# LUNA+: More Succinct Post-Quantum ZK-SNARKs from Computational Privacy

Yuki Kume
Monash University
Melbourne, Australia
yuki0003@student.monash.edu

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

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

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

## Abstract

We present LUNA+, a refinement of the LUNA designated-verifier lattice-based ZK-SNARG that achieves significantly improved concrete succinctness. While the original LUNA scheme achieves quasi-optimal asymptotic proof length ($O(\lambda)$), its practical parameters are constrained by its statistical privacy analysis. This analysis, founded on a Leftover Hash Lemma with Leakage (LHLL), necessitates the use of polynomially large, but still significant "smudging" noise to guarantee statistical uniformity. This noise inflation directly propagates to larger lattice dimension and modulus parameters and, consequently, larger proof and CRS sizes.

Our core contribution is a new privacy analysis that replaces this statistical foundation with a computational one. We demonstrate that the circuit privacy of LUNA's re-randomization procedure can be securely based on the computational hardness of the Matrix Hint-Module Learning With Errors (MH-MLWE) problem. This computational approach avoids the need for large statistical noise and enables a key optimization: we decouple the secret re-randomization noise from the fresh masking noise. We then formalize and solve an optimization problem to find the minimal noise parameters that satisfy both correctness and the MH-MLWE security reduction. In the process, we also introduce a new problem called Coset Error Knapsack MH-MLWE in which the MLWE error is sampled from a coset of a lattice, which we show is as hard as the standard MH-MLWE problem, and may be of independent interest.

This new analysis results in substantial concrete efficiency gains. For a 128-bit security level and an R1CS instance of size $2^{16}$, LUNA+ reduces the proof size by $\approx$ 25% (from 5.60 KB to 4.22 KB) and the compressed CRS size by $\approx$ 73% (from 2.06 GB to 0.54 GB) compared to the original LUNA. These succinctness improvements are also accompanied by performance gains, including up to a $\approx$ 1.73× speedup in setup, a $\approx$ 1.53× speedup in addition and a 1.44× speedup in decryption for the implementation parameters.

## 1 Introduction

Zero-Knowledge Succinct Non-interactive Arguments (ZK-SNARGs) [17, 20] are a fundamental cryptographic primitive enabling a prover to convince a verifier of the truth of an NP statement without revealing the underlying witness, using a proof substantially shorter than the witness itself. ZK-SNARGs have found numerous applications, including verifiable computation [30] and privacy-preserving cryptocurrencies [2]. The emergence of quantum computers necessitates the migration to post-quantum cryptographic schemes. While lattice-based cryptography currently offers the most promising path towards quantum-resistant ZK-SNARGs, achieving proof succinctness comparable to pre-quantum, pairing-based constructions (e.g., Groth16 [18] with sub-kilobyte proofs) remains a significant challenge [19, 29].

While recent progress in (plain) lattice-based ZK-SNARGs reduced the proof length (e.g. [4, 6, 13]), the shortest concrete proof lengths to date are achieved in the *designated-verifier* (DV) preprocessing model [5, 7]. In this paradigm, a Linear Probabilistically Checkable Proof (LPCP) [14] is compiled with a linear-only additively homomorphic vector encryption scheme. The prover homomorphically computes an encryption of the LPCP answers using a proof vector derived from the witness, allowing the verifier to decrypt and check the LPCP verification predicate [15, 19].

A critical bottleneck in these constructions is achieving succinct zero-knowledge. Early efficient constructions, such as Ishai et al. (ISW) [19], relied on "exponential smudging" [16], adding large noise to mask the internal randomness leaked during homomorphic evaluation. While ensuring statistical privacy, this ciphertext re-randomization technique causes proof lengths to scale sub-optimally as $\Omega(\lambda^2)$ with the security parameter $\lambda$. Steinfeld et al. [29] subsequently introduced LUNA, the first lattice-based DV ZK-SNARG candidate achieving *asymptotically* quasi-optimal proof length $\tilde{O}(\lambda)$. LUNA replaced simple smudging with a refined re-randomization technique based on a lattice Leftover Hash Lemma with Leakage (LHLL), enabling statistical circuit privacy with noise parameters polynomial in the privacy parameter.

Despite LUNA's asymptotic optimality, its reliance on a *statistical* privacy argument via the LHLL introduced significant concrete overhead. Masking the witness-dependent coset structure requires a Gaussian noise parameter large enough to statistically "smooth" the distribution [29], inflating the ciphertext modulus $q$ and lattice dimension $n$, resulting in bloated proof and CRS sizes.

*Our Contribution.* In this work, we present LUNA+, a refinement of the LUNA ZK-SNARG that achieves improved concrete succinctness by shifting the foundation of its circuit privacy from a statistical argument to a *computational* one. Our core idea is to replace the privacy analysis based on the LHLL with one based on a

variant of the computational hardness of the **Matrix Hint-Module Learning With Errors (MH-MLWE)** problem [12, 32].

The MH-MLWE problem posits that a standard MLWE sample $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$ remains computationally indistinguishable from uniform even if the adversary is given an additional "hint" $\mathbf{z} = \mathbf{Ms} + \mathbf{y}$ about the MLWE secret $\mathbf{s}$ (or a hint about the MLWE error $\mathbf{e}$, in the knapsack variant of the problem, where $\mathbf{b} = \mathbf{Ae}$), where $\mathbf{M}$ is a public hint matrix and $\mathbf{y}$ is fresh 'hint mask' noise. Crucially, reductions from standard MLWE [12, 32] show that this computational hardness holds under specific conditions relating the variances of the error $\mathbf{s}$ (parameter $\sigma_s$) and the hint noise $\mathbf{y}$ (parameter $\sigma_y$).

We demonstrate that the privacy of the re-randomization process in LUNA's `HGSW.Add` procedure can be established based on the hardness of a variant of the MH-MLWE problem, which we call Coset Error Knapsack MH-MLWE (CEK-MH-MLWE). In the CEK-MH-MLWE problem, the MLWE sample, given in knapsack form $(\mathbf{A}, \mathbf{b} = \mathbf{Ae})$, has the MLWE error $\mathbf{e}$ sampled from a discrete Gaussian distribution on a coset $\Lambda + \mathbf{c}$ of a lattice $\Lambda$, while the hint is still given as $\mathbf{z} = \mathbf{Me} + \mathbf{y}$ with a fresh discrete Gaussian mask $\mathbf{y}$ on the integer lattice (not a lattice coset). The hardness of CEK-MH-MLWE ensures that $(\mathbf{A}, \mathbf{b})$ is computationally indistinguishable from uniform and hence hides the coset shift $\mathbf{c}$. In the application of CEK-MH-MLWE to LUNA+'s HGSW privacy analysis, $\mathbf{b}$ corresponds to the re-randomized HGSW ciphertext, while the MLWE error $\mathbf{e}$ in the CEK-MH-MLWE problem corresponds to the HGSW ciphertext re-randomization vector (sampled from a coset of a gadget lattice corresponding to the LUNA+ hidden witness), while the hint $\mathbf{z}$ corresponds to the noise accumulated during re-randomization of the HGSW ciphertext. We note that a variant of the MH-MLWE with a similar name, namely Coset MH-MLWE, was recently introduced in [23], but the latter differs from ours: in [23] the lattice coset appears in the distribution support of the hint mask $\mathbf{y}$, whereas in CEK-MH-MLWE the lattice coset is in the support of the MLWE error vector $\mathbf{e}$.

By relying on the computational hardness of CEK-MH-MLWE, we guarantee computational circuit privacy for the HGSW homomorphic encryption scheme in LUNA without requiring the large noise parameters needed for statistical uniformity under the LHLL. Furthermore, to demonstrate the hardness of CEK-MH-MLWE based on the standard MLWE assumption, we show a probabilistic and polynomial time (PPT) reduction from the standard MH-MLWE to our new CEK-MH-MLWE problem, for appropriate parameter settings.

Our computational framework based on CEK-MH-MLWE also enables a key optimization in parameter settings. The original LUNA used a single noise parameter $r$ for both the secret re-randomization vectors (analogous to $\mathbf{s}$) and the fresh masking noise (analogous to $\mathbf{y}$). We decouple these into distinct Gaussian width parameters $r_x$ (re-randomization) and $r_y$ (masking noise), and solve an optimization problem minimizing total noise variance (which dictates $q$ and proof size) subject to the MH-MLWE security constraint relating $r_x$ and $r_y$.

*Security Assumptions Summary.* Overall, the circuit privacy of HGSW (and LUNA+) relies only on the standard MLWE hardness assumption. Our circuit privacy reduction proceeds through the following chain:

(1) MLWE to Matrix-Hint MLWE (MH-MLWE): known reduction from [12, 32].
(2) MH-MLWE to our CEK-MH-MLWE: reduction in Lemma 4.1.
(3) CEK-MH-MLWE to HGSW Circuit Privacy: reduction in Theorem 3.2.

We introduce the intermediate CEK-MH-MLWE problem for ease of exposition and because its hardness may be of independent interest for future protocol design. The ZK-SNARG soundness of LUNA+ relies on the same 'HGSW Strictly Linear Targeted Malleability' assumption inherited from LUNA [29]. As with LUNA, under slightly stronger but plausible assumptions regarding the linear-only property of HGSW and LPCP knowledge soundness [29], LUNA+ can also be shown to be a ZK-SNARK (argument of knowledge).

*Impact of Quantum LWE Sampling.* Debris-Alazard, Fallahpour, and Stehlé [9] provided a quantum oblivious LWE sampling algorithm that puts into question the knowledge soundness of certain lattice-based designated-verifier zkSNARKs. However, as discussed in the original LUNA paper [29, Appendix A], LUNA's soundness relies on a variant of the linear-only property (strictly linear targeted malleability, or SLTM) that is unaffected by this quantum sampler, since the SLTM property does not require knowledge extraction from LWE samples. This analysis applies equally to LUNA+, as LUNA+ inherits the same soundness argument and SLTM-based linear-only property as LUNA.

*Our Evaluation Results.* LUNA+ significantly improves concrete succinctness over LUNA while maintaining 128-bit security and quasi-optimal asymptotic proof length. By leveraging CEK-MH-MLWE hardness and optimizing decoupled noise parameters $(r_x, r_y)$, we achieve substantially smaller system parameters $(q, n)$. For 128-bit security and R1CS size $2^{16}$, **LUNA** [29] requires $\log q \approx 49.5$, $n = 64$, yielding a proof size of **5.60 KB** and compressed CRS of **2.06 GB**, with $\log r \approx 26.7$. LUNA+ reduces this to $\log q \approx 37.11$, $n = 48$, yielding **4.22 KB** proofs and **0.54 GB** compressed CRS, with $(\log r_x, \log r_y) \approx (14.21, 18.59)$, a **25%** proof size reduction and over **3.8×** CRS reduction. Our implementation also outperforms the LUNA implementation, achieving speedups of up to **1.73×** in setup, **1.53×** in addition, and **1.44×** in decryption. These improvements make LUNA+ more practical for bandwidth and storage-constrained applications.

## 2 Preliminaries

### 2.1 Notation

We denote the security parameter by $\lambda$. A function $f(\lambda)$ is negligible, denoted $\mathrm{negl}(\lambda)$, if for every positive polynomial $p(\cdot)$, there exists an integer $N$ such that for all $\lambda > N$, we have $f(\lambda) < 1/p(\lambda)$. We use PPT to denote probabilistic polynomial-time. For a distribution $\mathcal{D}$, we write $x \leftarrow \mathcal{D}$ to denote sampling. The integer set $\{1, \ldots, n\}$ is denoted by $[n]$.

*Vectors and Matrices.* We denote column vectors by bold lowercase letters (e.g., $\mathbf{v}$) and matrices by bold uppercase letters (e.g., $\mathbf{M}$). The transpose of a vector $\mathbf{x}$ is $\mathbf{x}^T$. The transpose and inverse of a matrix $\mathbf{M}$ are $\mathbf{M}^T$ and $\mathbf{M}^{-1}$, respectively. The zero matrix and identity matrix of dimensions $m \times n$ and $n \times n$ are denoted by $\mathbf{0}_{m \times n}$

and $\mathbf{I}_n$. For a vector $\mathbf{v} \in \mathbb{R}^n$, its Euclidean norm is $\|\mathbf{v}\|$ and its infinity norm is $\|\mathbf{v}\|_\infty$. For a matrix $\mathbf{M}$, we use $\|\mathbf{M}\|$ (resp. $\|\mathbf{M}\|_\infty$ and $\|\mathbf{M}\|_1$) to denote the spectral Euclidean norm (resp. spectral $\infty$ and 1 norm) of $\mathbf{M}$.

*Rings and Modules.* For a positive integer $q$, $\mathbb{Z}_q$ is the ring of integers modulo $q$. Our constructions are built over power-of-two cyclotomic rings. For a power-of-two integer $d$, we define the ring of polynomials $R = \mathbb{Z}[x]/(x^d + 1)$. Its modular counterpart is $R_q = R/qR = \mathbb{Z}_q[x]/(x^d + 1)$. Elements in $R_q$ are polynomials of degree less than $d$ with coefficients in $\mathbb{Z}_q$. Vectors and matrices over $R$ or $R_q$ are modules. For an element $a \in R$, we denote its canonical embedding into $\mathbb{Z}^d$ as a vector of its coefficients.

## 2.2 Lattices and Discrete Gaussians

We first recall some fundamental definitions.

*Definition 2.1 (Lattice).* An $n$-dimensional lattice $\Lambda$ is a discrete additive subgroup of $\mathbb{R}^n$. It can be represented by linearly independent vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ such that $\Lambda = \{\sum_{i=1}^n c_i \mathbf{b}_i \mid c_i \in \mathbb{Z}\}$.

*Definition 2.2 (Discrete Gaussian Distribution).* For a positive real parameter $\sigma > 0$ and a vector $\mathbf{c} \in \mathbb{R}^n$, the Gaussian function on $\mathbb{R}^n$ is defined as $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\|\mathbf{x} - \mathbf{c}\|^2/\sigma^2)$. The discrete Gaussian distribution over a lattice $\Lambda \subset \mathbb{R}^n$ is defined by the probability mass function $\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{x})}{\rho_{\sigma,\mathbf{c}}(\Lambda)}$ for any $\mathbf{x} \in \Lambda$, where $\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{\mathbf{y}\in\Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{y})$. When $\mathbf{c} = \mathbf{0}$, we simply write $\rho_\sigma$ and $\mathcal{D}_{\Lambda,\sigma}$.

Let $\Lambda \subset \mathbb{R}^m$ be a lattice generated by a basis matrix $\mathbf{B} \in \mathbb{Z}^{m\times m}$, such that $\Lambda_\mathbf{B} = \mathbf{B}\mathbb{Z}^m$. Let $\mathbf{c} \in \mathbb{R}^m$ be a shift vector defined by $\mathbf{c} = \mathbf{B}\mathbf{c}_0$ for some $\mathbf{c}_0 \in \mathbb{Z}^m$. We denote the discrete Gaussian distribution over a lattice coset $\Lambda + \mathbf{c}$ with parameter $\sigma$ and center $\mu$ as $\mathcal{D}_{\Lambda+\mathbf{c},\sigma,\mu}$.

Lemma 2.3 (Gaussian Convolution). *Let $\Lambda$ be a lattice. The sum of two independent discrete Gaussian samples approximates a discrete Gaussian with summed covariances: $\mathcal{D}_{\Lambda,\sigma,0} + \mathcal{D}_{\Lambda,\sigma,\mu} \approx \mathcal{D}_{\Lambda,\sqrt{2}\sigma,\mu}$.*

Lemma 2.4 (Linear Transformation). *Let $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\Sigma,0}$. For a full-rank matrix $\mathbf{B}$, the random variable $\mathbf{y} = \mathbf{B}\mathbf{x}$ follows a discrete Gaussian distribution over $\Lambda_\mathbf{B}$ with covariance $\Sigma' = \mathbf{B}\Sigma\mathbf{B}^T$.*

Lemma 2.5 (Gaussian Tail Bound [19, 31]). *Let $\sigma > 0$ be the standard deviation parameter (related to the Gaussian width $s$ by $s = \sigma\sqrt{2\pi}$). For a value $e$ sampled from the discrete Gaussian distribution over $\mathbb{Z}$ with parameter $s$, $\mathcal{D}_{\mathbb{Z},s}$, and for any threshold $t > 0$, the probability that the absolute value of $e$ exceeds $t$ is bounded by:*

$$\Pr[|e| > t] \le 2\exp\left(-\frac{\pi t^2}{s^2}\right).$$

*Definition 2.6 (The $g_{rand}^{-1}$ Algorithm [25, 29]).* Let $\beta, q \in \mathbb{Z}$, with $\beta \ge 2$, and let the gadget vector be $\mathbf{g}^T = (1, \beta, \beta^2, \ldots, \beta^{m_q-1}) \in R^{1\times m_q}$, where $m_q = \lceil \log_\beta q \rceil$. For any $a \in R_q$, there exists an efficient randomized algorithm, denoted $g_{\text{rand}}^{-1}(a)$, that outputs a row vector $\mathbf{x}^T \in R^{1\times m_q}$ satisfying $\mathbf{x}^T\mathbf{g} = a \pmod q$. The output $\mathbf{x}^T$ follows a discrete Gaussian distribution $\mathcal{D}_{\Lambda_q^\perp(\mathbf{g})+\mathbf{c},r}$ for some Gaussian parameter $r$ and any fixed vector $\mathbf{c}$ such that $\mathbf{c}^T\mathbf{g} = a \pmod q$. This procedure extends element-wise to input vectors $\mathbf{a} \in R_q^\rho$. For the block gadget matrix $\mathbf{G} = \mathbf{I}_\rho \otimes \mathbf{g} \in R_q^{L\times\rho}$ (where $L = \rho m_q$), the output $\mathbf{x}^T = g_{\text{rand}}^{-1}(\mathbf{a})$ satisfies $\mathbf{x}^T\mathbf{G} = \mathbf{a} \pmod q$, and $\mathbf{x}$ is sampled from $\mathcal{D}_{\Lambda_q^\perp(\mathbf{G})+\mathbf{c},r}$ for an appropriate $\mathbf{c}$. Here, we recall the notation $\Lambda_q^\perp(\mathbf{H}) := \{\mathbf{v} \in R^l : \mathbf{H}\cdot\mathbf{v} = 0 \bmod q\}$ for a $q$-ary $R$-module (SIS) lattice with respect to a matrix $\mathbf{H} \in R_q^{v\times l}$.

## 2.3 Module Hardness Assumptions

The security of our scheme is based on the hardness of the Module Learning With Errors (MLWE) problem and its variants.

*Definition 2.7 (Module Learning With Errors (MLWE) [22]).* Let $n, l, d$ be positive integers, $q$ be a modulus, and $\chi$ be an error distribution over $R_q$. The $\text{MLWE}_{n,l,d,q,\chi}$ decision problem is to distinguish between samples of the form $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$, where $\mathbf{A} \leftarrow \mathcal{U}(R_q^{l\times n})$, $\mathbf{s} \leftarrow \chi^n$, and $\mathbf{e} \leftarrow \chi^l$; and samples of the form $(\mathbf{A}, \mathbf{u})$, where $\mathbf{u} \leftarrow \mathcal{U}(R_q^l)$. The MLWE assumption states that this distinguishing advantage is negligible for any PPT adversary.

We also utilize the knapsack variant of the MLWE problem.

*Definition 2.8 (Knapsack MLWE (knMLWE) [10]).* Let $n, m, h$ be positive integers such that $n > m$, $q$ be a modulus, and $\chi$ be an error distribution over $R$. The $\text{knMLWE}_{n,m,h,\chi}$ decision problem asks an adversary $\mathcal{A}$ to distinguish between $(\mathbf{A}, \mathbf{SA})$ where $\mathbf{A} \leftarrow \mathcal{U}(R_q^{n\times m})$ and $\mathbf{S} \leftarrow \chi^{h\times n}$; and $(\mathbf{A}, \mathbf{U})$ where $\mathbf{U} \leftarrow \mathcal{U}(R_q^{h\times m})$. The knMLWE assumption states that the distinguishing advantage is negligible for any PPT adversary $\mathcal{A}$.

The hardness of knMLWE is closely related to standard MLWE via duality arguments. Its hardness has been established based on MLWE in prior works [10, 11], and this knapsack formulation proves particularly suitable for analyzing certain protocols, including aspects of our zero-knowledge proof construction later in this paper.

