Towards Practical Iterative Rejection Sampling: A Compact and Efficient Signature over Module Lattices

Yifan Ming[1], Jipeng Zhang[2], Zihan Liu[1], Guofeng Tang[3], Pengfei Chen[4], Yutao Sun[5], Si Gao[1], Cong Zhang[4,✉], and Long Chen[1,✉]

[1] Institute of Software, Chinese Academy of Sciences, China
mingyifan2024@iscas.ac.cn, liuzihan2024@iscas.ac.cn, gaosi@iscas.ac.cn, chenlong@iscas.ac.cn
[2] National University of Singapore, Singapore
jp-zhang@outlook.com
[3] Singapore Management University, Singapore
tang.guofeng789@gmail.com
[4] Zhejiang University, China
chenpengfei155@gmail.com, congresearch@zju.edu.cn
[5] Beijing Electronic Science and Technology Institute, China
sun_yutao@foxmail.com

Abstract. Lattice signatures face a strict trade-off among compactness, implementation simplicity, and reliance on standard lattice assumptions: ML-DSA-44 requires a 2420-byte signature (3732 bytes combined) and HAETAE-120 takes 1474 bytes (2466 bytes combined), while Falcon-512 achieves 555 bytes but relies on complex floating-point arithmetic. We propose Shuttle, a compact Fiat–Shamir signature built on a standard MLWE public-key structure with unforgeability bound to MSIS in the random oracle model. At NIST Level I, Shuttle achieves a signature size of 1175 bytes (and 2167 bytes combined)—a 51% reduction in signature size over ML-DSA-44 and 20% smaller than HAETAE-120 using purely integer arithmetic. Shuttle resolves prior compact schemes' limitations via three core techniques: (1) replacing secret-dependent rejection in the iterative sampling loop with a deterministic transition bounded by Rényi divergence, relegating restarts solely to public bounds and encoding checks (occurring with negligible probability $\approx 2^{-30}$); (2) reformulating transition logic in the logarithmic domain into simple integer interval comparisons; and (3) employing an asymmetric stretch-and-compress mechanism to offset MLWE parameter expansion. By eliminating secret-dependent rejection from the inner loop, Shuttle achieves constant-time execution with fast signing (1406k cycles, about 3× faster than HAETAE-120) and verification faster than ML-DSA-44.

1 Introduction

As Post-Quantum Cryptography (PQC) advances towards practical deployment, lattice-based digital signatures have become a core component in the migration

of public-key cryptography due to their computational efficiency, solid security proofs, and implementation maturity. Among various lattice signature design paradigms, the *Fiat-Shamir with aborts* framework introduced by Lyubashevsky [14,15] is undoubtedly one of the most successful and widely adopted methods. In this paradigm, the signer uses rejection sampling to ensure that the output signature distribution is completely independent of the secret key, thereby hiding the secret-dependent shift.

Following this design trajectory, several highly influential standards and schemes have emerged. For instance, Dilithium [6], which has been officially standardized by NIST as ML-DSA [16], achieves excellent practical performance by employing uniform distribution sampling and simple, robust arithmetic. The recent HAETAE scheme [3] further reduces signature sizes while maintaining high efficiency by introducing bimodal rejection sampling over hyperballs. However, to achieve extreme compactness, traditional FS signatures often rely on discrete Gaussian distributions (such as the early BLISS scheme [5]), which typically introduce complex sampling logic and potential side-channel security vulnerabilities [12,9,17].

Iterative Rejection Sampling for Compact Signatures. Recently, Gärtner [11] introduced an iterative rejection-sampling framework for compact Fiat–Shamir lattice signatures. The main idea is to exploit the sparsity of the challenge. In a standard bimodal construction, the signer samples a masking vector $\mathbf{y}$, derives a sparse challenge c, and forms a response $\mathbf{z} = \mathbf{y} + sc'$, where c' is a signed version of c. The rejection sampler must then hide the full secret-dependent shift sc'. Since this shift is the sum of several secret-key rotations, its norm can be significantly larger than that of each individual summand, forcing the target distribution to be sufficiently wide.

Gärtner's key observation is that the signs of the non-zero challenge coefficients can be selected one at a time. If $c = \sum_{i=1}^{\tau} X^{j_i}$, then one can write $c' = \sum_{i=1}^{\tau} b_i X^{j_i}$ with $b_i \in \{-1, +1\}$ and construct the response iteratively as $\mathbf{z}_0 = \mathbf{y}$ and $\mathbf{z}_i = \mathbf{z}_{i-1} \pm v_i$, where $v_i = X^{j_i} s$. Each rejection-sampling step therefore only has to handle the short shift v_i, rather than the full shift sc'. Since the rejection probability is governed by the ratio between the Gaussian width and the shift norm, this permits the use of a much narrower response distribution.

