{S}}, \tilde{\boldsymbol{a}})$

The proof is done via a hybrid argument (similar to [39])

- Hybrid_0: This hybrid is equivalent to the original game of zero-knowledge of Π_{SNARK}:
  The challenger samples $b \leftarrow_r \{0,1\}$. 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 $\mathcal{A}$. The adversary outputs a statement $\boldsymbol{x}$ and a witness w. The challengers checks the circuit satisfiability, i.e. $\mathcal{CS}_N(\boldsymbol{x}, \boldsymbol{w}) = 1$. If the check succeeds, the challenger computes the following:
  If $b = 0$, the challenger replies with $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)$. If $b = 1$, the challenger replies with $\tilde{\pi} \leftarrow \mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, x)$.
  At the end of the experiment, $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. The final output of the experiment is $b' \in \{0,1\}$ indicating a win of $\mathcal{A}$ if $b' = b$.
- Hybrid_1: This hybrid is the same as Hybrid_0 except the proof $\boldsymbol{\pi}$ is constructed using $\mathcal{S}_{\mathsf{HGSW.Encrypt}}$. Let $\boldsymbol{Q} \in \mathbb{F}^{m \times k}$ be the query matrix sampled by the challenger to construct $\mathsf{crs} = (N, \boldsymbol{C}_1, \ldots, \boldsymbol{C}_m)$ and let $\mathsf{st} = (\mathsf{st}_{\mathsf{LPCP}}, \boldsymbol{S})$ be the corresponding verification state. To compute π the challenger runs $\boldsymbol{\pi} \leftarrow \Pi_{\mathsf{LPCP}}.\mathsf{Prove}(1^N, \boldsymbol{x}, \boldsymbol{w})$ and computes $\pi \leftarrow \mathcal{S}_{\mathsf{HGSW.Encrypt}}(1^\lambda, \boldsymbol{S}^T, \boldsymbol{a})$, where $\boldsymbol{a} \leftarrow \sum_{i=1}^{m} \pi_i \boldsymbol{q}_i = \boldsymbol{Q}\boldsymbol{\pi}$ where $\boldsymbol{q}_i$ is the i-th row of $\boldsymbol{Q}$.
- Hybrid_2: This hybrid is the same as Hybrid_1 except the challenger runs $\mathcal{S}_{\mathsf{LPCP}}$ to simulate the proof π and crs. The challenger samples $(\mathsf{st}, \boldsymbol{Q}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_{\mathsf{LPCP},1}(1^N)$. Then the challenger computes $\boldsymbol{a} \leftarrow \mathcal{S}_{\mathsf{LPCP},2}(\mathsf{st}_{\mathcal{S}}\boldsymbol{x})$ which replaces $\boldsymbol{a} = \boldsymbol{Q}\boldsymbol{\pi}$ in Hybrid_1.

We denote by $\mathsf{Hyb}_i(\mathcal{A})$ the output of Hybrid_i. The difference between Hybrid_0 and Hybrid_1 is the computation of π using $\mathsf{S}_{\mathsf{HGSW.Encrypt}}$. Since our HGSW is ϵ-circuit private, then

$$|\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1]| =\leq 2\epsilon.$$

Now, the difference between Hybrid_1 and Hybrid_2 is the computation of $\boldsymbol{S}, \boldsymbol{a}$ using $\mathcal{S}_{\mathsf{LPCP}}$. Therefore if Π_{LPCP} is honest-verifier zero-knowledge then

$$\Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1] = \Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1]$$

Since the challenger behaves independently of the bit b it follows $\Pr[\mathsf{Hyb}_2 = 1] = 1/2$. □