Our computational privacy analysis relies on a more recent variant, Matrix Hint-MLWE, which formalizes the intuition that MLWE remains hard even if some correlated linear information ("hints") about the secret is revealed. Under specific conditions on the noise parameters, the hint does not make the problem significantly easier. The initial concept of Hint-MLWE, focusing on hints related to secrets and errors in proof systems, was introduced by Kim et al. [21]. This was later generalized to the Matrix Hint-MLWE setting, allowing for matrix-based hints, in works such as [12, 32]. Our work utilizes a generalized version similar to that in [32], which considers non-square hint matrices and provides refined reduction conditions suitable for our parameter optimization.

*Definition 2.9 ((Primal) Matrix Hint-MLWE [12, 32]).* Let $n, l, k, d$ be positive integers, $q$ a modulus, $\sigma_e$ denotes the standard deviation parameter for the discrete Gaussian error distribution and $\chi_s, \chi_e, \chi_y, \chi_M$ be distributions over $R^{l-n}, R^l, R^k, R^{k\times l}$ respectively, where $R := \mathbb{Z}[x]/(x^d + 1)$, and $R_q := \mathbb{Z}_q[x]/(x^d + 1)$. The (primal) Matrix Hint-MLWE decision problem, denoted $\text{MH-MLWE}_{n,l,k,d,q,\sigma_e,\chi_s,\chi_y,\chi_M}$, is to distinguish between two distributions:

(1) **Real Distribution** $\text{MH-MLWE}^0(b = 0)$**:** $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}, \mathbf{M}, \mathbf{z} = \mathbf{M}\mathbf{e} + \mathbf{y})$, where $\mathbf{A} \leftarrow \mathcal{U}(R_q^{l\times(l-n)})$, $\mathbf{M} \leftarrow \chi_M$, $\mathbf{s} \leftarrow \chi_s$, $\mathbf{e} \leftarrow \chi_e := \mathcal{D}_{R^l,\sigma_e}$, and $\mathbf{y} \leftarrow \chi_y$.

(2) **Random Distribution** $\text{MH-MLWE}^1(b = 1)$**:** $(\mathbf{A}, \mathbf{b}, \mathbf{M}, \mathbf{z} = \mathbf{M}\mathbf{e} + \mathbf{y})$, where all variables are sampled as above, except $\mathbf{b} := \mathbf{u}$ with $\mathbf{u} \leftarrow \mathcal{U}(R_q^l)$.

The Matrix Hint-MLWE assumption asserts that for any PPT adversary $\mathcal{A}$, the advantage $\mathrm{Adv}^{\text{MH-MLWE}}_{\mathcal{A}}(\lambda)$ in distinguishing these two cases is negligible, where:

$$\mathrm{Adv}^{\text{MH-MLWE}}_{\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}=1|\text{MH-MLWE}^1]- \Pr[\mathcal{A}=1|\text{MH-MLWE}^0]|.$$

*Definition 2.10 (Knapsack MH-MLWE).* Let $n, m, k, d$ be positive integers, $q$ a modulus, and $\chi_S, \chi_Y$ be distributions over $R_q = \mathbb{Z}_q[x]/(x^d+1)$. The K-MH-MLWE$_{n,m,k,d,q,\chi_S,\chi_Y}$ problem is to distinguish between two distributions:

(1) **Real Distribution:** $(\mathbf{A}, \mathbf{b}^T = \mathbf{s}^T\mathbf{A}, \mathbf{M}, \mathbf{z}^T = \mathbf{s}^T\mathbf{M}+\mathbf{y}^T)$, where $\mathbf{A} \leftarrow \mathcal{U}(R_q^{n\times m})$, $\mathbf{M} \leftarrow \mathcal{D}_M$ (from a specified distribution of hint matrices over $R_q^{n\times k}$), $\mathbf{s}^T \leftarrow \chi_S^{1\times n}$, and $\mathbf{y}^T \leftarrow \chi_Y^{1\times k}$.
(2) **Random Distribution:** $(\mathbf{A}, \mathbf{u}^T, \mathbf{M}, \mathbf{z}^T = \mathbf{s}^T\mathbf{M}+\mathbf{y}^T)$, where all variables are sampled as above, but $\mathbf{u}^T \leftarrow \mathcal{U}(R_q^{1\times m})$.

The K-MH-MLWE assumption asserts that for any PPT adversary, the advantage in distinguishing these two cases is negligible.

Remark 1. *The hardness of the Knapsack MLWE problem (Definition 2.8) is closely related to the standard MLWE problem. As discussed in [12] and [11], under standard conditions,* knMLWE$_{n,m,h,d,q,\chi}$ *is computationally as hard as* MLWE$_{n-m,n,d,q,\chi}$*. A similar relationship is expected to hold between the K-MH-MLWE problem (Definition 2.10) and the standard MH-MLWE problem (Definition 2.9), building upon the reductions for MH-MLWE to standard MLWE presented in [12, 32]. However, the analysis in the proof of the MH-MLWE reduction in [12, 32] does not make any assumption on the distribution of the matrix* $\mathbf{A}$*; whether it is of the form* $[\mathbf{I}|\mathbf{A}']$ *with* $\mathbf{A}'$ *random (Primal variant) or of the form* $[\mathbf{A}'_1|\mathbf{A}'_2]$ *with both random (Knapsack), the proof holds unchanged. However, we emphasize that LUNA+ and our CEK-MH-MLWE hardness only require the* primal *variant of MH-MLWE (see Lemma 4.1), which is the variant explicitly shown to be as hard as MLWE in [12, 32]. Therefore, this remark is not required for the security of LUNA+. The Knapsack variants are more convenient for constructing schemes involving row vectors or matrix products where the secret appears on the left, as is the case in our later proofs regarding the HGSW scheme.*

Remark 2 (Row Vector Structure in K-MH-MLWE). *When* $h = 1$ *in Definition 2.10, the secret* $\mathbf{S}$ *becomes a row vector* $\mathbf{s}^T \in R_q^{1\times n}$*, and the problem reduces to distinguishing* $(\mathbf{A}, \mathbf{s}^T\mathbf{A}, \mathbf{M}, \mathbf{s}^T\mathbf{M}+\mathbf{y}^T)$ *from* $(\mathbf{A}, \mathbf{u}^T, \mathbf{M}, \mathbf{s}^T\mathbf{M}+\mathbf{y}^T)$ *where* $\mathbf{A} \in R_q^{n\times m}$*,* $\mathbf{M} \in R_q^{n\times k}$*,* $\mathbf{s}^T \leftarrow \chi_s^{1\times n}$*,* $\mathbf{y}^T \leftarrow \chi_y^{1\times k}$*, and* $\mathbf{u}^T \leftarrow \mathcal{U}(R_q^{1\times m})$*. This formulation aligns naturally with HGSW ciphertexts, and we assume this structure in our proof.*

Lemma 2.11 (Hardness of Knapsack Matrix Hint-MLWE (Variant of [12, 32])). *The hardness* K-MH-MLWE$_{dim_{\mathbf{S}},m,k,d,q,\chi_S,\chi_Y}$*, can be reduced from a standard* MLWE$_{n-m,n,d,q,\chi}$ *problem. Let the distributions* $\chi_S$ *and* $\chi_Y$ *be discrete Gaussians with standard deviations* $\sigma_S$ *and* $\sigma_Y$ *respectively. Let* $B$ *be a high-probability bound on the squared spectral norm of the integer representation of the hint matrix,* $\|\tilde{\mathbf{M}}\|_2^2 \leq B$*. The reduction holds, and K-MH-MLWE is computationally as hard as a standard* MLWE *problem with a secret from a Gaussian distribution of standard deviation* $\sigma$*, provided the following conditions hold for a reduction parameter* $\delta > 0$*:*

(1) ***Sampleability Condition:***

$$\frac{1}{(1+\delta)\sigma^2+\delta_0} \geq \frac{1}{\sigma_S^2} + \frac{B}{\sigma_Y^2},$$

*where* $dim_{\mathbf{S}}$ *is the dimension of the secret* $\mathbf{S}$ *over the ring* $R$*, and* $\delta_0 := \sqrt{\frac{\ln(2(dim_{\mathbf{S}})d)+4}{\pi}}$.
(2) ***Convolution Condition:***

$$\sigma \geq \sqrt{1+1/\delta}\cdot\eta_\epsilon(\mathbb{Z}^{(dim_{\mathbf{S}})d}),$$

*for some* $\epsilon \in (0, 1/2)$*, where* $\eta_\epsilon(\cdot)$ *is the smoothing parameter of a lattice.*

Remark 3. *In our HGSW scheme, the secret vector* $\mathbf{s}$ *corresponds to the aggregated re-randomization vector* $\mathbf{x}_{agg}$*. This vector is constructed by concatenating* $m$ *individual re-randomization vectors* $\mathbf{x}_i$*, each of dimension* $m_q$ *over the ring* $R$*. Therefore, the total dimension of the secret vector is* $dim_{\mathbf{s}} = m \cdot m_q$*. If these conditions are met, there exists an efficient reduction from the standard MLWE problem (with parameter* $\sigma$*) to the K-MH-MLWE problem. The hardness holds due to the underlying reduction for standard MH-MLWE [12, 32] combined with the known duality between standard and Knapsack MLWE [10, 11].*

## 2.4 Key results from LUNA [29]

Our work, like LUNA [29], is based on a variant of the GSW homomorphic encryption scheme [16], which we refer to as Half-GSW (HGSW). It is a linear-only vector encryption scheme whose security relies on MLWE. The formal definition of Linear-Only Vector Encryption and its required properties (additive homomorphism, CPA security, circuit privacy, and the strictly linear targeted malleability property) are given in Appendix A.1. The LUNA ZK-SNARG uses HGSW as a black-box instantiation of this primitive: an LPCP is compiled with HGSW following the framework of Bitansky et al. [5]. Consequently, improving the circuit privacy analysis and parameter selection of HGSW (as we do in this work) directly improves LUNA, without requiring any modification to the outer SNARG construction. We summarize the HGSW scheme algorithms from [29] below. Here, we denote by $\mathbf{g} \in R^{m_q}$ the gadget vector $\mathbf{g}^T = (1, \beta, \ldots, \beta^{m_q-1})$ for a base $\beta$.

- HGSW.Setup$(1^\lambda, m, l')$: Samples a secret key matrix $\mathbf{S} \in R^{n\times l'}$, and for $i \in [m]$, public matrices $\mathbf{A}_i \in R_q^{m_q\times n}$ and error matrices $\mathbf{E}_i \in R^{m_q\times l'}$ with small entries. The secret key is $sk = \mathbf{S}$ and the public parameters are $pp = (\{\mathbf{A}_i, \mathbf{E}_i\}_{i\in[m]})$. In the ZK-SNARG context, these parameters form the CRS.
- HGSW.Encrypt$(i, sk, \boldsymbol{\mu})$: This is a secret-key operation used conceptually in the CRS setup. To encrypt a message vector $\boldsymbol{\mu}_i \in R_p^{l'}$, it outputs a ciphertext matrix $\mathbf{C}_i = [\mathbf{A}_i|\mathbf{A}_i\mathbf{S}+\mathbf{E}_i] + \frac{q}{p}\cdot[\mathbf{0}^{m_q\times n}|\boldsymbol{\mu}_{i,1}\mathbf{g}|\ldots|\boldsymbol{\mu}_{i,l'}\mathbf{g}] \in R_q^{m_q\times(n+l')}$. The set of all $\{\mathbf{C}_i\}_{i\in[m]}$ constitutes the CRS for the SNARG.
- HGSW.Add$(\{\mathbf{C}_i\}, \{a_i\})$: To compute a linear combination of ciphertexts with scalars $\{a_i\}$, it first computes re-randomization vectors $\mathbf{x}_i^T \leftarrow g^{-1}_{\text{rand}}(a_i) \in R^{1\times m_q}$. It then computes the scaled ciphertexts $\tilde{\mathbf{c}}_i^T = \mathbf{x}_i^T\mathbf{C}_i$. Finally, it sums them up with fresh masking noise $\mathbf{y}$: $\mathbf{c}^{*T} = \sum_i \tilde{\mathbf{c}}_i^T + [\mathbf{0}^{1\times n}|\mathbf{y}^T]$. The output $\mathbf{c}^*$ is a single-row Regev-style ciphertext.

- HGSW.Decrypt($sk, \mathbf{c}^*$): The final ciphertext $\mathbf{c}^{*T} = [\mathbf{b}^T|\mathbf{d}^T]$ is decrypted by computing $\mathbf{d}^T - \mathbf{b}^T\mathbf{S}$ and rounding to recover the message vector.

LUNA scheme [29] relied on the following theorems for statistical privacy and correctness, which we later replace with Theorem 3.2 and Theorem 5.1.

Theorem 2.12 (Private Re-randomization of MLWE Samples [29]). *Let $R_q := \mathbb{Z}_q[x]/(x^d+1)$ with $d$ a power of 2, $q = p\bar{q}$ with prime $\bar{q} = 2l_q + 1 \pmod{4l_q}$, where $l_q \geq 2$ is a power of 2 so that $x^d+1$ splits into $l_q$ irreducible factors $f^{(u)}(x) \pmod{\bar{q}}$ for $u \in [l_q]$, where each $f^{(u)}(x)$ has degree $d/l_q$. Let $L, v \in \mathbb{Z}$, $0 < \epsilon \leq 1/2$, $G = I_v \otimes g \in R_q^{L\times v}$ with $g^T = (1, \beta, \ldots, \beta^{m_q-1})$ for an integer $\beta \geq 2$, $m_q := \lceil \log_\beta(q) \rceil$, $v := L/m_q \geq 1$, and $\bar{q} \geq 3\beta^2/(\beta-1)$. For $A \leftarrow \mathcal{U}(R_q^{L\times n})$, $E \leftarrow \mathcal{D}_{R,s}^{L\times l'}$ and $\bar{x} \leftarrow \mathcal{D}_{(\Lambda_q^\perp(G)+c)\times R^{l'},r}$, let $\bar{E} := \begin{pmatrix} E \\ I_{l'} \end{pmatrix} \in R^{(L+l')\times l'}$, $\bar{A} := \begin{pmatrix} A \\ 0_{l'\times n} \end{pmatrix} \in R^{(L+l')\times n}$. Let $E_\infty := s\sqrt{\frac{2\ln(Ll'd/\epsilon)+\ln\ln(Ll'd/\epsilon)}{2\pi}} \geq 1$ with $\epsilon/(Ll'd) \leq 0.001$. If*

$$q > \max(2(L+l')dc_p\sqrt{1+4s^2Ld/(2\pi)}, \quad 2r(s\sqrt{Ld}+1)\sqrt{\ln(2l'd/\epsilon)/\pi}), \tag{1}$$

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

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

$$p_1(c_1) := \sum_{k=0}^{l_q-1} \frac{\binom{l_q}{k}\bar{q}^{((1-k/l_q)n+v+l')d} \cdot (2\sqrt{d}\bar{q}^{1-k/l_q}/c_1+1)^{Ld}}{\bar{q}^{Ld(1-k/l_q)}} \leq \epsilon \tag{3}$$

*$c_2 := 4\beta$, and $Ld2^{-l'd} \leq \epsilon$*

$$c_3 := 8\beta s\sqrt{dl'/(2\pi)} \text{ with } 4s^2dl' \geq 2\pi. \tag{4}$$

$$r_G := (\sqrt{m_q}(\beta-1)+\sqrt{l'd}((m_q-1)(\beta-1)+1)E_\infty)\cdot\sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}}. \tag{5}$$

*then we have*

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

Theorem 2.13 (Correctness (Additive Homomorphism) [29]). *Let $\lambda$ be the security parameter, and let $p, q, n, m_q, l', d, m, v$ be positive integers. Let $s$ be the Gaussian parameter for initial errors $E_i$, and let $r$ be the Gaussian width parameter for re-randomization vectors $x_i$ and masking noise $y_j$. Assume $p, q, n, m_q, l', d, m, v, s, r = poly(\lambda)$ and $n > 8$. If the ciphertext modulus $q$ satisfies*

$$q > 2ps\sqrt{(r^2(m\cdot m_q + m/v)d+1)\ln(2((m\cdot m_q+m/v)d+1)/\epsilon_{corr})/\pi} \tag{6}$$

*then the HGSW scheme satisfies the Additive Homomorphism property for $m$ additions using the HGSW.Add algorithm, with correctness error probability at most $\epsilon_{corr}$.*

Lemma 2.14 (Gaussian LHL over modules [29]). *Let $0 \leq \epsilon < 1/2$, $G = I_v \otimes g \in R_q^{L\times v}$ with $g^T = (1, \beta, \ldots, \beta^{m_q-1})$ and $v := L/m_q$. Let $E \leftarrow \mathcal{D}_{R,s}^{L\times l'}$, and $\bar{E} := \begin{pmatrix} E \\ I_{l'} \end{pmatrix}$. Let $E_\infty := s\sqrt{\frac{2\ln(Ll'd/\epsilon)+\ln\ln(Ll'd/\epsilon)}{2\pi}} \geq 1$ with $\epsilon/(Ll'd) \leq 0.001$. If*

$$r \geq (\sqrt{m_q}(\beta-1)+\sqrt{l'd}((m_q-1)(\beta-1)+1)E_\infty)\cdot\sqrt{\frac{\ln(2Ld(1+\epsilon^{-1}))}{\pi}},$$

*then we have*

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

# 3 Computational Circuit Privacy from Matrix Hint-MLWE

Our primary contribution is a new privacy analysis for HGSW.Add's re-randomization. The original LUNA [29] established statistical circuit privacy via a lattice-based Leftover Hash Lemma with Leakage (LHLL, Theorem 2.12), requiring noise parameter $r$ large enough to statistically overwhelm leakage ("smudging"), leading to conservative parameters and larger proofs.

Our improvement stems from shifting to computational privacy guarantee. We demonstrate that the privacy of the re-randomization procedure can be founded on the computational hardness of the Matrix Hint-MLWE problem [12, 32]. This problem formalizes the scenario where an adversary, given a standard MLWE sample, also receives an additional linear "hint" about the underlying secret. The hardness assumption states that this hint provides no significant advantage in distinguishing the MLWE sample from a uniform one.

By showing that the structure of our re-randomization process can be mapped to a Matrix Hint-MLWE instance, we can rely on its computational hardness, as established in Lemma 2.11. This allows for significantly smaller re-randomization noise parameters, as the noise only needs to be large enough for the computational problem to remain hard, not to achieve statistical uniformity. This results in a tighter overall analysis, smaller system parameters (e.g., modulus q), and ultimately, more succinct proofs.

## 3.1 Mapping HGSW Re-randomization to Matrix Hint-MLWE

To establish the connection, we first analyze the structure of the data an adversary (or the designated verifier) can observe. The output of the `HGSW.Add` algorithm is a Regev-style ciphertext vector $\mathbf{c}^{*T} = [\mathbf{b}^T|\mathbf{d}^T]$. The components are formed as follows:

$$\mathbf{b}^T = \sum_{i=1}^m \mathbf{x}_i^T\mathbf{A}_i$$

$$\mathbf{d}^T = \mathbf{b}^T\mathbf{S} + \underbrace{\left(\sum_{i=1}^m \mathbf{x}_i^T\mathbf{E}_i + \mathbf{y}^T\right)}_{\text{accumulated noise and masking}} + \frac{q}{p}\sum_{i=1}^m a_i\boldsymbol{\mu}_i.$$

Here, the $\{\mathbf{x}_i\}$ are the secret re-randomization vectors derived from the witness scalars $\{a_i\}$, the $\{\mathbf{A}_i\}$ and $\{\mathbf{E}_i\}$ are public matrices from the CRS, $\mathbf{S}$ is the HGSW secret key, and $\mathbf{y}$ is fresh masking noise sampled by the prover.

A designated verifier, who knows the secret key $\mathbf{S}$ and the message sum $\sum a_i\boldsymbol{\mu}_i$, can decrypt the ciphertext and isolate the total noise term. Critically, this process reveals two distinct but related structures to an observer:

(1) A public, MLWE-like sample: The vector $\mathbf{b}^T$ is a linear combination of the public matrices $\{\mathbf{A}_i\}$ where the coefficients are the secret re-randomization vectors $\{\mathbf{x}_i\}$.
(2) A "hint" about the secret: The total accumulated noise, $\mathbf{e}_{\text{total}}^T = \sum_{i=1}^m \mathbf{x}_i^T\mathbf{E}_i + \mathbf{y}^T$, is also a linear combination of public

matrices $\{\mathbf{E}_i\}$ with the same secret coefficients $\{\mathbf{x}_i\}$, masked by fresh noise $\mathbf{y}$.