The key technical component is a local transition associated with a shift v. Given a current state $\mathbf{y}$, the transition chooses between the two candidates $\mathbf{y}+v$ and $\mathbf{y}-v$, corresponding to the two possible signs of the next challenge coefficient. These two choices cannot simply be made uniformly, since doing so would generally distort the distribution of the updated state. Instead, Gärtner defines two probability functions $f_v(\mathbf{y})$ and $g_v(\mathbf{y})$, which give the probabilities of moving to $\mathbf{y} + v$ and $\mathbf{y} - v$, respectively; with the remaining probability, the transition outputs $\perp$ and the signing attempt is restarted. The functions f_v and g_v are designed to enforce an exact distribution-preservation property: conditioned on success, if the input state follows the target Gaussian distribution, then so does the output state. This requires a local balance of Gaussian probability mass be-

tween neighboring points on the one-dimensional lattice line $\mathbf{y}+\mathbb{Z}v$. Solving this balance condition leads to formulas for f_v and g_v expressed through alternating infinite sums of Gaussian ratios, involving terms of the form $\rho_g(\mathbf{y}+tv)/\rho_g(\mathbf{y})$ for integers t. Thus, Gärtner's compactness mechanism is not merely a heuristic sign-selection rule; it is an exact rejection-sampling transition that carefully couples sign selection with Gaussian mass correction.

This insight leads to a major reduction in signature size. By replacing one large rejection step with a sequence of local transitions for the short shifts $v_i = X^{j_i}s$, Gärtner's framework achieves Fiat–Shamir lattice signatures whose combined signature and verification-key size approaches that of compact hash-and-sign schemes such as Falcon [19]. It therefore represents an important conceptual breakthrough for compact lattice signatures within the Fiat–Shamir paradigm.

At the same time, the same mechanism also exposes the main implementation pressure points. The compactness gain relies on exact local correction: each transition uses the probability functions f_v and g_v to balance Gaussian mass, and it aborts when this correction cannot be performed within the prescribed probability budget. Consequently, the abort behavior, the exactness of the implemented transition kernel, and the evaluation of the Gaussian-ratio probabilities become central components of the signing algorithm rather than peripheral implementation details.

Implementation Challenges of Iterative Sampling. However, this compactness mechanism also makes rejection sampling a central component of the signing algorithm. The distributional correction is achieved through an abort-based rejection procedure that is performed repeatedly during signing. While theoretically elegant, this raises a natural question: *can the compactness benefit of iterative sign selection be preserved while making the algorithm more implementation-friendly?* The challenge is not simply the use of discrete Gaussian distributions, for which efficient and constant-time implementations have been extensively studied. Rather, it is the abort-based nature of the framework that creates several aspects requiring additional care in a practical implementation:

1. **Abort events as potential leakage samples.** In the original iterative rejection-sampling framework, each transition may output $\perp$, and the signing attempt is restarted if any transition aborts. The rejection event is generated by a probability comparison whose threshold depends on the current state $\mathbf{y}$ and the secret-dependent shift $v = X^j s$. More concretely, the functions $f_v(\mathbf{y})$ and $g_v(\mathbf{y})$ are expressed through Gaussian ratios, which depend on quantities such as $\langle \mathbf{y}, v\rangle$ and $\|v\|^2$. Therefore, the abort bit is not a purely public random event; it is correlated with the secret-dependent transition being performed.
   The security proof hides this information by discarding rejected attempts and analyzing only the distribution conditioned on success. A physical implementation, however, may expose rejection events through timing, power,

or micro-architectural behavior before they are discarded. Such observations can provide auxiliary samples correlated with the secret-dependent inner products used in the transition. This is particularly relevant in the iterative setting, where the round in which an abort occurs also indicates which rotation of the secret key was being processed. Thus, abort-based rejection sampling introduces a natural side-channel leakage surface that can be exploited by adversaries without any specialised equipment.