To formalize this, let the aggregated secret re-randomization vector be $\mathbf{x}_{\text{agg}}^T = [\mathbf{x}_1^T | \dots | \mathbf{x}_m^T]$, and let the aggregated public and error matrices be the concatenated matrices $\mathbf{A}_{\text{agg}}^T := [\mathbf{A}_1^T | \mathbf{A}_2^T | \dots | \mathbf{A}_m^T]$ and $\mathbf{E}_{\text{agg}}^T := [\mathbf{E}_1^T | \mathbf{E}_2^T | \dots | \mathbf{E}_m^T]$, similar to the definition of $\mathbf{x}_{\text{agg}}$. The verifier effectively observes:

- An MLWE-like sample: $\mathbf{b}^T = \mathbf{x}_{\text{agg}}^T \mathbf{A}_{\text{agg}}$
- A hint about the secret $\mathbf{x}_{\text{agg}}$: $\mathbf{e}_{\text{total}}^T = \mathbf{x}_{\text{agg}}^T \mathbf{E}_{\text{agg}} + \mathbf{y}^T$

This structure matches the form of the Matrix Hint-MLWE problem, but with a critical distinction regarding the error distribution. In standard MH-MLWE, the error is typically a centered discrete Gaussian. However, in our HGSW construction, the "secret" $\mathbf{x}_{\text{agg}}$ is sampled via the $g_{\text{rand}}^{-1}$ algorithm, which outputs vectors from a specific coset of the gadget lattice, namely $\Lambda_q^{\perp}(\mathbf{G}_{\text{agg}}) + \mathbf{c}$.

Consequently, the effective MLWE error distribution in the verifier's view follows a discrete Gaussian over a *coset* of a certain $q$-ary lattice, rather than a standard discrete Gaussian over an integer lattice. Therefore, we explicitly reduce the circuit privacy of HGSW from the **Coset Error Knapsack Matrix Hint-MLWE (CEK-MH-MLWE)** problem.

*Definition 3.1 (Coset Error Knapsack Matrix Hint-MLWE (CEK-MH-MLWE)).* Let $n, l, k, d$ be positive integers and $q$ be a modulus. Let $\Lambda \subset \mathbb{R}^l$ be a $q$-ary lattice with determinant $q^{v_\Lambda}$. Let $\chi_e', \chi_y', \chi_M'$ be distributions over $R^l, R^k, R^{k\times l}$ respectively, where $R := \mathbb{Z}[x]/(x^d + 1)$, and $R_q := \mathbb{Z}_q[x]/(x^d + 1)$. The Coset Error Knapsack (Dual) Matrix Hint-MLWE problem, denoted CEK-MH-MLWE$_{n,l,k,d,q,\sigma_e',\chi_y',\chi_M',\Lambda}$, is defined by the following game between a challenger and a two-phase adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$:

(1) **Phase 1:** The challenger samples $\mathbf{A}' \leftarrow \mathcal{U}(R_q^{n\times l})$ and $\mathbf{M}' \leftarrow \chi_{M'}$ (where $\mathbf{M}' \in R_q^{k\times l}$). The adversary $\mathcal{A}_1(\mathbf{A}', \mathbf{M}')$ outputs a coset defining vector $\mathbf{c} \in R_q^l$ (representing a shift in the lattice $\Lambda$) and state information st.

(2) **Phase 2:** The challenger samples a secret error $\mathbf{e}' \leftarrow \mathcal{D}_{\Lambda+\mathbf{c},\sigma_e'}$, a hint noise $\mathbf{y}' \leftarrow \chi_y'$, and a bit $b \leftarrow \{0, 1\}$.
   - If $b = 0$ (Real), generate $\mathbf{b}' = \mathbf{A}'\mathbf{e}'$.
   - If $b = 1$ (Random), generate $\mathbf{b}' \leftarrow \mathcal{U}(R_q^n)$.

   The challenger computes the hint $\mathbf{z}' = \mathbf{M}'\mathbf{e}' + \mathbf{y}'$. The adversary $\mathcal{A}_2(\mathbf{b}', \mathbf{z}', \text{st})$ outputs a guess $b'$.

The advantage $\text{Adv}_{\mathcal{A}}^{\text{CEK-MH-MLWE}}(\lambda)$ is defined as $|\Pr[b' = b] - 1/2|$. The problem is hard if the advantage is negligible for any PPT adversary.

Remark 4. *Lapiha and Prest [23] introduced Coset Hint-MLWE problem, where their hint mask* $\mathbf{y}$ *is sampled from a coset of a lattice Gaussian, while our definition of coset error is for the MLWE error* $\mathbf{e}$*. Our MLWE error is sampled from a coset of a Gaussian over the orthogonal lattice generated by Gadget matrix* $\mathbf{G}$*.*

The precise mapping used in the reduction is detailed in Table 1.

The fundamental security property we need for circuit privacy is that the public part of the ciphertext, $\mathbf{b}^T$, should be computationally indistinguishable from a uniformly random vector, even to an adversary who sees the hint, $\mathbf{e}_{\text{total}}^T$. This is exactly the guarantee provided by the hardness of the CEK-MH-MLWE problem,

**Table 1: Mapping HGSW params to CEK-MH-MLWE instance.**

| HGSW Parameter | CEK-MH-MLWE Param |
|---|---|
| Agg. re-randomization $\mathbf{x}_{\text{agg}}$ | MLWE Secret $\mathbf{s}$ (Coset Sample) |
| Agg. public CRS $\mathbf{A}_{\text{agg}}$ | MLWE Challenge matrix $\mathbf{A}'$ |
| Agg. error matrix $\mathbf{E}_{\text{agg}}$ | MLWE Hint Matrix $\mathbf{M}'$ |
| Fresh ctxt masking noise $\mathbf{y}$ | MLWE Hint mask noise $\mathbf{y}'$ |
| Public ctxt $\mathbf{b}^T$ | MLWE Sample $\mathbf{b}' = \mathbf{s}^T\mathbf{A}'$ |
| Acc. error $\mathbf{e}_{\text{tot}}^T$ | MLWE Hint $\mathbf{z}' = \mathbf{s}^T\mathbf{M}' + \mathbf{y}'$ |

as defined below, and established in Lemma 2.11. In contrast, the original LUNA framework [29] achieved statistical circuit privacy through the LHLL argument (Theorem 2.12. The LHLL required the re-randomization noise (analogous to $\mathbf{x}_{agg}$) to be large enough (parameter $r$ in Theorem 2.12) to statistically "smudge" or overwhelm the information leaked through the hint term (analogous to $\mathbf{e}_{\text{total}}^T$). Relying on the computational hardness of a variant of MH-MLWE allows us to use significantly smaller noise parameters, as computational security holds even when statistical uniformity does not, leading to improved efficiency.

## 3.2 Computational Circuit Privacy of HGSW and the Security Games

We now formalize the reduction outlined in the previous section from CEK-MH-MLWE to computational circuit privacy of the HGSW scheme, using a standard game-hopping proof. The proof demonstrates that the output of the real `HGSW.Add` algorithm is computationally indistinguishable from the output of an ideal simulator that only knows the final message sum.

Theorem 3.2 (Computational Circuit Privacy of HGSW). *Let the HGSW scheme be defined with parameters* $\lambda, m, l', d, q, p, n, v, m_q, \beta, s$*. Let the re-randomization vectors* $\mathbf{x}_k$ *be sampled from a discrete Gaussian with width parameter* $r_x$ *($\sigma_s = r_x/\sqrt{2\pi}$), and the fresh masking noise* $\mathbf{y}_j$ *be sampled from a discrete Gaussian with width parameter* $r_y$ *($\sigma_y = r_y/\sqrt{2\pi}$). Let* $\mathbf{G} = \mathbf{I}_m \otimes \mathbf{g}^T \in R^{m\times mm_q}$ *denote a gadget matrix, where* $\mathbf{g}^T = (1, \beta, \beta^2, \dots, \beta^{m_q-1}) \in R^{1\times m_q}$*.*

*The HGSW scheme with the above parameters is computationally circuit private if the HGSW parameters satisfy the conditions of Lemma 2.14 for some* $\epsilon > 0$ *and* $L := mm_q$*, and the* Coset Error Knapsack Matrix Hint-MLWE *problem* CEK-MH-MLWE$_{n,l,k,d,q,\sigma_e',\chi_y',\chi_M',\Lambda}$ *is hard with the following parameters:*

*(1)* $l = mm_q$ *(Knapsack MLWE error dimension)*
*(2)* $k = l'$ *(Hint dimension)*
*(3)* $\sigma_e' = r_x$ *(Knapsack MLWE error Gaussian parameter)*
*(4)* $\chi_y' = \mathcal{D}_{\mathbb{Z}^k, r_y}$ *(Hint mask dimension)*
*(5)* $\chi_M' = (\mathcal{D}_{\mathbb{Z}^{l'}, s})^{mm_q}$ *(Hint matrix distribution)*
*(6)* $\Lambda = \Lambda_q^{\perp}(\mathbf{G})$ *(lattice for error coset)*
*(7)* $v_\Lambda = m$ *(syndrome dimension of lattice for error coset)*

*Specifically, for any PPT circuit privacy adversary* $\mathcal{A}$ *against HGSW, there exists a PPT adversary* $\mathcal{B}$ *against the* Coset Error Knapsack Matrix Hint-MLWE *problem* CEK-MH-MLWE$_{n,l,k,d,q,\sigma_e',\chi_y',\chi_M',\Lambda}$ *such*

*that:*

$$\mathrm{Adv}^{\text{circ-priv}}_{\mathcal{A}}(\lambda) \le \mathrm{Adv}^{\text{CEK-MH-MLWE}}_{\mathcal{B}}(\lambda) + 5\epsilon.$$

Proof. We proceed with a sequence of hybrid games. For any PPT adversary $\mathcal{A}$, let $\mathrm{Adv}_{\mathcal{A},i}$ be its advantage in distinguishing the challenger's output in Game $i$ from a simulated one. The goal is to show that the advantage in the real world (Game 0) is negligible.

The circuit privacy experiment (Definition A.3) involves a challenger and an adversary $\mathcal{A}$. The challenger runs $(pp, sk) \leftarrow \mathrm{HGSW.Setup}(1^\lambda)$, gives $(pp, sk)$ to $\mathcal{A}$, who then returns a set of messages $\{\boldsymbol{\mu}_i\}_{i\in[m]}$. The challenger computes ciphertexts $\{\mathbf{C}_i\}_{i\in[m]}$ and gives them to $\mathcal{A}$, who responds with scalars $\{a_i\}_{i\in[m]}$. Let $\boldsymbol{\mu}_{\text{sum}} = \sum_{i\in[m]} a_i\boldsymbol{\mu}_i$. The challenger provides a final ciphertext $\mathbf{c}^*$ to $\mathcal{A}$, who must distinguish whether it was generated by the real Add algorithm or a simulator.

**Game 0 (Real World):** This is the standard circuit privacy experiment. The challenger computes the final ciphertext $\mathbf{c}_0^*$ using the real HGSW.Add algorithm.

(1) For each scalar $a_i$, compute the re-randomization vector $\mathbf{x}_i^T \leftarrow g_{\text{rand}}^{-1}(a_i)$.
(2) Aggregate the secret vectors into $\mathbf{x}_{\text{agg}}^T = (\mathbf{x}_1^T, \ldots, \mathbf{x}_m^T)$.
(3) Aggregate public and error matrices as $\mathbf{A}_{\text{agg}}^T := [\mathbf{A}_1^T|\ldots|\mathbf{A}_m^T]$ and $\mathbf{E}_{\text{agg}}^T := [\mathbf{E}_1^T|\ldots|\mathbf{E}_m^T]$.
(4) Sample fresh masking noise $\mathbf{y}^T \leftarrow \mathcal{D}_{\sigma_y}^{l'd}$.
(5) Compute the public part $\mathbf{b}^T = \mathbf{x}_{\text{agg}}^T\mathbf{A}_{\text{agg}}$.
(6) Compute the accumulated error part $\mathbf{e}_{\text{total}}^T = \mathbf{x}_{\text{agg}}^T\mathbf{E}_{\text{agg}} + \mathbf{y}^T$.
(7) The final ciphertext is $\mathbf{c}_0^{*T} = [\mathbf{b}^T|\mathbf{b}^T\mathbf{S} + \mathbf{e}_{\text{total}}^T + \frac{q}{p}\boldsymbol{\mu}_{\text{sum}}]$.

The challenger returns $\mathbf{c}_0^*$ to $\mathcal{A}$.

**Game 1:** This game is identical to Game 0, except the challenger replaces the public part $\mathbf{b}^T$ with a vector $\mathbf{b}_{\text{unif}}^T$ sampled uniformly at random from $R_q^{1\times n'}$. The final ciphertext is $\mathbf{c}_1^{*T} = [\mathbf{b}_{\text{unif}}^T|\mathbf{b}_{\text{unif}}^T\mathbf{S} + \mathbf{e}_{\text{total}}^T + \frac{q}{p}\boldsymbol{\mu}_{\text{sum}}]$, where $\mathbf{e}_{\text{total}}^T$ is computed as in Game 0.

Lemma 3.3. *Under the CEK-MH-MLWE assumption, for any PPT adversary $\mathcal{A}$, the advantage in distinguishing Game 0 from Game 1 is negligible.*

Proof. We build a PPT reduction algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to break the CEK-MH-MLWE assumption. The reduction proceeds in two phases, following the structure of the CEK-MH-MLWE game (Definition 3.1).

$\mathcal{B}$ simulates the circuit privacy experiment for $\mathcal{A}$ as follows:

(1) **Phase 1:** $\mathcal{B}$ receives the CEK-MH-MLWE challenge matrix $\mathbf{A}_{chall}$ and hint matrix $\mathbf{M}_{chall}$ from its challenger. $\mathcal{B}$ samples the HGSW secret key $\mathbf{S}$ independently by itself. Note that $\mathbf{S}$ is *not* the CEK-MH-MLWE secret; the aggregated re-randomization vector $\mathbf{x}_{\text{agg}}$ corresponds to the CEK-MH-MLWE secret. $\mathcal{B}$ sets $\mathbf{A}_{\text{agg}} := \mathbf{A}_{chall}$ and $\mathbf{E}_{\text{agg}} := \mathbf{M}_{chall}$, and runs the Phase 1 adversary $\mathcal{A}_1$ on input $(pp, sk) := (\mathbf{A}_{\text{agg}}, (\mathbf{E}_{\text{agg}}, \mathbf{S}))$. $\mathcal{A}_1$ returns a message vector $\{\boldsymbol{\mu}_i\}_{i\in[m]}$.
(2) **Phase 2:** $\mathcal{B}$ computes the ciphertexts $\mathbf{C}_i := \mathrm{HGSW.Encrypt}(i, sk, \boldsymbol{\mu}_i) = [\mathbf{A}_i|\mathbf{A}_i\mathbf{S} + \mathbf{E}_i] + \frac{q}{p}\cdot[\mathbf{0}^{m_q\times n}|\boldsymbol{\mu}_{i,1}\mathbf{g}|\ldots|\boldsymbol{\mu}_{i,l'}\mathbf{g}]$ (note that $\mathcal{B}$ knows $(\mathbf{A}_{\text{agg}}, \mathbf{E}_{\text{agg}}, \mathbf{S})$), and provides $\{\mathbf{C}_i\}_{i\in[m]}$ to $\mathcal{A}$. The adversary responds with the scaling coefficients $\{a_i\}_{i\in[m]}$. Now $\mathcal{B}$ computes a coset vector $\mathbf{c}^T := (\mathbf{c}_1^T, \ldots, \mathbf{c}_m^T)$ satisfying $\mathbf{c}_i^T\cdot\mathbf{g} = a_i$ for $i \in [m]$, as in the HGSW.Add algorithm (see $g_{\text{rand}}^{-1}$ in Definition 2.6), and $\mathcal{B}$ returns coset $\mathbf{c}$ to its CEK-MH-MLWE challenger.
(3) The challenger returns $\mathbf{b}_{chall}$ to $\mathcal{B}$, where $\mathbf{b}_{chall}^T = \mathbf{x}_{\text{agg}}^T\mathbf{A}_{\text{agg}}$ (Real case) or $\mathbf{b}_{chall}^T$ is uniformly random (Random case). Note that $\mathbf{x}_{\text{agg}}$ is the CEK-MH-MLWE secret, sampled from $\mathcal{D}_{\Lambda_q^\perp(\mathbf{G}_{\text{agg}})+\mathbf{c},r_x}$.
(4) $\mathcal{B}$ constructs the final ciphertext $\mathbf{c}^{*T} = [\mathbf{b}_{chall}^T|\mathbf{b}_{chall}^T\mathbf{S}+\mathbf{e}_{\text{total}}^T + \frac{q}{p}\boldsymbol{\mu}_{\text{sum}}]$, where $\mathbf{e}_{\text{total}}^T = \mathbf{z}_{chall}^T$ is the hint from the CEK-MH-MLWE challenger (satisfying $\mathbf{z}_{chall}^T = \mathbf{x}_{\text{agg}}^T\mathbf{E}_{\text{agg}} + \mathbf{y}^T$ in the Real case). $\mathcal{B}$ sends $\mathbf{c}^*$ to $\mathcal{A}$.
(5) $\mathcal{A}$ outputs a bit $b'$. $\mathcal{B}$ outputs $b'$ as its guess.

In the Real CEK-MH-MLWE case ($\mathbf{b}_{chall}^T = \mathbf{x}_{\text{agg}}^T\mathbf{A}_{\text{agg}}$), the ciphertext $\mathbf{c}^{*T}$ is distributed exactly as in Game 0. In the Random case ($\mathbf{b}_{chall}^T$ uniform), the ciphertext is distributed as in Game 1. Therefore, any advantage $\mathcal{A}$ has in distinguishing Game 0 from Game 1 translates directly into the same advantage for $\mathcal{B}$ in breaking CEK-MH-MLWE. □

**Game 2 (Simulated World):** This game is identical to Game 1, except the accumulated noise part $\mathbf{e}_{\text{total}}^T$ is replaced with a simulated value $\mathbf{e}_{\text{sim}}^T$ drawn from the target statistical distribution defined in LUNA's LHLL (Theorem 2.12). This game corresponds to the output of the ideal simulator $\mathcal{S}$. The final ciphertext is $\mathbf{c}_2^{*T} = [\mathbf{b}_{\text{unif}}^T|\mathbf{b}_{\text{unif}}^T\mathbf{S} + \mathbf{e}_{\text{sim}}^T + \frac{q}{p}\boldsymbol{\mu}_{\text{sum}}]$.

Lemma 3.4. *If the HGSW parameters satisfy the conditions of Lemma 2.14 with $L := mm_q$, then Game 1 and Game 2 are statistically indistinguishable.*

Proof. In Game 1, the public part $\mathbf{b}_{\text{unif}}^T$ is uniform and independent of the secret $\mathbf{x}_{\text{agg}}$. The noise term is $\mathbf{e}_{\text{total}}^T = \mathbf{x}_{\text{agg}}^T\mathbf{E}_{\text{agg}} + \mathbf{y}^T$. To see that the preconditions of Lemma 2.14 are satisfied, note that in the notation of that lemma, $\bar{\mathbf{E}} := \begin{pmatrix}\mathbf{E}\\ \mathbf{I}_{l'}\end{pmatrix}$ is defined as a discrete Gaussian matrix $\mathbf{E}$ concatenated with the identity matrix. Writing $\bar{\mathbf{x}}^T := (\mathbf{x}_{\text{agg}}^T, \mathbf{y}^T)$, we have $\bar{\mathbf{x}}^T\bar{\mathbf{E}} = \mathbf{x}_{\text{agg}}^T\cdot\mathbf{E}_{\text{agg}} + \mathbf{y}^T$, which matches the form of the accumulated error $\mathbf{e}_{\text{total}}^T$. Therefore, Lemma 2.14 proves that, under its specified parameter conditions, the distribution of this accumulated noise is statistically close to the simulated Gaussian distribution $\mathbf{e}_{\text{sim}}^T$. Since statistical indistinguishability implies computational indistinguishability, no PPT adversary can distinguish Game 1 from Game 2. □

**Conclusion of Proof:** We have established a chain of indistinguishability:

$$\text{Game 0} \underset{c}{\approx} \text{Game 1} \underset{s}{\approx} \text{Game 2}$$

By a standard hybrid argument, the real world (Game 0) is computationally indistinguishable from the simulated world (Game 2). This means that for any PPT adversary $\mathcal{A}$, its advantage in the circuit privacy experiment is negligible, which completes the proof of computational circuit privacy. □

Remark 5 (Tightness of Reductions). *The reductions involved in Theorem 3.2 are tight when $\varepsilon = 2^{-\kappa}$ for security parameter $\kappa$, which*

*is the value of $\varepsilon$ we use for setting parameters. Explicitly, the reduction from MLWE to Circuit Privacy of HGSW loses an additive term $\varepsilon_{red}$ in advantage, where $\varepsilon_{red} \leq 5\varepsilon \leq 5 \cdot 2^{-\kappa}$ by Lemma 2.14. Similarly, the reduction in Lemma 4.1 loses an additive factor at most $\varepsilon \leq 2^{-\kappa}$ and a negligible multiplicative factor $(1-\varepsilon) \geq 1 - 2^{-\kappa}$.*

## 4 Reducing MH-MLWE to CEK-MH-MLWE

In our privacy analysis, the re-randomization vectors are sampled from a discrete Gaussian distribution over a specific coset of the gadget lattice $\Lambda_q^{\perp}(\mathbf{G}) + \mathbf{c}$, rather than a standard integer lattice. To strictly justify the use of the Matrix Hint-MLWE hardness assumption in this setting, we provide a reduction showing that the Matrix Hint-MLWE problem over a lattice coset is at least as hard as the standard Matrix Hint-MLWE problem over the integer lattice.

### 4.1 The Reduction

We recall the notation $\Lambda_q^{\perp}(\mathbf{H}) := \{\mathbf{v} \in R^l : \mathbf{H} \cdot \mathbf{v} = 0 \bmod q\}$ for a $q$-ary $R$-module (SIS) lattice with respect to a matrix $\mathbf{H} \in R_q^{v \times l}$.

Lemma 4.1 (Reduction: Standard to Coset Hint-MLWE). *Let $n, l, k, d, v_\Lambda$ be non-negative integers with*

$$v_\Lambda + 2 \leq n \leq l - (v_\Lambda + 1), \tag{7}$$

*$q$ a prime modulus, $\sigma_e, \sigma'_e$ be Gaussian deviation parameters for error distributions over $R^l$ and $\chi_s, \chi_e, \chi_y, \chi_M$ and $\chi'_e, \chi'_y, \chi'_M$ be distributions over $R^{l-n}, R^l, R^k, R^{k\times l}$ and $R^l, R^k, R^{k\times l}$ respectively, where $R := \mathbb{Z}[x]/(x^d+1)$, and $R_q := \mathbb{Z}_q[x]/(x^d+1)$. Moreover let $x^d + 1 = \prod_{i\in[r]} f_i(x) \bmod q$ denote the splitting of $x^d+1$ into $r$ irreducible polynomials over $\mathbb{Z}_q$, so that $R_q$ splits into $r$ fields $\mathbb{F}_i$ each of degree $d/r$ over $\mathbb{Z}_q$ for $i \in [r]$.*

*Let $0 < \varepsilon < 1/2$ be negligible in the security parameter $\lambda$.*

*Let $\mathbf{H} \in R_q^{v_\Lambda \times l}$ and let $\Lambda := \Lambda_q^{\perp}(\mathbf{H})$ denote a $q$-ary module lattice over $R$ with $\mathrm{rank}_{\mathbb{F}_i}(\mathbf{H}) = v_\Lambda$ for all $i \in [r]$ and a column $R$-basis $\mathbf{B}_\Lambda \in R^{l\times l}$. Assume there exists a polynomial time (in the security parameter $\lambda$) algorithm* GenΛ *that outputs $(\mathbf{H}, \mathbf{B}_\Lambda)$ for $\Lambda$ with basis spectral norm $\|\mathbf{B}_\Lambda\|$. Suppose the following conditions on the parameters are satisfied:*

- *Distribution relations:*

$$\chi_M := \chi'_M \cdot \mathbf{B}_\Lambda, \chi_y := \chi'_y \tag{8}$$

- *Sampleability condition:*

$$\sigma'_e \geq \sqrt{2\sigma_e^2 + \delta^2}\|\mathbf{B}_\Lambda\|, \tag{9}$$

  *where $\delta := \sqrt{\ln(2ld+4)/\pi}$.*
- *Uniformity conditions:*

$$4rnq^{-\frac{d}{r}(n-v_\Lambda)} \leq \varepsilon, \tag{10}$$

  *and*

$$4\sqrt{2^r q^{-\frac{d}{r}(l-(v_\Lambda+n))}} \leq \varepsilon. \tag{11}$$

*There exists a PPT reduction from the (Primal) Matrix Hint-MLWE decision problem* $\text{MH-MLWE}_{n,l,k,d,q,\chi_s,\sigma_e,\chi_y,\chi_M}$ *to the Coset Error Knapsack (Dual) Matrix Hint-MLWE problem* $\text{CEK-MH-MLWE}_{n,l,k,d,q,\sigma'_e,\chi'_y,\chi'_M,\Lambda}$. *Specifically, for any PPT adversary $\mathcal{A}$ against the CEK-MH-MLWE problem* $\text{CEK-MH-MLWE}_{n,l,k,d,q,\sigma'_e,\chi'_y,\chi'_M,\Lambda}$, *there exists a ppt adversary $\mathcal{B}$ against the (primal) Matrix Hint-MLWE problem* $\text{MH-MLWE}_{n,l,k,d,q,\chi_s,\sigma_e,\chi_y,\chi_M}$ *such that*

$$(1-\varepsilon) \cdot \mathrm{Adv}_{\mathcal{A}}^{\text{CEK-MH-MLWE}}(\lambda) \leq \mathrm{Adv}_{\mathcal{B}}^{\text{MH-MLWE}}(\lambda) + \varepsilon, \tag{12}$$

Proof. We provide a sketch of the proof here. The full proof is given in Appendix B. Given an input MH-MLWE instance $(\mathbf{A}, \mathbf{b}, \mathbf{M}, \mathbf{z})$, where $\mathbf{A} \leftarrow \mathcal{U}(R_q^{l\times(l-n)})$, $\mathbf{M} \leftarrow \chi_M$, $\mathbf{s} \leftarrow \chi_s$, $\mathbf{e} \leftarrow \chi_e := \mathcal{D}_{R^l,\sigma_e}$, and $\mathbf{y} \leftarrow \chi_y$ and $\mathbf{u} \leftarrow \mathcal{U}(R_q^l)$, with $\mathbf{b} = \mathbf{As} + \mathbf{e}$ in the Real case ($b = 0$) or $\mathbf{b} = \mathbf{u}$ in the Random case ($b = 1$), the algorithm $\mathcal{B}$ works as follows:

*Phase 1:* $\mathcal{B}$ first runs $\text{Gen}\Lambda(\lambda)$ to get $(\mathbf{H}_\Lambda, \mathbf{B}_\Lambda)$, computes $\hat{\mathbf{A}} := \mathbf{B}_\Lambda\mathbf{A} \bmod q$ and tests if $\hat{\mathbf{A}} \notin \text{GoodA}$, and in that case, $\mathcal{B}$ outputs a random bit $b' \leftarrow \mathcal{U}(\{0,1\})$. Here, GoodA denotes the set of matrices $\hat{\mathbf{A}} \in R_q^{l\times(l-n)}$ such that for all $i \in [r]$, $\hat{\mathbf{A}}_i := \hat{\mathbf{A}} \mod f_i \in \mathbb{F}_i^{l\times(l-n)}$ has full rank $l-n$ over the field $\mathbb{F}_i$.

- If $\hat{\mathbf{A}} \in \text{GoodA}$, $\mathcal{B}$ samples each of the $n$ rows of a matrix $\mathbf{A}' \in R_q^{n\times l}$ independently from the distribution $\chi_{A'} := \mathcal{U}(I_q^{\perp}(\hat{\mathbf{A}}))$ on $R_q^l$, where

$$I_q^{\perp}(\hat{\mathbf{A}}) := \{\bar{\mathbf{a}} \in R_q^l : \bar{\mathbf{a}}^T\hat{\mathbf{A}} = \mathbf{0} \bmod q\}.$$

  This distribution $\chi_{A'}$ is sampleable in PPT by linear algebra in each of the fields $\mathbb{F}_i$ for $i \in [r]$ and CRT.
- $\mathcal{B}$ computes $\mathbf{M}' := \mathbf{M} \cdot \mathbf{B}_\Lambda^{-1}$ (note that $\mathbf{M} \in R^{k\times l}$ by the assumption (8) on $\chi_M$).
- $\mathcal{B}$ runs $\mathcal{A}_1(\mathbf{A}', \mathbf{M}')$ to get the coset center $\mathbf{c} \in R^l$ and adversary state st.

*Phase 2:* $\mathcal{B}$ samples $\mathbf{d}_1 \leftarrow \mathcal{D}_{R^l,\sigma_e,-\mathbf{c}_0}$, where $\mathbf{c}_0 := \mathbf{B}_\Lambda^{-1} \cdot \mathbf{c}$, and $\mathbf{d}_4 \leftarrow \mathcal{D}_{\Lambda,\sqrt{\Sigma_{flood}},0}$, where $\Sigma_{flood} = (\sigma'_e)^2\mathbf{I} - \Sigma_3$ and $\Sigma_3 = 2\sigma^2\mathbf{B}_\Lambda\mathbf{B}_\Lambda^T$. Then it computes

- $\mathbf{b}' = \mathbf{A}' \cdot \mathbf{B}_\Lambda \cdot (\mathbf{b} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{d}_4 \bmod q$
- $\mathbf{z}' = \mathbf{z} + \mathbf{M} \cdot (\mathbf{b} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{M}' \cdot \mathbf{d}_4$
- $\mathcal{B}$ runs $\mathcal{A}_2(\mathbf{b}', \mathbf{z}', \text{st})$ to get $b' \in \{0,1\}$ and returns $b'$.

The leftover hash Lemma in rings and the fact that $l - (v_\Lambda + n)$ is sufficiently large (see (11) in Th. 4 of [8]) implies the uniformity of $\mathbf{b}'$ in Random case given $\mathbf{A}'$ (Game 1 of the full proof). Using linear algebra and statistical distance arguments (by excluding the proper subideals using the fact that $\bar{A}$ is not in BadA and $n - v_\Lambda$ sufficiently large (see (10)), we can conclude that $\mathbf{A}'$ is uniform (Games 2 and 3 of the full proof). Also (10) can be used to show that $\mathbf{A}$ that are not in BadA have a probability $\geq 1 - \epsilon$ negligibly close to 1. In order to transform MH-MLWE Instance into Real CEK-MH-MLWE instance, we first change the center of the Gaussian distribution (using $\mathbf{d}_1$) and later add the coset $\mathbf{c}$ using the shifted $\mathbf{c}_0$ and apply the same to both $\mathbf{A}$ and $\mathbf{M}$ to get $\mathbf{A}'$ and $\mathbf{M}'$. respectively, with the desired distributions (Game 4 in the detailed proof).

□

Remark 6. *The above Lemma is stated as general as possible as we believe can be of independent interest to applied cryptography community. Note that $v_\Lambda = m$ and $l = mm_q$ for our application in Luna+ with $\Lambda := \Lambda_q^{\perp}(\mathbf{G}^T)$, for gadget matrix $\mathbf{G}$. In this case, since the $\mathbf{B}_\Lambda$ when represented over $\mathbb{Z}$ is block diagonal with md blocks of dimension $m_q$, we can set $\delta := \sqrt{\ln(2m_q+4)/\pi}$ in* (9).

## 4.2 Parameter Constraints

To determine the optimal parameters for LUNA+, we must satisfy the constraints imposed by the reduction from Standard Matrix Hint-MLWE to the Coset Error Knapsack variant used in our scheme. These constraints ensure that the re-randomization noise is sufficient to hide the geometric structure of the lattice coset ("flooding") and that the public matrices are statistically indistinguishable from uniform.

*4.2.1 The Flooding Condition.* The reduction (Lemma 4.1) requires a "flooding" step to mask the lattice basis transformation. This imposes a lower bound on the target Gaussian width $\sigma'_e$ (used for re-randomization in LUNA+) relative to the standard MLWE parameter $\sigma_e$ (denoted here as $\sigma_{std}$) and the geometry of the gadget lattice basis $\mathbf{B}_\Lambda$.

Specifically, the covariance matrix $\Sigma_{flood} = (\sigma'_e)^2\mathbf{I} - 2\sigma_{std}^2\mathbf{B}_\Lambda\mathbf{B}_\Lambda^T$ must be positive definite. This implies:

$$\sigma'_e \geq \sqrt{2}\|\mathbf{B}_\Lambda\|_2 \cdot \sigma_{std}, \tag{13}$$

where $\|\mathbf{B}_\Lambda\|_2$ is the spectral norm of the basis.

In LUNA+, we denote the Gaussian width of the re-randomization vectors as $r_x$, where $r_x = \sigma'_e\sqrt{2\pi}$. We define the flooding factor $\gamma$ based on the spectral norm of the gadget basis, bounded by $\|\mathbf{B}_\Lambda\|_2 \leq \sqrt{2\beta(\beta-1)m_q}$ [25]:

$$\gamma := \sqrt{2}\|\mathbf{B}_\Lambda\|_2 \approx 2\sqrt{\beta(\beta-1)m_q}. \tag{14}$$

Using this factor, the flooding condition requires that the implemented noise $r_x$ be significantly larger than the underlying standard MLWE noise $r_{std} = \sigma_{std}\sqrt{2\pi}$:

$$r_x \geq \gamma \cdot r_{std} \implies \sigma_{std} \leq \frac{r_x}{\gamma\sqrt{2\pi}}. \tag{15}$$

This inequality ensures that the re-randomization noise is sufficiently large to "smudge" the distortions introduced by sampling secrets from the lattice coset.

*4.2.2 Uniformity Conditions.* To ensure that the reduction holds, specifically that the public matrix $\mathbf{A}'$ and the random sample $\mathbf{b}'$ in the CEK-MH-MLWE game are statistically close to uniform - the parameters must satisfy specific geometric and probability bounds derived in the reduction proof of Lemma 4.1. Namely, Let $n$ be the module dimension, $l = v_\Lambda \cdot m_q$ be the total dimension of the gadget matrix. The dimensions must satisfy $v_\Lambda + 2 \leq n \leq (m_q - 1) \cdot v_\Lambda$ according to (7). For a statistical security parameter $\kappa$ (e.g., $\kappa = 40$ or 64), the uniformity bounds in (10) and (11) must hold to ensure the statistical distance is at most $2^{-\kappa}$.

*4.2.3 Combined Security Constraint.* For the underlying Standard MH-MLWE problem to be hard, the parameters must satisfy the **Sampleability Condition** established in Lemma 2.11. By substituting the minimal standard deviation $\sigma_{std}$ permitted by the **Convolution Condition** ($\sigma_{std} \geq \sqrt{1+1/\delta} \cdot \eta_\epsilon$) into the Sampleability inequality, we derive a unified security constant $K$:

$$K := \frac{(1+\delta)^2}{\delta}\eta_\epsilon^2(\mathbb{Z}^{(m\cdot m_q)d}) + \delta_0, \tag{16}$$

where $\eta_\epsilon$ is the smoothing parameter of the underlying integer lattice, and $\delta, \delta_0$ are the reduction constants. The combined security constraint on the Gaussian width parameters $r_x$ and $r_y$ then becomes:

$$\frac{\gamma^2}{r_x^2} + \frac{B}{r_y^2} \leq \frac{1}{2\pi K}. \tag{17}$$

# 5 Deriving Optimal Parameters

We now solve the optimization problem to find the parameters that minimize proof size. Our goal is to minimize the total noise variance (which dictates the required modulus $q$) subject to the combined security constraint derived above.

## 5.1 Correctness Constraint

The modulus $q$ must be large enough to tolerate the noise accumulated during homomorphic evaluation.

To perform the optimization, we derive the high-probability bound $B$ on the squared spectral norm of the aggregated error matrix $\mathbf{E}_{\text{agg}}$ in Appendix C. This yields $B \approx (m \cdot m_q)l'd^2E_{max}^2$, where $N_{\text{coeffs}} = (m \cdot m_q)l'd$ is the total count of coefficients and $E_{max}$ is the maximum coefficient magnitude derived via the Gaussian tail bound.

Theorem 5.1 (Correctness of Noise Decoupled Additive Homomorphism). *Let $\lambda$ be the security parameter, and let $p, q, n, m_q, l', d, m, v$ be positive integers. Let $s$ be the Gaussian parameter for initial errors $\mathbf{E}_i$, and let $r_x, r_y$ be the Gaussian width parameters for the re-randomization vectors $\mathbf{x}_i$ and the masking noise $\mathbf{y}_j$, respectively. Assume $p, q, n, m_q, l', d, m, v, s, r_x, r_y = \text{poly}(\lambda)$ and $n > 8$. Let $\theta_{corr} := 2((m \cdot m_q + m/v)d + 1)/\epsilon_{corr}$ denote the union-bound factor over the $(m \cdot m_q + m/v)d + 1$ integer noise coordinates. If the ciphertext modulus $q$ satisfies*

$$q > 2ps\sqrt{\frac{(r_x^2(m \cdot m_q)d + r_y^2(m/v)d + 1)\ln\theta_{corr}}{\pi}}, \tag{18}$$

*then the HGSW scheme satisfies the Additive Homomorphism property for $m$ additions, with correctness error probability at most $\epsilon_{corr}$.*

To obtain the smallest possible $q$, we must minimize the noise variance function $f(r_x, r_y)$ inside the square root:

$$f(r_x, r_y) = r_x^2(m \cdot m_q)d + r_y^2(m/v)d + 1, \tag{19}$$

where $f(r_x, r_y)$ is the total noise variance function. We try to minimize this function:

$$f(r_x, r_y) = C_x r_x^2 + C_y r_y^2 + 1, \tag{20}$$

where the coefficients are determined by the scheme parameters:

$$C_x = (m \cdot m_q)d \quad \text{and} \quad C_y = (m/v)d.$$

Here, $m$ denotes the total number of homomorphic additions (corresponding to the number of rows in the LPCP query matrix), $d$ is the ring dimension, $m_q = \lceil \log_\beta q \rceil$ is the gadget dimension, and $v$ is the aggregation block size parameter used in the `HGSW.Add` algorithm (specifying that one fresh noise vector $\mathbf{y}$ is added for every $v$ re-randomized ciphertexts).

## 5.2 The Optimization Problem

We formulate the problem as a constrained minimization using the variance function and the security constraint (treated as an equality for the minimal solution).

**Minimize:**

$$f(r_x, r_y) = C_x r_x^2 + C_y r_y^2 + 1$$

**Subject to:**

$$g(r_x, r_y) = \frac{\gamma^2}{r_x^2} + \frac{B}{r_y^2} - C_{sec} = 0$$

where we define the constant $C_{sec} = \frac{1}{2\pi K}$.