2. **Exact per-step distribution preservation.** The original analysis relies on a strong invariant: conditioned on success, every transition preserves the target distribution exactly. This is a powerful theoretical property, but it is also an exact algebraic condition on the transition probabilities. The functions f_v and g_v are chosen so that the probability mass moved from the two inputs $\mathbf{y} + v$ and $\mathbf{y} - v$ balances perfectly and the successful output distribution is identical to the target distribution.
   This exact invariant is fragile under implementation. The probabilities f_v and g_v are defined through infinite alternating sums of Gaussian ratios, whereas an actual implementation can only use truncated sums, finite-precision arithmetic, and approximate comparisons. These approximations do not merely perturb a number; they define a different transition kernel. Once the kernel is changed, the exact per-step identity used in the proof no longer holds automatically. A concrete implementation must therefore provide an additional argument that the implemented kernel remains close enough to the ideal one and that the resulting error does not accumulate dangerously over the iterative process. This makes approximation error a central proof issue rather than a routine implementation detail.
3. **Regularity of probability evaluation.** A direct implementation of the transition probabilities involves Gaussian ratios, or equivalently exponential terms. This is not merely a matter of arithmetic cost. The implementation history of lattice signatures shows that Gaussian-related probability tests are among the most delicate components to implement securely. In BLISS, Gaussian sampling and rejection sampling were exploited by cache and timing side-channel attacks [12,17]. In Falcon, compactness comes with a substantially more complex signer, which relies on floating-point arithmetic and carefully engineered discrete Gaussian sampling routines, including exponential-type Bernoulli tests and high-precision approximations [19,2,13].
   The same type of difficulty arises in the iterative rejection-sampling framework. The transition probabilities $f_v(\mathbf{y})$ and $g_v(\mathbf{y})$ are defined through Gaussian ratios involving the secret-dependent shift v. A direct implementation would therefore have to evaluate, or approximate, exponential quantities repeatedly inside the transition loop, once for each non-zero challenge coefficient. Such computations require careful range reduction, fixed- or floating-point approximation, table access, and probability comparison, while preserving both distributional accuracy and side-channel regularity. Thus, the probability-evaluation step is a genuine engineering bottleneck, and an implementation-oriented approach should reduce the online transition to simple arithmetic and comparisons whenever possible.

These observations motivate a different way to harness the power of iterative sampling. Instead of using an abort branch to enforce exact per-step distribution preservation, a fundamental question arises: *Can the signer always output one of the two shifted candidates and account for the induced distributional distortion directly in the security proof?* Successfully answering this question would allow us to retain the compactness advantage of iterative sign selection while moving toward a more regular and implementation-friendly Fiat–Shamir lattice signature with rejection-free iterative sampling.

1.1 Our Contributions

We answer this question affirmatively. Driven by the motivation to bridge the gap between the theoretical elegance of iterative sampling and the demands of practical engineering, we propose SHUTTLE, a compact and regular Fiat–Shamir lattice signature scheme with rejection-free iterative sampling. Our core contribution is to resolve the structural and computational bottlenecks of exact iterative rejection, replacing them with a framework designed from the ground up for implementation regularity. Specifically, our contributions are as follows:

1. Secret-Dependent-Rejection-Free Iterative Sampling via Rényi Divergence. We depart from the traditional requirement of exact per-step distribution preservation. Instead of enforcing local Gaussian balance through abort branches inside the secret-dependent loop, SHUTTLE allows each iterative transition to always output one of two shifted candidates. We treat the resulting distributional distortion not as a failure, but as an explicit error that is accumulated and strictly bounded via Rényi divergence in the security proof. This perspective shift completely eliminates the abort-based control flow from the inner iterative loop, transforming the core sampling process into a deterministic-length sequence of transitions. While outer-layer restarts are retained solely for public-parameter bounds and encoding checks, eliminating secret-dependent rejections fundamentally removes the side-channel leakage surfaces and timing variability associated with traditional rejection sampling.

2. Regular Probability Evaluation in the Logarithmic Domain. To address the complexity of evaluating Gaussian ratios inside the secret-dependent loop, we reformulate the transition logic in the logarithmic domain. Rather than computing or approximating exponential functions to determine transition probabilities, we show that the randomized choice between the two candidate shifts can be executed using simple integer or fixed-point arithmetic. This substantially reduces the number of transcendental-function evaluations, thereby lowering online signing overhead and yielding a highly regular execution profile that naturally aligns with constant-time implementation requirements.

3. Standard MLWE/MSIS Assumptions with Tailored Compression. While Gärtner's original framework relies on the non-standard NTWE assumption to achieve extreme compactness, SHUTTLE adopts a public-key structure based on standard Module-LWE (MLWE) for key indistinguishability, with unforgeability bound to bimodal self-target MSIS (which reduces to standard MSIS

in the random oracle model). To counteract the inherent size overhead introduced by switching away from NTWE to standard lattices, we propose a customized compression technique tailored specifically to the MLWE secret-key structure. This modular technique effectively offsets parameter expansion, allowing SHUTTLE to maintain highly competitive signature sizes while resting on a standard lattice foundation.