Let $X = r_x^2$ and $Y = r_y^2$. We define the Lagrangian $\mathcal{L}(X, Y, \lambda)$:

$$\mathcal{L} = C_x X + C_y Y - \lambda \left( \frac{\gamma^2}{X} + \frac{B}{Y} - C_{sec} \right). \tag{21}$$

Taking the partial derivatives and setting them to zero:

$$\frac{\partial \mathcal{L}}{\partial X} = C_x + \lambda \frac{-\gamma^2}{X^2} = 0 \implies \lambda = \frac{C_x X^2}{\gamma^2} \tag{22}$$

$$\frac{\partial \mathcal{L}}{\partial Y} = C_y + \lambda \frac{-B}{Y^2} = 0 \implies \lambda = \frac{C_y Y^2}{B} \tag{23}$$

Equating the expressions for $\lambda$ yields the relationship between the optimal $X$ and $Y$:

$$\frac{C_x X^2}{\gamma^2} = \frac{C_y Y^2}{B} \implies Y = X \cdot \sqrt{\frac{BC_x}{\gamma^2 C_y}} = X \cdot \frac{\sqrt{BC_x}}{\gamma \sqrt{C_y}} \tag{24}$$

Substituting this expression for $Y$ (in terms of $X$) back into the security constraint $\frac{\gamma^2}{X} + \frac{B}{Y} = C_{sec}$ and solving for $X$ gives the optimal squared re-randomization noise:

$$r_{x,opt}^2 = X = \frac{\gamma}{C_{sec}\sqrt{C_x}} \left( \gamma \sqrt{C_x} + \sqrt{BC_y} \right). \tag{25}$$

Consequently, the optimal squared masking noise is:

$$r_{y,opt}^2 = Y = \frac{\sqrt{B}}{C_{sec}\sqrt{C_y}} \left( \gamma \sqrt{C_x} + \sqrt{BC_y} \right). \tag{26}$$

Finally, we determine the minimum achievable noise variance by substituting the optimal values $r_{x,opt}^2$ and $r_{y,opt}^2$ back into the objective function $f(r_x, r_y)$:

$$f_{min} - 1 = C_x r_{x,opt}^2 + C_y r_{y,opt}^2 = \frac{1}{C_{sec}} \left( \gamma \sqrt{C_x} + \sqrt{BC_y} \right)^2. \tag{27}$$

Substituting back $C_{sec} = \frac{1}{2\pi K}$, we arrive at the final analytical bound for the minimum noise variance:

$$f_{min} = 2\pi K \left( \gamma \sqrt{C_x} + \sqrt{BC_y} \right)^2 + 1. \tag{28}$$

This value $f_{min}$ is plugged into the correctness inequality (18) to solve for the minimal valid modulus $q$. Unlike the original LUNA analysis, this minimum explicitly accounts for the flooding factor $\gamma$, balancing the costs of coset sampling against the relaxation provided by the computational hint assumption.

## 5.3 Concrete Parameters

Table 2 presents the concrete parameters for LUNA+ alongside the original LUNA [29] and ISW [19] schemes. All parameter sets target a 128-bit post-quantum security level and support an R1CS instance size of $2^{16}$. Our parameters are chosen to match the root-Hermite factor used in [29] ($\delta \approx 1.00427$). The results demonstrate a substantial improvement in concrete succinctness. By shifting from a statistical privacy argument to a computational one based on MH-MLWE, LUNA+ avoids the large "smudging" noise required by the LHLL.

*Noise Parameters:* LUNA scheme required a large re-randomization noise width of $\log r \approx 26.73$ to mask the witness structure. In contrast, our computational analysis allows for significantly smaller, decoupled parameters. For our most succinct parameter set ($p = 37, n = 48$), we achieve $\log r_x \approx 14.21$ and $\log r_y \approx 18.59$.

*Modulus and Dimension Reduction:* This reduction in noise variance directly impacts the required ciphertext modulus $q$. As shown in Table 2, $\log q$ drops from $\approx 49.54$ in LUNA to $\approx 37.11$ in LUNA+. Consequently, to maintain the same security level, the lattice dimension $n$ can be reduced from 64 to 48.

*Proof and CRS Size:* These reductions cascade into the final artifact sizes. LUNA+ achieves a proof size of **4.22 KB**, representing a **25% reduction** compared to LUNA's 5.60 KB. Most notably, the compressed CRS size achieves the biggest improvement, dropping drastically from 2.06 GB to **0.54 GB**. This represents a massive reduction of approximately **74%**, making the scheme significantly more practical for bandwidth-constrained environments.

We also compare our results against the recent work of Ahmadi et al. [1], which constructs a DV-zkSNARK relying solely on the standard Module-SIS assumption. While their construction achieves a Full CRS size of 1.3 GB (comparable to the uncompressed CRS of our optimized variants), their reliance on standard assumptions necessitates a significantly larger proof size of 20.75 KB. In contrast, by leveraging the MH-MLWE assumption, LUNA+ achieves proof sizes between 4.09 KB and 5.31 KB, representing a roughly 4× to 5× improvement in proof succinctness.

# 6 Implementation and Evaluation

We implemented our LUNA+ scheme, building upon the HGSW implementation framework of the original LUNA [29], to evaluate its concrete performance and validate the theoretical improvements stemming from our computational privacy analysis. Our implementation uses C++ with the PALISADE library [27] for underlying lattice cryptography operations, including NTT-based polynomial arithmetic and discrete Gaussian sampling. All benchmarks were performed on a machine equipped with an Intel Xeon E-2314 CPU @ 2.8 GHz (Turbo Boost disabled) and 128 GB RAM, running Linux Mint 21.2, compiled with gcc 11.4.0 (-O3). We compare LUNA+ against the original LUNA [29] and ISW [19] schemes, using the parameters derived in Section 5 (summarized in Table 2) targeting 128-bit post-quantum security. For ISW and LUNA, we use the performance figures reported in [29], which were benchmarked on the same hardware setup.

*Implementation Parameters.* Our benchmarks for LUNA+ use the parameter set corresponding to the last row of Table 2. Specifically, we set the plaintext modulus $p = 19$, the ring dimension $d = 32$, and the module dimension $n = 53$. The primary ciphertext modulus is set to $\log q \approx 40.92$, and the modulus after switching is $\log q' \approx 19.56$. The decoupled Gaussian width parameters for re-randomization are $(\log r_x, \log r_y) \approx (20.33, 20.84)$. We set the gadget base $\beta = 12784$, leading to a gadget dimension of $m_q = \lceil \log_\beta q \rceil = 3$. The initial error parameter is $s = 2$, the number of ring splitting factors mod $p$

**Table 2: Parameter Comparison for 128-bit Security and R1CS size $2^{16}$.**

| Scheme | $p$ | $n$ | $d$ | $\log q$ | $\log q'$ | $\log r$ / $(\log r_x, \log r_y)$ | Proof (KB) | CRS (GB) | Full CRS (GB) |
|---|---|---|---|---|---|---|---|---|---|
| ISW [19] | $2^{13}-1$ | 3600 | 2 | 186 | 36 | 32.64 | 0.66 | 21 | – |
| LUNA [29] | 37 | 64 | 32 | 49.54 | 20.79 | 26.73 | 5.6 | 2.06 | 28.37 |
| | 37 | 66 | 32 | 51.35 | 20.84 | 28.93 | 5.78 | 1.25 | 17.8 |
| | 277 | 83 | 32 | 64.7 | 24.07 | 40.02 | 8.28 | 0.63 | 11.12 |
| | 19 | 81 | 32 | 62.98 | 20.17 | 40.17 | 6.93 | 0.86 | 10.82 |
| Ahmadi et al. [1] | – | – | 16 | 32 | – | – | 20.75 | – | 1.3 |
| LUNA+ (This Work) | 37 | 48 | 32 | 37.11 | 20.38 | (14.21, 18.59) | 4.22 | 0.54 | 5.76 |
| | 37 | 49 | 32 | 37.44 | 20.41 | (15.03, 18.69) | 4.31 | 0.46 | 4.94 |
| | 277 | 52 | 32 | 40.19 | 23.4 | (14.56, 18.68) | 5.21 | 0.59 | 6.71 |
| | 277 | 53 | 32 | 40.71 | 23.43 | (15.54, 18.87) | 5.31 | 0.50 | 5.76 |
| | 19 | 47 | 32 | 36.53 | 19.39 | (14.97, 18.84) | 4.09 | 0.62 | 4.82 |
| | 19 | 53 | 32 | 40.92 | 19.56 | (20.33, 20.84) | 4.58 | 0.42 | 3.6 |

is $l_p = 2$ (giving a plaintext field of degree $f = d/l_p = 16$), and the sparsification parameter is $\tau = 1$.

| Component | R1CS Instance Size | | |
|---|---|---|---|
| | $2^{10}$ | $2^{12}$ | $2^{14}$ |
| ISW Setup+Encrypt | 33 | 62 | 174 |
| LUNA Setup+Encrypt | 2.55 | 10.87 | 43.21 |
| LUNA+ Setup+Encrypt | 1.56 | 6.25 | 24.97 |
| ISW Add | 0.26 | 0.75 | 2.71 |
| LUNA Add | 0.32 | 1.41 | 5.67 |
| LUNA+ Add | 0.23 | 0.92 | 3.71 |
| ISW Decrypt | 0.00300 | 0.00300 | 0.00300 |
| LUNA Decrypt | 0.00085 | 0.00085 | 0.00085 |
| LUNA+ Decrypt | 0.00059 | 0.00059 | 0.00059 |

**Table 3: Homomorphic encryption runtimes for ISW [19], LUNA and LUNA+ (in seconds) at 128-bit security level.**

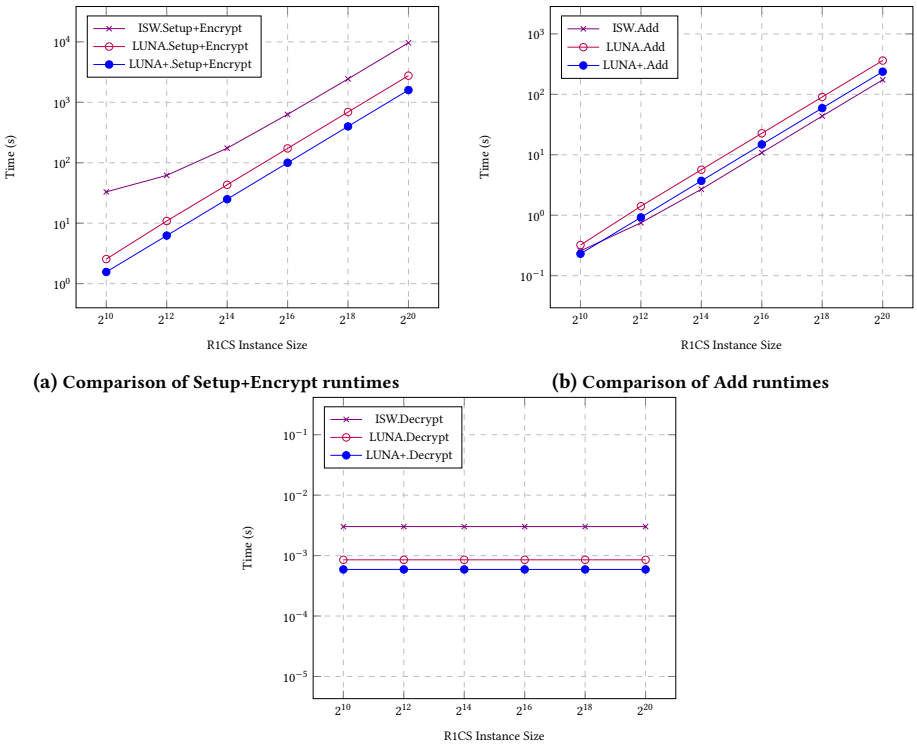
The performance results, presented in Figure 1 and Tables 3, confirming the significant concrete advantages offered by LUNA+.

*Setup and Encryption* - `Setup+Encrypt`. Figure 1(a) and Table 3 show the combined time for CRS setup and the initial encryption of LPCP queries. LUNA+ consistently outperforms LUNA, achieving a **1.63×–1.73×** speedup across the evaluated R1CS instance sizes. Both schemes are substantially faster than ISW, particularly for larger instances. This advantage arises because the computational privacy analysis based on MH-MLWE allows LUNA+ to use significantly smaller system parameters (modulus $q$ and dimension $n$) compared to the statistical LHLL analysis in LUNA, which required larger parameters (especially $q$) to achieve sufficient statistical smoothing noise, $r$. The smaller parameters directly lead to faster polynomial arithmetic (NTTs) and fewer operations overall during the encryption phase, which dominates the setup cost (Table 3, showing Encrypt time reduction).

*Homomorphic Addition* - `Add`. LUNA+ demonstrates a speedup of **1.39×–1.53×** compared to LUNA, and even surpasses ISW's Regev-based addition at $N_g = 2^{10}$, with a 1.13× speedup. However, for larger $N_g$ values, it incurs a slowdown of up to 1.36× compared to ISW. The HGSW `Add` procedure involves discrete Gaussian sampling for re-randomization ($g^{-1}_{\text{rand}}$) and masking ($\mathbf{y}$), operations not present in ISW's simpler vector additions. However, LUNA+ benefits from its smaller parameters $(n, q)$ compared to LUNA, slightly reducing the cost of these operations and the underlying polynomial multiplications, explaining its edge over LUNA. The crucial finding is that the computational privacy approach, despite requiring Gaussian sampling, permits parameters $(r_x, r_y)$ that are small enough ($\log r_x \approx 20.33$, $\log r_y \approx 20.84$) to yield prover performance competitive with ISW, unlike the large statistical parameter ($\log r \approx 27 - 40$) needed in LUNA which likely contributed to its slower `Add` time relative to LUNA+. The prover time scales linearly with the R1CS size ($N_g$) for all schemes.

*Decryption* - `Decrypt`. LUNA+ achieves decryption times around 0.59ms, which is roughly **1.44×** faster than LUNA (≈ 0.85ms) and over 5.08× faster than ISW (≈ 3ms). The HGSW decryption involves an inner product and rounding, inherently faster than Regev decryption. LUNA+'s advantage over LUNA is due to the smaller dimension $n$, which reduces the cost of the inner product $\mathbf{b}^T\mathbf{S}$. Most importantly, the verification time is independent of the R1CS instance size, making LUNA+ highly attractive for scenarios with resource-constrained verifiers or applications demanding constant verification overhead.

The empirical results strongly validate our approach. By replacing the statistical LHLL privacy argument of LUNA with a computational one based on Matrix Hint-MLWE hardness and optimizing the decoupled noise parameters $(r_x, r_y)$, LUNA+ achieves significant concrete gains. The ability to use much smaller re-randomization




**(a) Comparison of Setup+Encrypt runtimes**

**(b) Comparison of Add runtimes**

**(c) Comparison of Decrypt runtimes**

**Figure 1: Runtime comparison between ISW [19], LUNA [29], and LUNA+.**

noise parameters compared to the statistical requirement directly leads to smaller moduli $(q, q')$ and dimension $(n)$. This cascade results in faster setup, faster verification (5× faster than ISW), competitive prover times (only 1.36× slower than ISW, but faster than LUNA), and most notably, considerably shorter proofs (24-40% smaller than LUNA) and smaller compressed/full CRS (40-80% smaller than LUNA). LUNA+ therefore offers a compelling trade-off, advancing the practicality of post-quantum DV ZK-SNARGs, especially for applications prioritizing verifier efficiency and communication succinctness.

## 7 Conclusion

In this work, we presented LUNA+, a refinement of the LUNA designated-verifier ZK-SNARG scheme that achieves significant improvements in concrete succinctness while maintaining quasi-optimal asymptotic proof length. We addressed the primary efficiency bottleneck of the original LUNA construction: its reliance on a statistical privacy argument via LHLL, which necessitated large "smudging" noise parameters to ensure statistical uniformity.

Our core contribution is a paradigm shift to a *computational* circuit privacy analysis based on a new variant of the Matrix Hint-Module Learning With Errors (MH-MLWE) problem. A key theoretical advancement in this work is the formalization of the Coset Error Knapsack MH-MLWE (CEK-MH-MLWE) problem and a reduction demonstrating that its hardness is implied by the standard MLWE assumption, under appropriate parameters. This result validates the security of our HGSW re-randomization technique, where secrets are sampled from a lattice coset, without requiring the prohibitive noise levels needed for statistical indistinguishability.

Leveraging this computational foundation, we introduced a parameter optimization technique that decouples the re-randomization noise $(r_x)$ from the fresh masking noise $(r_y)$. By solving the resulting optimization problem, we derived minimal noise parameters that satisfy the security constraints of the CEK-MH-MLWE reduction while maximizing efficiency.

Compared to the original LUNA at the 128-bit security level, our scheme reduces the proof size by approximately 25% and achieves a dramatic reduction in the compressed CRS size of up to 73%. These succinctness gains are accompanied by practical performance improvements, including a $\approx 1.73\times$ speedup in setup and a 1.44× speedup in decryption.

LUNA+ demonstrates that computational hardness assumptions can effectively replace expensive statistical bounds in lattice-based privacy protocols. As our reduction for the Coset Error MH-MLWE problem is general, future work may explore its application to other cryptographic primitives that rely on discrete Gaussian sampling over lattice cosets, potentially yielding similar efficiency gains in broader contexts.

*Comparison with Non-Designated-Verifier Schemes.* While this work targets the designated-verifier (DV) preprocessing model, we now concretely position LUNA+ against state-of-the-art post-quantum *publicly verifiable* ZK arguments, which achieves public verifiability via the Fiat–Shamir transform in the (Q)ROM and therefore require no trusted setup or secret verification key. The trade-off

is that they give up the succinctness advantages that the DV setting enables.

*Lattice-based publicly verifiable arguments.* LaBRADOR [4] is the most compact lattice-based publicly verifiable argument for R1CS to date. For R1CS modulo $2^{64}+1$ it produces proofs of 47 KB at $2^{10}$ constraints and 58 KB at $2^{20}$ constraints, growing only logarithmically in the instance size. Compared against LUNA+'s 4.22 KB proof, LaBRADOR is therefore roughly 11× larger at $N_g = 2^{10}$ and 13.7× larger at $N_g = 2^{20}$. LaBRADOR's verifier runs in time linear in the R1CS size, whereas LUNA+'s verifier is constant-time ($\approx 0.67$ ms, independent of $N_g$, cf. Sec. 6). Greyhound [26] is a lattice-based polynomial commitment scheme that, when composed with LaBRADOR, yields a publicly verifiable argument with a sublinear verifier; it reports 53 KB evaluation proofs even for degree-$2^{30}$ polynomials under standard Module-SIS, roughly 12.6× larger than our proofs at a much larger problem scale. The base lattice NIZK framework of Lyubashevsky, Nguyen, and Plançon [24], used as the sub-protocol for short-pre-image relations inside LaBRADOR, is not itself succinct (proof size is linear in the witness dimension) and is therefore not competitive with LUNA+ as a SNARG for large R1CS instances.

*Hash-based transparent arguments.* Aurora [3] is a canonical transparent post-quantum SNARG in the ROM; its proof length stays below 130 KB even for several million R1CS constraints, but its prover runs in $O(n \log n)$ field operations and its verifier in $O(n)$, scaling linearly with the circuit. At $N_g = 2^{20}$ this is roughly 30× larger than LUNA+ and loses the constant-time verifier that makes LUNA+ attractive for resource-constrained verifiers.

DV lattice SNARGs in the preprocessing model remain the only known post-quantum route to sub-10-KB proofs with sub-millisecond, instance-size-independent verification. Publicly verifiable lattice and hash-based schemes trade an order-of-magnitude (or more) in proof size, and give up the constant verifier time of LUNA+, in exchange for eliminating trusted setup and the secret verification key. LUNA+ is therefore best deployed in settings where a trusted verifier is already assumed (e.g., a server validating many client proofs, or a single-verifier outsourced-computation workload) and where verifier latency and communication dominate the cost profile. A full empirical comparison of prover wall-clock times against LaBRADOR and Greyhound, using their reference implementations on matched hardware, remains interesting future work.

## Ethical Considerations

The work presented herein concentrates on advancing cryptographic theory and efficiency, specifically targeting the reduction of communication and computational costs in post-quantum zero-knowledge proof systems via novel cryptographic techniques. Our methodology relies purely on theoretical analysis, cryptographic proofs, and simulations. Any experimental validation was performed using synthetically generated data. No experiments involved accessing or processing real-world user information, interacting with live systems, or engaging human subjects. Consequently, this research does not directly pose risks such as undermining existing computer security systems, enabling the illicit collection of private data, or violating human rights.

## Acknowledgments

This work was partially supported by an Australian Research Council Discovery Grants DP250100229 and DP260100245.

## Open Science

To support reproducibility and facilitate future research, we have made the implementation of our HGSW protocol with the new parameters, including the benchmarking scripts publicly available, and is hosted athttps://zenodo.org/records/21736189.

## Generative AI Usage

We acknowledge the use of Google Gemini during the writing process of this manuscript. The tool was employed exclusively for the purpose of refining sentence structure, correcting grammatical errors, and improving the readability of the prose. No part of the scientific contribution, including the cryptographic proofs, protocol design, or experimental analysis, was generated by AI tools. The authors reviewed all AI-suggested edits for accuracy and retain full responsibility for the final content of the paper.