To demonstrate the practical efficiency of SHUTTLE, we provide concrete parameter sets targeting NIST security levels I, III, and V. As summarized in Table 2, our scheme achieves a remarkable balance between standard assumptions, implementation friendliness, and compactness. At Level I, SHUTTLE achieves a combined public key and signature size of 2167 bytes. This is smaller than the primary NIST standard ML-DSA-44 (Dilithium-2, 3732 bytes) and the recent compact candidate HAETAE-120 (2466 bytes). Falcon-512 remains smaller (1563 bytes) but relies on floating-point arithmetic. While eliminating secret-dependent rejection from the iterative sampling loop and using an MLWE-based public-key structure, SHUTTLE's sizes remain competitive with the non-standard NTWE-based instantiation of Gärtner (1703 bytes). Compared with the MLWE-compatible Gärtner-WR variant, SHUTTLE has a comparable combined size at Level I and smaller combined sizes at Levels III and V, demonstrating the concrete effectiveness of our tailored stretch-and-compress mechanism. [6]

1.2 Technical Overview

Our starting point is Gärtner's iterative signing framework, where a signature is constructed by sequentially processing the non-zero coefficients of the challenge. At each step, the signer transitions between states by choosing between two shifted candidates, $\mathbf{y} + v$ and $\mathbf{y} - v$, where v is a secret-dependent shift. The central conceptual barrier in this prior work is the insistence on *exact local distribution preservation*: the transition probabilities are meticulously designed to perfectly balance the Gaussian mass, and an abort-driven rejection step is invoked whenever this exact balance cannot be met.

SHUTTLE fundamentally departs from this paradigm. Instead of enforcing exactness algorithmically, we allow for approximation and absorb the resulting distortion mathematically. This conceptual shift is realized through three main technical ingredients.

Relaxing the Distributional Invariant. The core conceptual shift in SHUTTLE lies in abandoning the strict invariant of prior iterative samplers. In Gärtner's framework, every single transition is forced to output a distribution that is perfectly identical to its input (the ideal Gaussian). This exact step-by-step preservation is precisely what makes aborts unavoidable. We observe that this perfect local invariant is unnecessarily restrictive. Instead, our iterative sampler fundamentally allows the input and output distributions of each iteration to differ.

[6] Our reference implementation is publicly available at `https://github.com/Ji-Peng/SHUTTLE-NIST`.

In our rejection-free transition, the intermediate masking distribution is permitted to drift slightly at each step, yet the iterative hiding mechanism still functions correctly. Because we no longer enforce exact local preservation, the iterative transition loop never aborts and simply produces one of the two shifted candidates deterministically. To guarantee security, we only require the *final* accumulated distribution to be sufficiently close to the target. We quantify this global drift using Rényi divergence. By shifting the burden of exactness from the online algorithm (which required aborts to enforce the local invariant) to the offline security proof (which bounds the global divergence), we obtain an iterative transition loop that is inherently regular and free of secret-dependent restarts.

Logarithmic-Domain Transition Kernel. Even without aborts, evaluating the transition probabilities naively requires computing sums of Gaussian ratios—essentially evaluating multiple exponential functions of secret-dependent inner products. To bypass this computational bottleneck, we algebraically transform the randomized decision process into the logarithmic domain.

Instead of computing the transition probability directly, we reframe the randomized decision as testing whether a uniform sample u falls into a specific union of intervals, where the interval boundaries are individual Gaussian ratios. By mapping this membership test into the logarithmic domain, a check of the form $u \in [A, B)$ is equivalently evaluated as $\log u \in [\log A, \log B)$. Because the target distribution is Gaussian, the logarithmic boundaries simplify algebraically into elementary squared-norm differences. Consequently, for each transition, after a single evaluation of $\log u$ on public randomness, the remaining computation reduces to simple integer or fixed-point additions and interval comparisons. More precisely, each transition replaces several dozen exponential evaluations with one logarithm followed by approximately the same number of additions and comparisons. This mathematical simplification inherently protects the implementation from the side-channel vulnerabilities typically associated with high-precision or floating-point exponential approximations.

From NTWE compactness to MLWE compactness. The third idea concerns the underlying assumption. Gärtner's extreme compactness relies on the non-standard NTWE assumption. Instantiating the scheme from the standard MLWE assumption naively incurs a size penalty. Specifically, the MLWE secret vector inherently takes the form $s = (1, \mathbf{s}_1, \mathbf{s}_2)$. The fixed component 1 contributes zero entropy but still occupies a full ring element in the signature response, forcing a larger masking vector and inflating the signature size.

To bridge this gap, we introduce an asymmetric stretch-and-compress mechanism. The key observation is that because the first secret component is exactly 1, it is significantly shorter than the rest of the secret. Therefore, we can artificially stretch this single component by a constant factor α_1 during the sampling phase while increasing the overall Euclidean norm of the secret only mildly. This asymmetry allows us to aggressively divide and compress the corresponding first component of the signature response by the same factor α_1 before outputting it. This tailored compression effectively eliminates the redundant size overhead

of the MLWE structure, allowing SHUTTLE to base key indistinguishability on MLWE while remaining highly competitive with NTWE-based instantiations.

1.3 Related Work

The landscape of lattice-based signatures is characterized by a fundamental trade-off between computational efficiency, signature size, implementation simplicity, and the conservatism of the underlying hardness assumptions.

Standardized Fiat–Shamir Signatures. ML-DSA (Dilithium) [6,16] is the primary NIST standard for lattice-based signatures. It is built on the well-studied Module-LWE and Module-SIS assumptions and avoids complex Gaussian sampling, resulting in highly efficient signing and verification. However, its reliance on uniform sampling and conservative parameters leads to relatively large public keys and signature sizes, which can be a bottleneck in bandwidth-constrained applications.

Compact Signatures and Floating-Point Arithmetic. To achieve extreme compactness, Falcon [19] (and its variants like Mitaka [10]) utilizes the hash-and-sign paradigm over NTRU lattices. While Falcon produces very short signatures and keys, its signing procedure requires complex fast Fourier transforms (FFT) and high-precision floating-point arithmetic. This makes constant-time, side-channel-resistant implementations notoriously difficult and hardware-heavy, especially on embedded devices.

Alternative Compact Designs: HAETAE and HAWK. Recent works have explored alternative paths to compactness without floating-point arithmetic. HAETAE [3], a candidate in the ongoing NIST PQC standardization process, achieves compact sizes using Fiat–Shamir with hypercube/bimodal sampling. However, it incurs a noticeable performance penalty, suffering from slower signing times and a highly complex sampling procedure. On the other end of the spectrum, HAWK [7] achieves both remarkable efficiency and compactness using integer-only arithmetic. Yet, its security relies on the non-standard Lattice Isomorphism Problem (LIP), departing from the widely scrutinized MLWE/MSIS foundations.

2 Preliminaries

We use bold lowercase letters for vectors and bold uppercase letters for matrices. Plain lowercase letters are used for scalars or single ring elements, such as the challenge c, unless stated otherwise.

Ring elements. Let n be a power of two. We work over the cyclotomic ring $R = \mathbb{Z}[X]/(X^n+1)$, and write $R_q = R/qR \simeq \mathbb{Z}_q[X]/(X^n+1)$ for its reduction modulo q. Elements of R are represented by polynomials of degree less than n, and are identified with their coefficient vectors in $\mathbb{Z}^n$. Similarly, elements of R_q are represented by coefficient vectors in $\mathbb{Z}_q^n$. For a vector $\mathbf{x} \in R^k$, we identify $\mathbf{x}$ with

its coefficient embedding in $\mathbb{Z}^{kn}$, and define $\|\mathbf{x}\|_2$ as the Euclidean norm of this coefficient vector. Unless otherwise specified, additions and multiplications in R_q are taken modulo q and modulo $X^n + 1$. Expressions modulo $2q$ are interpreted over R_{2q}.

Throughout, we use round-half-up:

$$\mathrm{RoundHU}(x) := \left\lfloor x + \frac{1}{2} \right\rfloor.$$

For ring elements and vectors, $\mathrm{RoundHU}(\cdot)$ is applied coefficient-wise.

Discrete Gaussian distributions. For a real parameter $r > 0$ and a vector $x \in \mathbb{R}^d$, we write $\rho_r(x) = \exp(-\|x\|_2^2/2r^2)$. For a discrete set $S \subseteq \mathbb{R}^d$, we define $\rho_r(S) = \sum_{x \in S} \rho_r(x)$. The discrete Gaussian distribution over S with parameter r, denoted by $D_{S,r}$, assigns probability $D_{S,r}(x) = \rho_r(x)/\rho_r(S)$ to every $x \in S$. When the domain is clear from context, we simply write D_r.

In the analysis of our sampler, we use ρ_g to denote the ideal target discrete Gaussian distribution. In particular, when the sampler is defined over R^{l+m+1} and the Gaussian parameter is r, we write $\rho_g(x) = D_{R^{l+m+1},r}(x) = \rho_r(x)/\rho_r(R^{l+m+1})$.

Random Oracle Model (ROM). Let $\mathcal{I}_{*,S}$ be the set of all functions mapping $\{0,1\}^*$ to a finite set S. The random oracle model is an idealized model that samples h uniformly at random from $\mathcal{I}_{*,S}$. Every algorithm, including the adversary, can query an input $x \in \{0,1\}^*$ and obtain the corresponding output $h(x) \in S$.