## References

[1] Ahmadi, M.S., Eghlidos, T., Abdolmaleki, B., Nguyen, N.K.: A lattice-based designated verifier zkSNARK from standard assumptions. Cryptology ePrint Archive, Paper 2025/2099 (2025), https://eprint.iacr.org/2025/2099

[2] Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized Anonymous Payments from Bitcoin. In: 2014 IEEE Symposium on Security and Privacy. pp. 459–474. IEEE, San Jose, CA (May 2014). https://doi.org/10.1109/SP.2014.36, https://ieeexplore.ieee.org/document/6956581/

[3] Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. In: Advances in Cryptology – EUROCRYPT 2019. pp. 103–128 (2019). https://doi.org/10.1007/978-3-030-17653-2_4, ePrint 2018/828

[4] Beullens, W., Seiler, G.: LaBRADOR: Compact proofs for R1CS from module-SIS. In: Advances in Cryptology – CRYPTO 2023. pp. 518–548 (2023). https://doi.org/10.1007/978-3-031-38554-4_17, ePrint 2022/1341

[5] Bitansky, N., Chiesa, A., Ishai, Y., Ostrovsky, R., Paneth, O.: Succinct non-interactive arguments via linear interactive proofs. J. Cryptol. **35**(3) (Jul 2022). https://doi.org/10.1007/s00145-022-09424-4, https://doi.org/10.1007/s00145-022-09424-4

[6] Boneh, D., Chen, B.: Latticefold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In: Tauman Kalai, Y., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025. pp. 327–361. Springer Nature Switzerland, Cham (2025)

[7] Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Lattice-Based SNARGs and Their Application to More Efficient Obfuscation. In: Coron, J.S., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017, vol. 10212, pp. 247–277. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_9, http://link.springer.com/10.1007/978-3-319-56617-7_9, series Title: Lecture Notes in Computer Science

[8] Boudgoust, K., Lapiha, O.: Leftover hash lemma(s) over cyclotomic rings. Cryptology ePrint Archive, Paper 2025/1080 (2025), https://eprint.iacr.org/2025/1080

[9] Debris-Alazard, T., Fallahpour, P., Stehl'e, D.: Lattice-based quantum advantage from rotated measurements. In: Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC). pp. 1320–1328 (2024). https://doi.org/10.1145/3618260.3649624

[10] Esgin, M.F., Steinfeld, R., Tairi, E., Xu, J.: LeOPaRd: Towards practical post-quantum oblivious PRFs via 2HashDH paradigm. Cryptology ePrint Archive, Paper 2024/1615 (2024), https://eprint.iacr.org/2024/1615

[11] Esgin, M.F., Zhao, R.K., Steinfeld, R., Liu, J.K., Liu, D.: MatRiCT: Efficient, Scalable and Post-Quantum Blockchain Confidential Transactions Protocol. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 567–584. ACM, London United Kingdom (Nov 2019). https://doi.org/10.1145/3319535.3354200, https://dl.acm.org/doi/10.1145/3319535.3354200

[12] Espitau, T., Niot, G., Prest, T.: Flood and Submerse: Distributed Key Generation and Robust Threshold Signature from Lattices. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024, vol. 14926, pp. 425–458. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-68394-7_14,

https://link.springer.com/10.1007/978-3-031-68394-7_14, series Title: Lecture Notes in Computer Science
[13] Garreta, A., Lipmaa, H., Luhaäär, U., Osadnik, M.: Cyclo: Lightweight lattice-based folding via partial range checks. In: Daemen, J., Thomé, E. (eds.) Advances in Cryptology – EUROCRYPT 2026. pp. 33–62. Springer Nature Switzerland, Cham (2026)
[14] Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic Span Programs and Succinct NIZKs without PCPs. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology – EUROCRYPT 2013, vol. 7881, pp. 626–645. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_37, http://link.springer.com/10.1007/978-3-642-38348-9_37, series Title: Lecture Notes in Computer Science
[15] Gennaro, R., Minelli, M., Nitulescu, A., Orr'u, M.: Lattice-Based zk-SNARKs from Square Span Programs. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 556–573. ACM, Toronto Canada (Oct 2018). https://doi.org/10.1145/3243734.3243845, https://dl.acm.org/doi/10.1145/3243734.3243845
[16] Gentry, C., Sahai, A., Waters, B.: Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013, vol. 8042, pp. 75–92. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5, http://link.springer.com/10.1007/978-3-642-40041-4_5, series Title: Lecture Notes in Computer Science
[17] Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems. In: Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing. p. 291–304. STOC '85, Association for Computing Machinery, New York, NY, USA (1985). https://doi.org/10.1145/22145.22178, https://doi.org/10.1145/22145.22178
[18] Groth, J.: On the Size of Pairing-Based Non-interactive Arguments. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, vol. 9666, pp. 305–326. Springer Berlin Heidelberg, Berlin, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_11, http://link.springer.com/10.1007/978-3-662-49896-5_11, series Title: Lecture Notes in Computer Science
[19] Ishai, Y., Su, H., Wu, D.J.: Shorter and Faster Post-Quantum Designated-Verifier zkSNARKs from Lattices. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. pp. 212–234. ACM, Virtual Event Republic of Korea (Nov 2021). https://doi.org/10.1145/3460120.3484572, https://dl.acm.org/doi/10.1145/3460120.3484572
[20] Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing - STOC '92. pp. 723–732. ACM Press, Victoria, British Columbia, Canada (1992). https://doi.org/10.1145/129712.129782, http://portal.acm.org/citation.cfm?doid=129712.129782
[21] Kim, D., Lee, D., Seo, J., Song, Y.: Toward Practical Lattice-Based Proof of Knowledge from Hint-MLWE. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, vol. 14085, pp. 549–580. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-38554-4_18, https://link.springer.com/10.1007/978-3-031-38554-4_18, series Title: Lecture Notes in Computer Science
[22] Langlois, A., Stehl'e, D.: Worst-case to average-case reductions for module lattices. Des. Codes Cryptogr. **75**(3), 565–599 (Jun 2015). https://doi.org/10.1007/s10623-014-9938-4, http://link.springer.com/10.1007/s10623-014-9938-4
[23] Lapiha, O., Prest, T.: A lattice-based IND-CCA threshold KEM from the BCHK+ transform. Cryptology ePrint Archive, Paper 2025/1958 (2025), https://eprint.iacr.org/2025/1958
[24] Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Advances in Cryptology – CRYPTO 2022. pp. 71–101 (2022). https://doi.org/10.1007/978-3-031-15979-4_3, ePrint 2022/284
[25] Micciancio, D., Peikert, C.: Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012, vol. 7237, pp. 700–718. Springer Berlin Heidelberg, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41, http://link.springer.com/10.1007/978-3-642-29011-4_41, series Title: Lecture Notes in Computer Science
[26] Nguyen, N.K., Seiler, G.: Greyhound: Fast polynomial commitments from lattices. In: Advances in Cryptology – CRYPTO 2024. pp. 243–275 (2024). https://doi.org/10.1007/978-3-031-68403-6_8, ePrint 2024/1293
[27] Polyakov, Y., Rohloff, K., Ryan, G.W., Cousins, D.: PALISADE lattice cryptography library. https://gitlab.com/palisade/palisade-release (2022), release 1.11.9
[28] Salmond, D., Grant, A., Grivell, I., Chan, T.: On the rank of random matrices over finite fields (2016), https://arxiv.org/abs/1404.3250
[29] Steinfeld, R., Sakzad, A., Esgin, M.F., Kuchta, V., Yassi, M., Zhao, R.K.: Luna: Quasi-optimally succinct designated-verifier zero-knowledge arguments from lattices. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. p. 3167–3181. CCS '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3658644.3670345, https://doi.org/10.1145/3658644.3670345
[30] Walfish, M., Blumberg, A.J.: Verifying computations without reexecuting them. Commun. ACM **58**(2), 74–84 (Jan 2015). https://doi.org/10.1145/2641562, https://doi.org/10.1145/2641562
[31] Walfish, M., Blumberg, A.J.: Verifying computations without reexecuting them. Commun. ACM **58**(2), 74–84 (Jan 2015). https://doi.org/10.1145/2641562, https://doi.org/10.1145/2641562
[32] Wang, H., Steinfeld, R., Saarinen, M.J.O., Esgin, M.F., Yiu, S.M.: mmCipher: Batching post-quantum public key encryption made bandwidth-optimal. Cryptology ePrint Archive, Paper 2025/1000 (2025), https://eprint.iacr.org/2025/1000

# A Designated Verifier from LPCP and LOVE

## A.1 Designated-Verifier ZK-SNARGs from Linear PCPs

Succinct Non-interactive ARguments (SNARGs) enable a prover to convince a verifier of the validity of an NP statement with a proof that is significantly shorter than the associated witness. Zero-Knowledge SNARGs (ZK-SNARGs) additionally ensure that the proof reveals nothing about the witness beyond the validity of the statement. Our work focuses on ZK-SNARGs in the designated-verifier (DV) preprocessing model, following the compiler framework of Bitansky et al. [5], instantiated with lattice-based cryptography [7, 19, 29].

*Definition A.1 (Designated-Verifier ZK-SNARG (Preprocessing Model) [7, 19]).* Let $\mathcal{L}$ be an NP language with relation $R_{\mathcal{L}}$. A designated-verifier ZK-SNARG for $\mathcal{L}$ consists of a tuple of PPT algorithms $\Pi_{\text{SNARG}} = (\text{Setup}, \text{Prove}, \text{Verify})$:

- **Setup**$(1^\lambda, 1^N)$: On input security parameter $\lambda$ and circuit size bound $N$, outputs a common reference string $crs$ and a secret verification key $vk$.
- **Prove**$(crs, x, w)$: On input $crs$, statement $x$, and witness $w$ such that $(x, w) \in R_{\mathcal{L}}$, outputs a proof $\pi$.
- **Verify**$(vk, x, \pi)$: On input $vk$, statement $x$, and proof $\pi$, outputs 1 (accept) or 0 (reject).

The scheme must satisfy completeness, soundness (or knowledge soundness for SNARKs), and zero-knowledge against the designated verifier. Proofs must be succinct (polylogarithmic in $N$).

The compiler framework [5, 7] constructs such a SNARG using two main components:

(1) **Linear Probabilistically Checkable Proof (LPCP):** An information-theoretic proof system where the verifier makes linear queries to a proof string provided by the prover. Efficient LPCPs exist for languages like R1CS [14, 19].
(2) **Linear-Only Vector Encryption:** A secret-key encryption scheme that supports homomorphic evaluation of linear functions but prevents non-linear operations.

The ZK-SNARG works as follows:

- **Setup:** The designated verifier generates keys $(pp, sk)$ for the vector encryption scheme. It then obtains the linear query matrix $\mathbf{Q}$ from the LPCP setup for the target circuit size $N$. It encrypts the rows $\mathbf{q}_i$ of $\mathbf{Q}$ using $sk$ to get ciphertexts $\mathbf{C}_i = \text{Encrypt}(sk, \mathbf{q}_i)$. The $crs$ consists of $(pp, \{\mathbf{C}_i\})$ and

the verification key $vk$ contains $sk$ and LPCP verification information.

- **Prove:** Given a witness $w$, the prover first computes the LPCP proof vector $\boldsymbol{\pi}$. It then homomorphically evaluates the linear query function using the encrypted query rows from the $crs$: $\mathbf{C}^* \leftarrow \text{Add}(pp, \{\mathbf{C}_i\}, \{\pi_i\})$. The SNARG proof is $\pi = \mathbf{C}^*$.
- **Verify:** The verifier uses $sk$ to decrypt $\mathbf{C}^*$ to get the LPCP answer vector $\mathbf{a} = \text{Decrypt}(sk, \mathbf{C}^*)$. It then runs the LPCP verification on $\mathbf{a}$ and accepts if the LPCP accepts.

For this construction to be a ZK-SNARG, the vector encryption scheme must satisfy certain properties, notably additive homomorphism, CPA security, circuit privacy (to hide the prover's coefficients $\pi_i$), and the linear-only property (for soundness). Our Half-GSW scheme, with the computational circuit privacy analysis based on Matrix Hint-MLWE, provides an efficient lattice-based instantiation of the required vector encryption.

## A.2 Linear-Only Vector Encryption for ZK-SNARGs

The LUNA framework, following [5], builds a ZK-SNARG by compiling a Linear Probabilistically Checkable Proof (LPCP) with a linear-only vector encryption scheme. We formalize the required properties below, adapted from [19, 29].

*Definition A.2 (Vector Encryption Scheme).* Let $\mathbb{F}$ be a finite field. A secret-key additively-homomorphic vector encryption scheme over a vector space $\mathbb{F}^l$ consists of a tuple of PPT algorithms $\Pi_{\text{VE}} = (\text{Setup}, \text{Encrypt}, \text{Decrypt}, \text{Add})$:

- $\text{Setup}(1^\lambda, 1^l)$: On input a security parameter $\lambda$ and a plaintext dimension $l$, it outputs public parameters $pp$ and a secret key $sk$.
- $\text{Encrypt}(sk, \mathbf{v})$: On input the secret key $sk$ and a vector $\mathbf{v} \in \mathbb{F}^l$, it outputs a ciphertext $\mathbf{C}$.
- $\text{Decrypt}(sk, \mathbf{C})$: On input the secret key $sk$ and a ciphertext $\mathbf{C}$, it outputs a vector $\mathbf{v} \in \mathbb{F}^l$ or a special symbol $\perp$.
- $\text{Add}(pp, \{\mathbf{C}_i\}_{i\in[m]}, \{a_i\}_{i\in[m]})$: On input $pp$, a collection of ciphertexts $\{\mathbf{C}_i\}_{i\in[m]}$, and scalars $\{a_i\}_{i\in[m]} \in \mathbb{F}$, it outputs a new ciphertext $\mathbf{C}^*$.

The scheme must satisfy correctness (via additive homomorphism), CPA security, circuit privacy, and a linear-only property.

Property 1 (Additive Homomorphism). *For all $\lambda \in \mathbb{N}$, all $\{\mathbf{v}_i\}_{i\in[m]} \subseteq \mathbb{F}^l$, and all $\{a_i\}_{i\in[m]} \subseteq \mathbb{F}$ where $m = poly(\lambda)$, the following holds:*

$$\Pr\left[Decrypt(sk, \mathbf{C}^*) = \sum_{i\in[m]} a_i \mathbf{v}_i\right] = 1 - negl(\lambda)$$

*where $(pp, sk) \leftarrow Setup(1^\lambda, 1^l)$, $\mathbf{C}_i \leftarrow Encrypt(sk, \mathbf{v}_i)$, and $\mathbf{C}^* \leftarrow Add(pp, \{\mathbf{C}_i\}, \{a_i\})$.*

To achieve zero-knowledge, the scheme must also hide the coefficients $\{a_i\}$ used in the homomorphic computation.

*Definition A.3 (Circuit Privacy (Statistical vs. Computational) [29]).* A vector encryption scheme $\Pi_{\text{VE}}$ has circuit privacy if for any stateful PPT adversary $\mathcal{A}$, there is an efficient simulator $\mathcal{S}$ such that $\mathcal{A}$'s advantage in the distinguishing game described below is negligible. Let $\text{Game}^{\text{circ-priv}}_{\Pi_{\text{VE}},\mathcal{A},\mathcal{S}}(1^\lambda)$ denote the output of the game. The game proceeds as follows:

(1) The challenger runs $(pp, sk) \leftarrow \text{Setup}(1^\lambda, 1^l)$ and gives $(pp, sk)$ to $\mathcal{A}$. The adversary replies with a collection of vectors $\{\boldsymbol{\mu}_i\}_{i\in[m]}$.
(2) The challenger constructs $\mathbf{C}_i \leftarrow \text{Encrypt}(sk, \boldsymbol{\mu}_i)$ for all $i \in [m]$ and gives $\{\mathbf{C}_i\}_{i\in[m]}$ to $\mathcal{A}$. The adversary replies with a collection of scalars $\{a_i\}_{i\in[m]}$.
(3) The challenger computes $\boldsymbol{\mu}_{\text{sum}} = \sum_{i\in[m]} a_i \boldsymbol{\mu}_i$. It then computes a real ciphertext $\mathbf{C}^*_0 \leftarrow \text{Add}(pp, \{\mathbf{C}_i\}, \{a_i\})$ and a simulated ciphertext $\mathbf{C}^*_1 \leftarrow \mathcal{S}(1^\lambda, pp, sk, \boldsymbol{\mu}_{\text{sum}})$. It samples a bit $b \leftarrow \{0, 1\}$ and sends $\mathbf{C}^*_b$ to $\mathcal{A}$.
(4) The adversary outputs a bit $b'$. The game's output is 1 if $b' = b$, and 0 otherwise.

If the advantage $\left|\Pr[\text{Game}^{\text{circ-priv}}_{\Pi_{\text{VE}},\mathcal{A},\mathcal{S}}(1^\lambda) = 1] - \frac{1}{2}\right|$ is negligible, the scheme has **computational circuit privacy**. If the output distributions of Add and $\mathcal{S}$ are statistically close, it has **statistical circuit privacy**.

The 'linear-only' property is formalized as *Strictly Linear Targeted Malleability*, which asserts that an adversary cannot perform non-linear operations.

*Definition A.4 (Strictly Linear Targeted Malleability (SLTM) [7, 29]).* A vector encryption scheme $\Pi_{\text{VE}}$ satisfies SLTM if for any PPT adversary $\mathcal{A}$ and plaintext generator $\mathcal{M}$, there exists a (possibly unbounded) simulator $\mathcal{S}$ such that no PPT distinguisher $\mathcal{D}$ can win the game $\text{Expt}^{\text{SLTM}}_{\Pi_{\text{VE}},\mathcal{A},\mathcal{M},\mathcal{S},\mathcal{D},z}(1^\lambda)$ with non-negligible advantage. The game proceeds as follows: a bit $b$ is flipped. If $b = 0$ (REAL), ciphertexts are generated, manipulated by $\mathcal{A}$, and the result is decrypted. If $b = 1$ (IDEAL), the simulator $\mathcal{S}$ produces coefficients $\boldsymbol{\pi}$ and the result is computed as a linear combination of plaintexts. The distinguisher $\mathcal{D}$ must guess $b$.

# B Proof of Lemma 4.1

Proof. The following is the full reduction proof of Lemma 4.1. We proceed in game hops, starting from $\text{Game}_0$, which is identical to the game in the Definition of the Coset Error Knapsack Hint MLWE Problem (Def. 3.1), and ending in $\text{Game}_3$, which is identical to the reduction experiment above for $\mathcal{B}$ breaking the standard MH-MLWE problem. In each game $i$, we let $p_i := \Pr_{\text{Game}_i}[b' = b]$ and let $\text{Adv}_i := |p_i - 1/2|$ denote the advantage. We upper bound the difference $|\text{Adv}_i - \text{Adv}_{i-1}|$ to upper bound the advantage loss in each game hop. Note that $|\text{Adv}_i - \text{Adv}_{i-1}| \leq |p_i - p_{i-1}|$.

$\text{Game}_0$: (The original game) This is the original game of Coset Error Knapsack Matrix Hint MLWE Problem (Def. 3.1), CEK-MH-MLWE. $\text{Game}_1$: (Switch $\mathbf{b}'$ in Random case) We change the distribution of $\mathbf{b}'$ in the Random case $b = 1$ as follows:

- If $b = 1$ (Random), generate $\mathbf{b}' = \mathbf{A}' \cdot (\mathbf{B}_\Lambda \cdot \mathbf{u}_1) \bmod q$, where $\mathbf{u}_1 \leftarrow \mathcal{U}(R_q^l)$ (instead of $\mathbf{b}' \leftarrow \mathcal{U}(R_q^l)$ in the previous game).