2.1 Lattice Assumptions

The security of our scheme is based on the assumed hardness of the decisional MLWE problem and the MSIS problem. The decision MLWE problem and normal form MSIS problem that we rely upon are given by the following definitions.

Definition 1 (Search $\mathsf{MSIS}_{n,q,k,\ell,\beta}$ Problem). *For positive integers q, k, ℓ, a positive real number β and the dimension n of $\mathcal{R}_q$, we say that the advantage of an adversary $\mathcal{A}$ solving the search $\mathsf{MSIS}_{n,q,k,\ell,\beta}$ problem is*

$$\mathsf{Adv}^{\mathsf{MSIS}}_{n,q,k,\ell,\beta}(\mathcal{A}) = \Pr\left[\begin{array}{c} 0 < \|\mathbf{y}\|_2 < \beta \wedge \\ (\mathbf{A} \mid \mathbf{I}_k) \cdot \mathbf{y} = 0 \pmod q \end{array} \middle| \, \mathbf{A} \leftarrow \mathcal{R}_q^{k \times \ell}; \mathbf{y} \in \mathcal{R}_q^{k+\ell} \leftarrow \mathcal{A}(\mathbf{A}) \right].$$

Definition 2 (Decisional $\mathsf{MLWE}_{q,m,\ell,\mathcal{D}}$ Problem). *For integers m, ℓ, and a probability distribution $\mathcal{D}$, we say that the advantage of algorithm $\mathcal{A}$ in solving the decisional $\mathsf{MLWE}_{q,m,\ell,\mathcal{D}}$ problem is $\mathsf{Adv}^{\mathsf{MLWE}}_{q,m,\ell,\mathcal{D}} := |\Pr[b = 1 \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m \times \ell}; \mathbf{t} \leftarrow \mathcal{R}_q^m; b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{t})] - \Pr[b = 1 \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m \times \ell}; \mathbf{s}_1 \leftarrow \mathcal{D}^\ell; \mathbf{s}_2 \leftarrow \mathcal{D}^m; b \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2)]|$.*

The BimodalSelfTargetMSIS problem is parametrized by a size bound $B_v > 0$ and integers $q, m, \ell > 0$. Furthermore, it depends on a hash function H from $\mathcal{R}_{2q}^m \times \mathcal{M}$ to $\mathcal{C}$ where $\mathcal{M} \subseteq \{0,1\}^*$ is a message space and $\mathcal{C} \subseteq \mathcal{R}_2$ is a challenge space.

Definition 3 ($\mathsf{BimodalSelfTargetMSIS}_{\mathsf{H},q,m,\ell,B_v}$ **problem [3]).** *Given* $\mathbf{A}_0 \in \mathcal{R}_q^{m \times \ell}$ *and* $\mathbf{b} \in \mathcal{R}_q^m$ *with elements sampled uniformly at random in* $\mathcal{R}_q$*, an adversary* $\mathcal{A}$ *against the* $\mathsf{BimodalSelfTargetMSIS}_{\mathsf{H},q,m,\ell,B_v}$ *problem is to produce* $(\mathbf{z}, c, \mu) \in (\mathcal{R}^{m+\ell+1}, \mathcal{C}, \mathcal{M})$ *such that* $\mathbf{z} \neq \mathbf{0}$*,* $\|\mathbf{z}\|_2 < B_v$ *and* $\mathsf{H}(\mathbf{A}\mathbf{z} - qc\mathbf{j} \pmod{2q}, \mu) = c$*, where* $\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}] \pmod{2q}$.

In the ROM, the BimodalSelfTargetMSIS problem reduces to the standard MSIS problem [3]. In the QROM, reducing full unforgeability of Fiat–Shamir-with-aborts signatures to standard assumptions remains open, which our work does not address.

2.2 Rényi Divergence

We recall the definition of the Rényi divergence in the convention used by Prest [18]. For a distribution D, we write $\mathrm{Supp}(D)$ for its support.

Definition 4 (Rényi divergence). *Let* P *and* Q *be two distributions such that* $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$*. For* $a \in (1, +\infty)$*, the Rényi divergence of order* a *from* P *to* Q *is defined as* $R_a(P\|Q) = (\sum_{x \in \mathrm{Supp}(P)} P(x)^a / Q(x)^{a-1})^{1/(a-1)}$*. For* $a = +\infty$*, it is defined as* $R_\infty(P\|Q) = \max_{x \in \mathrm{Supp}(P)} P(x)/Q(x)$.