We first observe that $B_\Lambda \cdot \mathcal{U}(R_q^l) = \mathcal{U}(I_{q,\Lambda})$, where $I_{q,\Lambda} := \Lambda \bmod q \subseteq R_q^l$ is the $R_q$-module $R_q^l$ corresponding to the $q$-ary lattice $\Lambda$. Hence, we have $|p_1 - p_0| \leq \Delta_1$, where $\Delta_1 := \Delta(D_0, D_1)$ is the statistical

distance between the distributions $D_0$ and $D_1$, where:

$$D_0 : (\mathbf{A}' \leftarrow U(R_q^{n\times l}), \mathbf{b}' \leftarrow \mathcal{U}(R_q^l)) \tag{29}$$

$$D_1 : (\mathbf{A}' \leftarrow U(R_q^{n\times l}), \mathbf{b}' = \mathbf{A}' \cdot \mathcal{U}(I_{q,\Lambda}) \bmod q). \tag{30}$$

We can upper bound $\Delta_1$ by applying the Leftover Hash Lemma over cyclotomic rings via counting ideals approach, namely Theorem 4 in [8], to get the bound

$$\Delta_1 \leq \frac{1}{2}\sqrt{\sum_{S\subset[r]} N(I_S)^n \cdot p(S, I_{q,\Lambda})}, \tag{31}$$

where

$$p(S, I_{q,\Lambda}) := \Pr_{x,x'\leftarrow\mathcal{U}(I_{\Lambda,q})}[\langle x_1 - x'_1, \ldots, x_m - x'_m\rangle = I_S], \tag{32}$$

where we recall that the ideals of $R_q$ are in a one-to-one correspondence with subsets of $[r]$, with ideal $I_S := \langle\prod_{i\in S} f_i\rangle$ corresponding to subset $S \subseteq [r]$, and have ideal norm $N(I_S) = q^{\frac{d}{r}|S|}$. Since $I_{\Lambda,q}$ is a finite additive group, we have that $x - x'$ is uniform in $I_{\Lambda,q}$ when $x, x'$ are uniform and independent in $I_{\Lambda,q}$. It follows that $p(S, I_{q,\Lambda}) = \Pr_{x\leftarrow\mathcal{U}(I_{\Lambda,q})}[\langle x_1, \ldots, x_m\rangle = I_S] = \Pr_{x\leftarrow\mathcal{U}(I_{\Lambda,q})}[x \in I_S^l]$. Since $B_\Lambda$ has full rank $l - v_\Lambda$ modulo each field $\mathbf{F}_i$ for $i \in [r]$, we have that the CRT slot vector $(x \bmod f_1, \ldots, x \bmod f_r)$ is uniform on $I_{\Lambda,q,1} \times \cdots \times I_{\Lambda,q,r}$ when $x$ is uniform on $I_{\Lambda,q}$, where $I_{\Lambda,q,i} := I_{\Lambda,q} \bmod f_i$ for $i \in [r]$ is a subspace of $\mathbb{F}_i$ of rank $l - v_\Lambda$. It follows that

$$p(S, I_{q,\Lambda}) = \prod_{i\in S} \Pr_{x\leftarrow\mathcal{U}(I_{\Lambda,q,i})}[x \bmod f_j = 0^l] \tag{33}$$

$$= \prod_{i\in S} \frac{1}{|\mathbb{F}_i|^{l-v_\Lambda}} = q^{-\frac{d}{r}(l-v_\Lambda)|S|}, \tag{34}$$

for all $|S|\subset [r]$. We get that $|\mathrm{Adv}_1 - \mathrm{Adv}_0|$ is upper bounded by

$$|p_1 - p_0| \leq \Delta_1 \tag{35}$$

$$\leq \frac{1}{2}\sqrt{\sum_{S\subset[r]} q^{n\frac{d}{r}|S|} \cdot q^{-\frac{d}{r}(l-v_\Lambda)|S|}}) \tag{36}$$

$$\leq \frac{1}{2}\sqrt{\sum_{S\subset[r]} 2^r \cdot q^{-\frac{d}{r}(l-(v_\Lambda+n))}}), \tag{37}$$

which is upper bounded by $\varepsilon/4$ by the uniformity condition (11).

$\mathrm{Game}_2$: (Restrict to full rank GoodA) We change the experiment in Phase 1 as follows:

- **Phase 1:** $\mathcal{B}$ first runs $\mathrm{Gen}\Lambda(\lambda)$ to get $(\mathbf{H}, \mathbf{B}_\Lambda)$, computes $\hat{\mathbf{A}} := \mathbf{B}_\Lambda\mathbf{A} \bmod q$ and tests if $\hat{\mathbf{A}} \notin \mathrm{GoodA}$, and in that case, $\mathcal{B}$ outputs a random bit $b' \leftarrow \mathcal{U}(\{0, 1\})$. Here, GoodA denotes the set of matrices $\hat{\mathbf{A}} \in R_q^{l\times(l-n)}$ such that for all $i \in [r]$, $\hat{\mathbf{A}}_i := \hat{\mathbf{A}} \bmod f_i \in \mathbb{F}_i^{l\times(l-n)}$ has full rank $l - n$ over the field $\mathbb{F}_i$. Note that $\mathbf{A}'$ is still sampled as $\mathbf{A}' \leftarrow \mathcal{U}(R_q^{n\times l})$ as in the previous game.

Since the distribution of $\mathbf{A}'$ is the same as in the previous game, we have

$$p_2 := \Pr_{\mathrm{Game}_2}[b' = b] \tag{38}$$

$$= \Pr_{\mathrm{Game}_2}[b' = b|\mathrm{GoodA}] \cdot \Pr_{\mathrm{Game}_2}[\mathrm{GoodA}] \tag{39}$$

$$+ \Pr_{\mathrm{Game}_2}[b' = b|\overline{\mathrm{GoodA}}] \cdot \Pr_{\mathrm{Game}_2}[\overline{\mathrm{GoodA}}] \tag{40}$$

$$= \Pr_{\mathrm{Game}_1}[b' = b] \cdot \Pr_{\mathrm{Game}_2}[\mathrm{GoodA}] \tag{41}$$

$$+ \Pr_{\mathrm{Game}_2}[b' = b|\overline{\mathrm{GoodA}}] \cdot (1 - \Pr_{\mathrm{Game}_2}[\mathrm{GoodA}]) \tag{42}$$

$$= \Pr_{\mathrm{Game}_2}[\mathrm{GoodA}] \cdot (p_1 - 1/2) + 1/2. \tag{43}$$

We show below that assuming uniformity condition (10) holds,

$$\Pr_{\mathrm{Game}_2}[\mathrm{GoodA}] \geq 1 - \varepsilon, \tag{44}$$

and therefore:

$$\mathrm{Adv}_2 = |p_2 - 1/2| = \Pr_{\mathrm{Game}_2}[\mathrm{GoodA}] \cdot \mathrm{Adv}_1 \geq (1 - \varepsilon) \cdot \mathrm{Adv}_1. \tag{45}$$

To lower bound $\Pr_{\mathrm{Game}_2}[\mathrm{GoodA}]$, we apply the following well-known result (see, e.g. Lemma 1 in [28]) on the probability a uniformly random rectangular matrix over a finite field is full-rank.

Lemma B.1. *Let $\mathbb{F}$ denote a finite field and let $\mathbf{C} \leftarrow \mathcal{U}(\mathbb{F}^{\rho\times c})$ with $\rho \geq c$. Then*

$$\Pr[\mathrm{rank}_{\mathbb{F}}(\mathbf{C}) = c] = \prod_{i=\rho-(c-1)}^{\rho} \left(1 - \frac{1}{|\mathbb{F}|^i}\right). \tag{46}$$

We in fact use the following generalization of this Lemma, where the columns of $\mathbf{B}$ are uniform in a subspace.

Corollary B.2. *Let $\mathbb{F}$ denote a finite field and let $\mathbf{C} \in \mathbb{F}^{\rho\times c}$ be sampled with its columns independently uniform in a subspace $I \subseteq \mathbb{F}^\rho$ of rank $\rho - \delta$ over $\mathbb{F}$ with $\rho - \delta \geq c$ and $\delta \geq 0$. Then*

$$\Pr[\mathrm{rank}_{\mathbb{F}}(\mathbf{C}) = c] = \prod_{i=(\rho-\delta)-(c-1)}^{\rho-\delta} \left(1 - \frac{1}{|\mathbb{F}|^i}\right). \tag{47}$$

Proof. Since $I$ is a subspace of rank $\rho - \delta$ in $\mathbb{F}^\rho$, it has a basis matrix $\mathbf{B}_I \in \mathbb{F}^{\rho\times(\rho-\delta)}$. Therefore, the matrix $\mathbf{C} = \mathbf{B}_I \cdot \mathbf{V} \in \mathbf{F}^{\rho\times c}$ has columns independently and uniformly distributed in $I$ when $\mathbf{V}$ is sampled uniformly at random from $\mathbb{F}^{(\rho-\delta)\times c}$. It follows that the desired probability is $p := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})}[\mathrm{rank}_{\mathbb{F}}(\mathbf{B}_I \cdot \mathbf{V}) = c]$. Note that $\mathrm{rank}_{\mathbb{F}}(\mathbf{B}_I \cdot \mathbf{V}) = \mathrm{rank}_{\mathbb{F}}(\mathbf{V}) - \mathrm{rank}_{\mathbb{F}}(\mathrm{RightKer}(\mathbf{B}_I) \cap \mathrm{Im}(\mathbf{V})) = \mathrm{rank}_{\mathbb{F}}(\mathbf{V})$ since $\mathbf{B}_I$ has full rank and therefore a trivial right-kernel. It follows that $p := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})}[\mathrm{rank}_{\mathbb{F}}(\mathbf{V}) = c]$, and the result follows from Lemma B.1. □

We apply Corollary B.2 to compute the probability $\Pr[\hat{\mathbf{A}} \in \mathrm{GoodA}]$. We first recall that the columns of $\hat{\mathbf{A}} = \mathbf{B}_\Lambda\mathbf{A} \bmod q$ are uniform in $I_{q,\Lambda}$, where $I_{q,\Lambda} := \Lambda \bmod q \subseteq R_q^l$ is the $R_q$-module $R_q^l$ corresponding to the $q$-ary lattice $\Lambda$. Therefore, by the CRT bijection, for $i \in [r]$, the $i$'th CRT components of $\hat{\mathbf{A}}$, namely $\hat{\mathbf{A}}_i := \hat{\mathbf{A}} \bmod f_i \in \mathbb{F}_i^{l\times(l-n)}$ are independent and uniform in $I_{q,\Lambda,i} := I_{q,\Lambda} \bmod f_i \subseteq \mathbf{F}_i^l$ denote the subspace of the field $\mathbf{F}_i$ corresponding $I_{q,\Lambda}$. Since $I_{q,\Lambda,i} = \{\mathbf{v} \in \mathbb{F}_i^l : \mathbf{H} \cdot \mathbf{v} = \mathbf{0} \bmod q\}$, the assumption $\mathrm{rank}_{\mathbb{F}_i}(\mathbf{H}) = v_\Lambda$ implies

that $I_{q,\Lambda,i}$ is a subspace of $\mathbb{F}_i$ of rank $l - v_\Lambda$. By Corollary B.2 with $\rho = l$, $\delta = v_\Lambda$, since $n \geq v_\Lambda$ and $c = l - n$, $|\mathbb{F}_i| = q^{d/r}$, it follows that

$$\Pr[\hat{\mathbf{A}} \in \text{GoodA}] = \prod_{i\in[r]} \Pr_{\hat{\mathbf{A}}_i \leftarrow \mathcal{U}(I^{l-n}_{q,\Lambda,i})} [\text{rank}_{\mathbb{F}_i}(\hat{\mathbf{A}}_i) = l - n] \tag{48}$$

$$= \prod_{j=n-v_\Lambda+1}^{l-v_\Lambda} \left(1 - q^{-\frac{d}{r}\cdot j}\right)^r \tag{49}$$

$$\geq \prod_{j=n-v_\Lambda+1}^{l-v_\Lambda} \exp\left(-2rq^{-\frac{d}{r}\cdot j}\right) \tag{50}$$

$$= \exp\left(-2r \sum_{j=n-v_\Lambda+1}^{l-v_\Lambda} q^{-\frac{d}{r}\cdot j}\right) \tag{51}$$

$$= \exp\left(-2r\psi q^{-\frac{d}{r}\cdot(n-v_\Lambda+1)}\right) \tag{52}$$

$$\geq 1 - \varepsilon_G, \text{ where } \varepsilon_G := 2r\gamma q^{-\frac{d}{r}\cdot(n-v_\Lambda+1)}, \tag{53}$$

where $\psi$ is given by

$$\psi := \frac{1 - q^{-(d/r)\cdot(l-n))}}{1 - q^{-d/r}} \leq \frac{1}{1 - q^{-\frac{d}{r}}} := \gamma. \tag{54}$$

Note that $\gamma \leq \frac{1}{1-q^{-1}} \leq 2$ since $r \leq d$ and $q \geq 2$). Above, we used the inequalities $1 - x \geq \exp(-2x) \geq 1 - 2x$ for $x \in [0, 1/2]$ and the uniformity condition (10).

Game$_3$: (Primal $\mathbf{A}$ to Dual $\mathbf{A}'$ - Uniformity of $\mathbf{A}'$) We change the experiment in Phase 1 to sample $\mathbf{A}'$ differently as follows:

- If $\hat{\mathbf{A}} \in$ GoodA, $\mathcal{B}$ samples each of the $n$ rows of a matrix $\mathbf{A}' \in R_q^{n\times l}$ independently from the distribution $\chi_{A'} := \mathcal{U}(I_q^\perp(\hat{\mathbf{A}}))$ on $R_q^l$, where $I_q^\perp(\hat{\mathbf{A}}) := \{\bar{\mathbf{a}} \in R_q^l : \bar{\mathbf{a}}^T\hat{\mathbf{A}} = \mathbf{0} \bmod q\}$. This distribution $\chi_{A'}$ is sampleable in PPT by linear algebra in each of the fields $\mathbb{F}_i$ for $i \in [r]$ and CRT.

Note that $\mathbf{A}'$ in this game satisfies

$$\mathbf{A}' \cdot \hat{\mathbf{A}} = \mathbf{0} \mod q. \tag{55}$$

Therefore, we have

$$|p_3 - p_2| \leq \Delta_3, \tag{56}$$

where

$$\Delta_3 := \Delta\left(\mathcal{U}(R_q^l)^n, \chi_{A'}^n\right) \leq n \cdot \Delta\left(\mathcal{U}(R_q^l), \chi_{A'}\right) \tag{57}$$

is the statistical distance between the distributions $\mathbf{A}' \in R_q^{n\times l}$ in the Games 2 and 3, where $\chi_A$ is taken over the uniformly random choice of $\mathbf{A} \in$ GoodA. We show that the following bound holds

$$\Delta_3 \leq 4nrq^{-\frac{d}{r}\cdot(n-v_\Lambda)}. \tag{58}$$

To establish the bound, we first study the distribution $\chi_{A'}$. Fix any $\bar{\mathbf{a}}' \in R_q^l$. Then we have

$$\chi_{A'}(\bar{\mathbf{a}}') = \sum_{\bar{\hat{\mathbf{A}}}\in\text{GoodA}} \Pr[\hat{\mathbf{A}} = \bar{\hat{\mathbf{A}}}] \cdot \chi_{A'}(\bar{\mathbf{a}}'|\bar{\hat{\mathbf{A}}}), \tag{59}$$

where $\chi_{A'}(\cdot|\bar{\hat{\mathbf{A}}})$ denotes the distribution $\chi_{A'}$ conditioned on $\mathbf{A} = \bar{\hat{\mathbf{A}}}$, namely the uniform distribution on $I_q^\perp(\bar{\hat{\mathbf{A}}})$. By definition of GoodA, for all $\bar{\hat{\mathbf{A}}} \in$ GoodA and $i \in [r]$, $\text{rank}_{\mathbb{F}_i}(\bar{\hat{\mathbf{A}}}_i) = l - n$ and therefore $|I_{q,i}^\perp(\bar{\hat{\mathbf{A}}}_i)| = |\mathbb{F}_i|^n$, where $I_{q,i}^\perp(\bar{\hat{\mathbf{A}}}_i) := \{\bar{\mathbf{a}}_i \in \mathbb{F}_i^l : \bar{\mathbf{a}}_i^T\bar{\hat{\mathbf{A}}}_i = \mathbf{0}\}$. It follows that $|I_q^\perp(\bar{\hat{\mathbf{A}}})| = \prod_{i\in[r]}|I_{q,i}^\perp(\bar{\hat{\mathbf{A}}}_i)| = |R_q^n|$ and hence

$$\chi_{A'}(\bar{\mathbf{a}}'|\bar{\hat{\mathbf{A}}}) = \begin{cases} \frac{1}{|R_q^n|} \text{ if } & \bar{\mathbf{a}}' \in \mathbf{I}_{\mathbf{q}}^\perp(\bar{\hat{\mathbf{A}}}) \\ 0, & \text{else.} \end{cases} \tag{60}$$

Furthermore, since $\hat{\mathbf{A}}$ is uniform on GoodA conditioned on passing the test $\hat{\mathbf{A}} \in$ GoodA, we get from (60) and (59) that

$$\chi_{A'}(\bar{\mathbf{a}}') = \frac{1}{|R_q^n|} p_{\text{Good}}(\bar{\mathbf{a}}') \tag{61}$$

where

$$p_{\text{Good}}(\bar{\mathbf{a}}') := \Pr_{\bar{\hat{\mathbf{A}}}\leftarrow\mathcal{U}(\text{GoodA})} [(\bar{\mathbf{a}}')^T\bar{\hat{\mathbf{A}}} = \mathbf{0}] = \frac{p(\bar{\mathbf{a}}')}{p_{\text{GoodA}}}, \tag{62}$$

and

$$p_{\text{GoodA}} := \Pr[\text{GoodA}] = \Pr_{\bar{\hat{\mathbf{A}}}\leftarrow\mathcal{U}(I_{q,\Lambda})} [\bar{\hat{\mathbf{A}}} \in \text{GoodA}]$$

has been bounded in the previous game in (44), and

$$p(\bar{\mathbf{a}}') := \Pr_{\bar{\hat{\mathbf{A}}}\leftarrow\mathcal{U}(I_{q,\Lambda})^{l-n}} [(\bar{\hat{\mathbf{A}}} \in \text{GoodA}) \wedge ((\bar{\mathbf{a}}')^T\bar{\hat{\mathbf{A}}} = \mathbf{0})]. \tag{63}$$

By the CRT bijection, we have

$$p(\bar{\mathbf{a}}') = \prod_{i\in[r]} p_i(\bar{\mathbf{a}}'_i), \tag{64}$$

where

$$p_i(\bar{\mathbf{a}}'_i) := \Pr_{\bar{\hat{\mathbf{A}}}_i\leftarrow\mathcal{U}(I_{q,\Lambda,i})^{l-n}} [(\text{rank}_{\mathbb{F}_i}(\bar{\hat{\mathbf{A}}}_i) = l - n) \wedge ((\bar{\mathbf{a}}'_i)^T\bar{\hat{\mathbf{A}}}_i = \mathbf{0})], \tag{65}$$

with $\bar{\mathbf{a}}'_i := \bar{\mathbf{a}}' \bmod f_i$, $\bar{\hat{\mathbf{A}}}_i := \bar{\hat{\mathbf{A}}} \bmod f_i$ for $i$ and we recall that $I_{q,\Lambda,i} = \{\mathbf{v} \in \mathbb{F}_i^l : \mathbf{H}\cdot\mathbf{v} = \mathbf{0} \bmod q\}$ is a subspace of $\mathbb{F}_i$ of rank $l - v_\Lambda$. To compute $p_i(\bar{\mathbf{a}}'_i)$ we use another extension of Corollary B.2 as follows.