Lemma 1 (Data Processing for Rényi Divergence). *Let* $(\Omega, \mathcal{F})$ *and* $(\Omega', \mathcal{F}')$ *be measurable spaces, and let* P, Q *be two probability measures on* $(\Omega, \mathcal{F})$*. Let* $f : \Omega \to \Omega'$ *be a measurable function, and denote by* $f(P)$ *and* $f(Q)$ *the push-forward measures induced by* f*, i.e.,*

$$f(P)(A) := P(f^{-1}(A)), \quad f(Q)(A) := Q(f^{-1}(A)) \quad \text{for all } A \in \mathcal{F}'.$$

Let $a > 1$*. If the Rényi divergence of order* a *satisfies*

$$R_a(P\|Q) \leq M,$$

then

$$R_a(f(P)\|f(Q)) \leq M.$$

2.3 From bounded to unbounded rejection

Theorem 1 (Bounded-to-unbounded rejection [4]). *Let* $\alpha \geq 0$*,* $\beta \in (0,1)$*, and let* H *be a hash function modeled as a random oracle. Assume that* $\Sigma = ((P_1, P_2), (V_1, V_2))$ *is a* (γ, β)*-correct identification protocol, and that the commitment message of* P_1 *has min-entropy* α*. Let* T *denote the runtime of one iteration of the protocol with the hash function. Let* $T' > BT$*. For any arbitrary*

adversary $\mathcal{A}$ against T'-UF-CMA security of $\mathrm{SIG}_\infty = \mathrm{FS}_\infty[\Sigma, H]$*, that issues at most Q_H queries to the random oracle H and Q_S classical queries to the signing oracle, and for any fixed integer B, the same adversary $\mathcal{A}$ against UF-CMA security of* $\mathrm{SIG}_B = \mathrm{FS}_B[\Sigma, H]$ *is such that* $|\mathrm{Adv}^{T'\text{-UF-CMA}}_{\mathrm{SIG}_\infty}(\mathcal{A}) - \mathrm{Adv}^{\text{UF-CMA}}_{\mathrm{SIG}_B}(\mathcal{A})|$ *is bounded as* $Q_S \cdot \beta^B + \frac{\beta^B \cdot 2^{-\alpha}}{(1-\beta)^3} + 2^{-\alpha} \cdot B \cdot Q_S \cdot (B \cdot Q_S + Q_H + 1)$ *in the ROM, and* $2^{-\alpha/2} \cdot \frac{3BQ_S}{2} \cdot \sqrt{B \cdot Q_S + Q_H + 1}$ *in the QROM.*

This also holds when replacing UF-CMA with UF-CMA$_1$ or sUF-CMA security.

3 Sampling Without Rejection

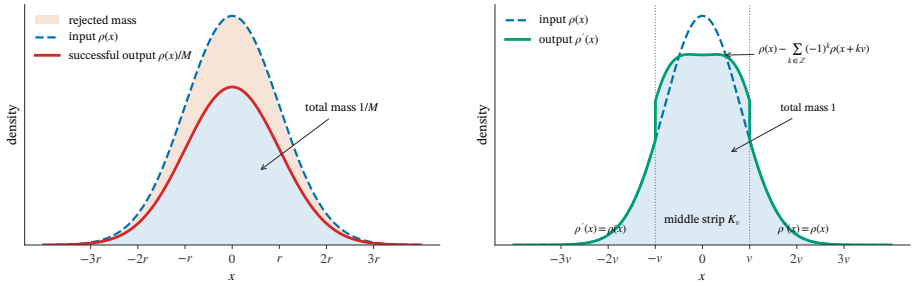
This section presents the core technical component of our construction: a sampling procedure without an explicit rejection step. We abstract away the surrounding Fiat–Shamir signature scheme and focus on the following problem. Given a masking vector $\mathbf{y}$, a secret vector s, and a sparse challenge c, the signer has to output a response of the form $\mathbf{z} = \mathbf{y} + sc'$, where c' differs from c only in the signs of its non-zero coefficients. More explicitly, if $c = \sum_{i=1}^{\tau} X^{j_i}$, then the signed challenge can be written as $c' = \sum_{i=1}^{\tau} b_i X^{j_i}$ with $b_i \in \{-1, +1\}$, and hence $sc' = \sum_{i=1}^{\tau} b_i v_i$ where $v_i = X^{j_i} s$. Thus, the iterative sampler only needs to decide, in the i-th step, whether to add or subtract the single shift v_i. The goal is to ensure that the distribution of the final response $\mathbf{z}$ is close to a target distribution that is independent of s. Traditional Fiat–Shamir-with-aborts signatures achieve this by rejection sampling. In contrast, our approach aims to obtain the same hiding effect through an iterative sampling procedure that always produces an output.