Corollary B.3. *Let $\mathbb{F}$ denote a finite field and let $\mathbf{C} \in \mathbb{F}^{\rho\times c}$ be sampled with its columns independently uniform in a subspace $I \subseteq \mathbb{F}^\rho$ of rank $\rho - \delta$ over $\mathbb{F}$ with $\rho - \delta \geq c + 1$ and $\delta \geq 0$. Let $I^\perp := \{\mathbf{a} \in \mathbb{F}^\rho : \mathbf{a}^T\mathbf{v} = \mathbf{0}$ for all $\mathbf{v} \in I\}$ denote the orthogonal complement of $I$ and let $\bar{\mathbf{a}} \in \mathbb{F}^\rho \setminus I^\perp$. Then*

$$\Pr[(\text{rank}_{\mathbb{F}}(\mathbf{C}) = c) \wedge (\bar{\mathbf{a}}^T\mathbf{C} = \mathbf{0})] = \frac{1}{|\mathbb{F}|^c} \cdot \prod_{i=\rho-(\delta+c)}^{\rho-\delta-1} \left(1 - \frac{1}{|\mathbb{F}|^i}\right). \tag{66}$$

Proof. Arguing similarly to the first part of the proof of Corollary B.2, we have that the desired probability is

$$p := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})} [(\text{rank}_{\mathbb{F}}(\mathbf{V}) = c) \wedge (\bar{\mathbf{b}}^T\cdot\mathbf{V}) = 0)], \tag{67}$$

where $\bar{\mathbf{b}}^T := \bar{\mathbf{a}}^T\mathbf{B}_I$ for $\mathbf{B}_I \in \mathbb{F}^{\rho\times(\rho-\delta)}$ being a basis for subspace $I$ over $\mathbb{F}$. Let $I_{\bar{\mathbf{b}}}^\perp := \{\mathbf{v} \in \mathbb{F}^{\rho-\delta} : \bar{\mathbf{b}}^T\cdot\mathbf{v} = \mathbf{0}\}$. Since $\bar{\mathbf{a}} \notin I^\perp$, we have $\bar{\mathbf{b}} \neq 0$ and hence $I_{\bar{\mathbf{b}}}^\perp$ is a subspace of $\mathbb{F}^{\rho-\delta}$ of rank $\rho - \delta - 1$. Furthermore, we have:

$$p := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})} [(\text{rank}_{\mathbb{F}}(\mathbf{V}) = c) \wedge (\mathbf{V} \in (I_{\bar{\mathbf{b}}}^\perp)^c)] \tag{68}$$

$$= p' \cdot p'', \tag{69}$$

where

$$p' := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})}[(\mathrm{rank}_{\mathbb{F}}(\mathbf{V}) = c) \mid (\mathbf{V} \in (I_{\mathfrak{b}}^{\perp})^c)] \tag{70}$$

$$= \Pr_{\mathbf{V}\leftarrow\mathcal{U}((I_{\mathfrak{b}}^{\perp})^c)}[\mathrm{rank}_{\mathbb{F}}(\mathbf{V}) = c] \tag{71}$$

$$= \prod_{i=\rho-(\delta+c)}^{\rho-\delta-1}\left(1 - \frac{1}{|\mathbb{F}|^i}\right), \tag{72}$$

where the last equality follows by applying Corollary B.2 using that $I_{\mathfrak{b}}^{\perp}$ is a subspace of $\mathbb{F}^{\rho-\delta}$ of rank $\rho-\delta-1$ and the assumption $\rho-\delta \geq c+1$ and

$$p'' := \Pr_{\mathbf{V}\leftarrow\mathcal{U}(R_q^{(\rho-\delta)\times c})}[\mathbf{V} \in (I_{\mathfrak{b}}^{\perp})^c] = \frac{1}{|\mathbb{F}|^c}, \tag{73}$$

where the last equality follows using the fact that $|I_{\mathfrak{b}}^{\perp}| = |\mathbb{F}|^{\rho-\delta-1}$ so each of the $c$ columns of $\mathbf{V}$ falls in $I_{\mathfrak{b}}^{\perp}$ with probability $\frac{1}{|\mathbb{F}|}$. The result now follows by plugging (70) and (73) into (68). □

To compute $p_i(\bar{\mathbf{a}}_i')$ in (65), we apply Corollary B.3 with parameters $\mathbb{F} = \mathbb{F}_i$, $I = I_{q,\Lambda,i}$, $\rho = l$, $c = l-n$, $\delta = v_\Lambda$, $\bar{\mathbf{a}} = \bar{\mathbf{a}}_i'$ to conclude using $n \geq v_\Lambda + 1$ (to satisfy $\rho - c \geq c+1$) that

$$p_i(\bar{\mathbf{a}}_i') = \frac{1}{|\mathbb{F}_i|^{l-n}} \cdot \prod_{j=n-v_\Lambda}^{l-v_\Lambda-1}\left(1 - \frac{1}{|\mathbb{F}_i|^j}\right) \text{ for } \bar{\mathbf{a}}_i' \notin I_{q,\Lambda,i}^{\perp}. \tag{74}$$

Now, plugging (74) into (64) and then into (62) and (59), with $|\mathbb{F}_i| = q^{\frac{d}{r}}$ for $i \in [r]$, we get

$$\chi_{A'}(\bar{\mathbf{a}}') = q^{-dl} \cdot \frac{p'}{p_{\mathrm{GoodA}}} \text{ for } \bar{\mathbf{a}}_i' \notin \mathrm{Bada}', \tag{75}$$

where

$$p' := \prod_{j=n-v_\Lambda}^{l-v_\Lambda-1}\left(1 - q^{-\frac{d}{r}\cdot j}\right)^r, \tag{76}$$

and we define the set of 'bad' $\bar{\mathbf{a}}'$ in $R_q^l$ as

$$\mathrm{Bada}' := \left\{\bar{\mathbf{a}}' \in R_q^l : \bigvee_{i\in[r]}(\bar{\mathbf{a}}_i' \in I_{q,\Lambda,i}^{\perp})\right\}. \tag{77}$$

Towards obtaining the upper bound (58), we now show that

$$\frac{p'}{p_{\mathrm{GoodA}}} = 1 - \varepsilon', \text{ where } \varepsilon' \leq 2rq^{-\frac{d}{r}(n-v_\Lambda)}. \tag{78}$$

Indeed, comparing (49) to (76) and using $p_{\mathrm{GoodA}} \leq 1$ we have:

$$\frac{p'}{p_{\mathrm{GoodA}}} = \frac{\left(1 - q^{-\frac{d}{r}\cdot(n-v_\Lambda)}\right)^r}{\left(1 - q^{-\frac{d}{r}\cdot(l-v_\Lambda)}\right)^r} \tag{79}$$

$$\geq \exp\left(-2rq^{-\frac{d}{r}\cdot(n-v_\Lambda)}\right) \tag{80}$$

$$\geq 1 - 2rq^{-\frac{d}{r}\cdot(n-v_\Lambda)}, \tag{81}$$

which established the claimed bound (78) on $\varepsilon'$, where we have used the inequalities $1 - x \geq \exp(-2x) \geq 1 - 2x$ for $x \in [0, 1/2]$. Plugging (79) into (75) we get

$$\chi_{A'}(\bar{\mathbf{a}}') = q^{-dl} \cdot (1 - \varepsilon') \text{ for } \bar{\mathbf{a}}_i' \notin \mathrm{Bada}'. \tag{82}$$

Therefore,

$$\Delta\left(\mathcal{U}(R_q^l), \chi_{A'}\right) = \frac{1}{2}\sum_{\bar{\mathbf{a}}'\in R_q^l}\left|q^{-dl} - \chi_{A'}(\bar{\mathbf{a}}')\right| \tag{83}$$

$$= \frac{1}{2}(S_1 + S_2), \tag{84}$$

where, using (82), we have

$$S_1 := \sum_{\bar{\mathbf{a}}'\in R_q^l\setminus\mathrm{Bada}'}\left|q^{-dl} - \chi_{A'}(\bar{\mathbf{a}}')\right| = \varepsilon', \tag{85}$$

and

$$S_2 := \sum_{\bar{\mathbf{a}}'\in\mathrm{Bada}'}\left|q^{-dl} - \chi_{A'}(\bar{\mathbf{a}}')\right| \tag{86}$$

$$\leq \sum_{\bar{\mathbf{a}}'\in\mathrm{Bada}'} q^{-dl} + \chi_{A'}(\bar{\mathbf{a}}') \tag{87}$$

$$\leq |\mathrm{Bada}'|\cdot q^{-dl} + (1 - \sum_{\bar{\mathbf{a}}'\in R_q^l\setminus\mathrm{Bada}'}\chi_{A'}(\bar{\mathbf{a}}')). \tag{88}$$

But since

$$\sum_{\bar{\mathbf{a}}'\in R_q^l\setminus\mathrm{Bada}'}\chi_{A'}(\bar{\mathbf{a}}') = \sum_{\bar{\mathbf{a}}'\in R_q^l\setminus\mathrm{Bada}'} q^{-dl} + \sum_{\bar{\mathbf{a}}'\in R_q^l\setminus\mathrm{Bada}'}(q^{-dl} - \chi_{A'}(\bar{\mathbf{a}}')) \tag{89}$$

$$= (1 - |\mathrm{Bada}'|\cdot q^{-dl}) + S_1, \tag{90}$$

we get

$$S_2 \leq 2|\mathrm{Bada}'|\cdot q^{-dl} + S_1. \tag{91}$$

Plugging (85) and (91) into (83) gives

$$\Delta\left(\mathcal{U}(R_q^l), \chi_{A'}\right) \leq \varepsilon' + |\mathrm{Bada}'|\cdot q^{-dl} \leq 4rq^{-\frac{d}{r}(n-v_\Lambda)}, \tag{92}$$

where we have used the upper bound (78) on $\varepsilon'$ and the following union bound on $|\mathrm{Bada}'|$:

$$|\mathrm{Bada}'| \leq \sum_{i\in[r]}\left|I_{q,\Lambda,i}^{\perp}\right|\cdot\prod_{j\in[r]\setminus i}|\mathbb{F}_j|^l \tag{93}$$

$$= \sum_{i\in[r]} q^{\frac{d}{r}v_\Lambda}\cdot\prod_{j\in[r]\setminus i} q^{\frac{d}{r}\cdot l} \tag{94}$$

$$= rq^{\frac{d}{r}\cdot((r-1)\cdot l + v_\Lambda)}, \tag{95}$$

which gives the bound

$$|\mathrm{Bada}'|\cdot q^{-dl} \leq rq^{-\frac{d}{r}(l-v_\Lambda)} \leq rq^{-\frac{d}{r}(n-v_\Lambda)}, \tag{96}$$

using $l > n$ and completes the proof of (92). Finally, plugging (92) into (57) completes the proof of the claimed bound (58) on $\Delta_3$.

Game$_4$*: (Real Standard MH-MLWE instance into Real CEK-MH-MLWE instance).* This game bridges the Real distribution of the standard MH-MLWE problem (Input) to the Real distribution of the CEK-MH-MLWE problem (Output). We construct the reduction algorithm $\mathcal{B}$ to perform the necessary Gaussian convolution, support shifting, and basis transformation steps. Recall that $\mathcal{B}$ takes as input a standard MH-MLWE tuple $(\mathbf{A}, \mathbf{b}, \mathbf{M}, \mathbf{z})$, where in the Real case $\mathbf{b} = \mathbf{As} + \mathbf{e}$ with $\mathbf{e} \leftarrow \mathcal{D}_{R^l,\sigma_e,0}$, and $\mathbf{z} = \mathbf{Me} + \mathbf{y}$ with $\mathbf{y} \leftarrow \chi_y$.

The algorithm $\mathcal{B}$ proceeds as follows:

(1) **Setup Shifts:** Using the coset shift $\mathbf{c}$ output by $\mathcal{A}_1$ in Phase 1, $\mathcal{B}$ computes the integer pre-image $\mathbf{c}_0 := \mathbf{B}_\Lambda^{-1}\mathbf{c}$. Note that since $\Lambda = \mathbf{B}_\Lambda R^l$, such a $\mathbf{c}_0$ exists in $R^l$.
(2) **Sample Noise:** $\mathcal{B}$ samples the following noise vectors:

- Center-shift noise: $\mathbf{d}_1 \leftarrow \mathcal{D}_{R^l,\sigma_e,-\mathbf{c}_0}$.
- Flooding noise: $\mathbf{e}_{flood} \leftarrow \mathcal{D}_{\Lambda,\sqrt{\Sigma_{flood}},0}$, where $\Sigma_{flood} := (\sigma_e')^2\mathbf{I} - 2\sigma_e^2\mathbf{B}_\Lambda\mathbf{B}_\Lambda^T$. This sampling is possible provided $\Sigma_{flood}$ is positive definite, which is guaranteed by the sampleability condition of (9).

(3) **Compute Target Sample $\mathbf{b}'$:** $\mathcal{B}$ computes the transformed sample vector:

$$\mathbf{b}' := \mathbf{A}'\mathbf{B}_\Lambda(\mathbf{b} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{A}'\mathbf{e}_{flood} \bmod q.$$

(4) **Compute Target Hint $\mathbf{z}'$:** $\mathcal{B}$ computes the transformed hint vector:

$$\mathbf{z}' := \mathbf{z} + \mathbf{M}(\mathbf{d}_1 + \mathbf{c}_0) + \mathbf{M}'\mathbf{e}_{flood},$$

where $\mathbf{M}' = \mathbf{M}\mathbf{B}_\Lambda^{-1}$ was computed in Phase 1.

(5) **Output:** $\mathcal{B}$ runs $\mathcal{A}_2(\mathbf{b}', \mathbf{z}', \text{st})$ and outputs the bit returned by $\mathcal{A}_2$.

*Analysis of the Real Case Distribution.* We analyze the distribution of $\mathbf{b}'$ and $\mathbf{z}'$ when the input is a real MH-MLWE sample (i.e., $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$). Substituting the structure of $\mathbf{b}$ into the equation for $\mathbf{b}'$:

$$\begin{aligned}\mathbf{b}' &= \mathbf{A}'\mathbf{B}_\Lambda(\mathbf{A}\mathbf{s} + \mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{A}'\mathbf{e}_{flood} \\ &= \mathbf{A}'\mathbf{B}_\Lambda\mathbf{A}\mathbf{s} + \mathbf{A}'\mathbf{B}_\Lambda(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{A}'\mathbf{e}_{flood}.\end{aligned}$$

Recall from the definition of $\mathbf{A}'$ in Phase 1 (Game 3) that $\mathbf{A}'$ is sampled from the orthogonal complement $I_q^\perp(\mathbf{B}_\Lambda\mathbf{A})$, meaning $\mathbf{A}'\mathbf{B}_\Lambda\mathbf{A} \equiv \mathbf{0} \bmod q$. This eliminates the term containing the secret $\mathbf{s}$:

$$\mathbf{b}' = \mathbf{A}'\left(\mathbf{B}_\Lambda(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{e}_{flood}\right).$$

We define the effective error vector $\mathbf{e}' := \mathbf{B}_\Lambda(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{e}_{flood}$. We analyze the distribution of $\mathbf{e}'$ in three steps:

(1) **Center Shift:** Let $\mathbf{e}_1 = \mathbf{e} + \mathbf{d}_1$. Since $\mathbf{e} \sim \mathcal{D}_{R^l,\sigma_e,0}$ and $\mathbf{d}_1 \sim \mathcal{D}_{R^l,\sigma_e,-\mathbf{c}_0}$, by the Gaussian convolution lemma, $\mathbf{e}_1 \approx \mathcal{D}_{R^l,\sqrt{2}\sigma_e,-\mathbf{c}_0}$.

(2) **Support Shift:** Adding $\mathbf{c}_0$ shifts the center to $\mathbf{0}$ and the support to the integer coset $R^l + \mathbf{c}_0$. Let $\mathbf{e}_2 = \mathbf{e}_1 + \mathbf{c}_0$. Then $\mathbf{e}_2 \sim \mathcal{D}_{R^l+\mathbf{c}_0,\sqrt{2}\sigma_e,0}$.

(3) **Basis Transformation & Flooding:** Multiplying by $\mathbf{B}_\Lambda$ transforms the support to $\Lambda + \mathbf{c}$ (since $\mathbf{B}_\Lambda\mathbf{c}_0 = \mathbf{c}$) and the covariance to $\Sigma_3 = 2\sigma_e^2\mathbf{B}_\Lambda\mathbf{B}_\Lambda^T$. Finally, adding $\mathbf{e}_{flood}$ (which is in $\Lambda$ and has covariance $\Sigma_{flood}$) results in a total covariance of $\Sigma_3 + \Sigma_{flood} = (\sigma_e')^2\mathbf{I}$.

Thus, the effective error $\mathbf{e}'$ follows the distribution $\mathcal{D}_{\Lambda+\mathbf{c},\sigma_e',0}$, which matches exactly the error distribution required for the Real CEK-MH-MLWE instance.

Next, we verify the hint consistency. Substituting $\mathbf{z} = \mathbf{M}\mathbf{e} + \mathbf{y}$ into the equation for $\mathbf{z}'$:

$$\begin{aligned}\mathbf{z}' &= (\mathbf{M}\mathbf{e} + \mathbf{y}) + \mathbf{M}(\mathbf{d}_1 + \mathbf{c}_0) + \mathbf{M}'\mathbf{e}_{flood} \\ &= \mathbf{M}(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{M}'\mathbf{e}_{flood} + \mathbf{y}.\end{aligned}$$

Using the relation $\mathbf{M} = \mathbf{M}'\mathbf{B}_\Lambda$, we can rewrite the first term:

$$\begin{aligned}\mathbf{z}' &= \mathbf{M}'\mathbf{B}_\Lambda(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{M}'\mathbf{e}_{flood} + \mathbf{y} \\ &= \mathbf{M}'\left(\mathbf{B}_\Lambda(\mathbf{e} + \mathbf{d}_1 + \mathbf{c}_0) + \mathbf{e}_{flood}\right) + \mathbf{y} \\ &= \mathbf{M}'\mathbf{e}' + \mathbf{y}.\end{aligned}$$

This is exactly the form of the hint in the Real CEK-MH-MLWE distribution, with the target hint noise $\mathbf{y}' = \mathbf{y}$.

Therefore, if the input to $\mathcal{B}$ is a Real MH-MLWE sample, the input to $\mathcal{A}_2$ is a Real CEK-MH-MLWE sample. Combined with the result from $\text{Game}_1$ (where Random maps to Random), we conclude that $\mathcal{B}$ successfully simulates the view of $\mathcal{A}$, and thus $(1-\varepsilon)\text{Adv}_{\mathcal{A}}^{\text{CEK-MH-MLWE}}(\lambda) \le \text{Adv}_{\mathcal{B}}^{\text{MH-MLWE}}(\lambda) + \varepsilon$. □

## C Derivation of Error Bounds

To solve the optimization problem in Section 5, we must quantify the parameter $B$, which is the high-probability upper bound on the squared spectral norm of the integer representation of the aggregated error matrix $\mathbf{E}_{\text{agg}}$.

The total number of homomorphic additions is $m$, and each involves a matrix $\mathbf{E}_k$ of size $m_q \times l'$. The aggregated matrix $\mathbf{E}_{\text{agg}}$ has dimensions $L \times l'$, where $L = m \cdot m_q$. Its integer representation, $\bar{\mathbf{E}}_{\text{agg}}$, is an $(m \cdot m_q \cdot d) \times (l'd)$ matrix containing $N_{\text{coeffs}} = (m \cdot m_q) \cdot l' \cdot d$ integer coefficients.

The coefficients are sampled from a discrete Gaussian distribution over $\mathbb{Z}$ with parameter $s$. Using the tail bound from Lemma 2.5, the probability that the absolute value of a single coefficient $e$ exceeds a threshold $t$ is $\Pr[|e| > t] \le 2\exp(-\pi t^2/s^2)$. To bound the maximum coefficient magnitude in the entire matrix, we apply the union bound. We require the probability that *any* coefficient exceeds our bound $E_{max}$ to be negligible ($2^{-\lambda}$):

$$N_{\text{coeffs}} \cdot 2\exp\left(-\frac{\pi E_{max}^2}{s^2}\right) \le 2^{-\lambda} \tag{97}$$

Solving for $E_{max}$, we obtain the high-probability bound:

$$E_{max} \approx s\sqrt{\frac{\ln(2 \cdot N_{\text{coeffs}} \cdot 2^\lambda)}{\pi}} \tag{98}$$

Finally, we bound the squared spectral norm $B \ge \|\bar{\mathbf{E}}_{\text{agg}}^T\|_2^2$ using matrix norm inequalities ($\|\mathbf{M}\|_2^2 \le \|\mathbf{M}\|_\infty\|\mathbf{M}\|_1$). The infinity-norm is bounded by $l'd \cdot E_{max}$ and the 1-norm by $(m \cdot m_q)d \cdot E_{max}$. Combining these yields:

$$B \approx (l'd \cdot E_{max}) \cdot ((m \cdot m_q)d \cdot E_{max}) = (m \cdot m_q)l'd^2E_{max}^2 \tag{99}$$