Figure 1 highlights the main difference between the original iterative rejection sampling approach and our sampling procedure. In the figure, v denotes one shift v_i used in a single iteration, rather than the full secret-dependent term sc'. The original step R_v has three possible outcomes: $\mathbf{y} - v$, $\mathbf{y} + v$, and $\perp$. The abort outcome is what allows one to preserve the input distribution exactly, conditioned on success. Our step T_v removes this third outcome. It always returns one of the two shifted candidates, and therefore the iterative procedure has a fixed number of steps.

This modification necessarily changes the invariant used in the analysis. Instead of requiring every intermediate distribution to be identical to the input distribution, we only require the final output distribution to be sufficiently close to a public target distribution. The probability p_v is therefore chosen to control the accumulated distributional error, which we quantify using Rényi divergence.

3.1 Different Input and Output Distributions

In the original iterative rejection-sampling framework, each transition is designed so that, conditioned on success, the output distribution is identical to the input target distribution. This invariant keeps every intermediate state independent



(a) Original step $\mathsf{R}_v(\mathbf{y})$

1: **return** $\mathbf{y} - v$ with probability $f_v(\mathbf{y})$
2: **return** $\mathbf{y} + v$ with probability $g_v(\mathbf{y})$
3: **return** $\perp$ with probability $1 - f_v(\mathbf{y}) - g_v(\mathbf{y})$

(b) Our step $\mathsf{T}_v(\mathbf{y})$

1: **return** $\mathbf{y} - v$ with probability $p_v(\mathbf{y})$
2: **return** $\mathbf{y}+v$ with probability $1-p_v(\mathbf{y})$

Fig. 1. Comparison between one step of iterative rejection sampling and our rejection-free transition.

of the secret shift, but it requires an accept/restart loop, increasing both the signing cost and the timing variability.

One may reduce the rejection probability by widening the sampling distribution, but this sacrifices the most compact parameter regimes. Our approach instead removes the abort outcome and allows the intermediate distributions to change. We only require that the final output distribution remains close to a public, secret-independent target distribution.

We now make explicit how one rejection-free transition changes the distribution. Let ρ be the input distribution over the sampling domain, and let $\widetilde{\mathsf{T}}_v$ denote the implemented transition associated with the shift direction v. For an input point x, the transition outputs one of the two neighboring points $x - v$ and $x + v$. We write

$$\widetilde{p}_v(x) = \Pr[\widetilde{\mathsf{T}}_v(x) = x - v],$$

and therefore

$$\Pr[\widetilde{\mathsf{T}}_v(x) = x + v] = 1 - \widetilde{p}_v(x).$$

Equivalently, the transition kernel is

$$\widetilde{\mathsf{T}}_v(x, y) = \begin{cases} \widetilde{p}_v(x), & \text{if } y = x - v, \\ 1 - \widetilde{p}_v(x), & \text{if } y = x + v, \\ 0, & \text{otherwise.} \end{cases}$$

Hence, for an arbitrary input distribution ρ, the output distribution is

$$\rho'_v(y) = \widetilde{\mathsf{T}}_v(\rho)(y) = \sum_x \rho(x)\widetilde{\mathsf{T}}_v(x,y).$$

Since the only preimages of y are $y+v$ and $y-v$, this becomes

$$\rho'_v(y) = \rho(y+v)\widetilde{p}_v(y+v) + \rho(y-v)\big(1-\widetilde{p}_v(y-v)\big).$$

This formula is the basic quantity that we need to control. In the ideal case, the transition probability is chosen so that the target distribution ρ_g is nearly invariant under this rejection-free step, namely

$$\widetilde{\mathsf{T}}_v(\rho_g)(y) \approx \rho_g(y) \qquad \text{for every } y.$$

More explicitly, under Gaussian input ρ_g, the one-step output distribution is

$$\rho'_{g,v}(y) = \rho_g(y+v)\widetilde{p}_v(y+v) + \rho_g(y-v)\big(1-\widetilde{p}_v(y-v)\big).$$

Thus the pointwise relative deviation from the target distribution is

$$\frac{\rho'_{g,v}(y)}{\rho_g(y)} - 1 = \frac{\rho_g(y+v)\widetilde{p}_v(y+v) + \rho_g(y-v)\big(1-\widetilde{p}_v(y-v)\big)}{\rho_g(y)} - 1.$$

We define the Gaussian-input relative error of the implemented transition by

$$\varepsilon_v = \mathrm{err}_{\rho_g}\big(\widetilde{\mathsf{T}}_v(\rho_g)\big) = \max_y \left| \frac{\widetilde{\mathsf{T}}_v(\rho_g)(y)}{\rho_g(y)} - 1 \right|.$$

Using the explicit expression above, this is

$$\varepsilon_v = \max_y \left| \frac{\rho_g(y+v)\widetilde{p}_v(y+v) + \rho_g(y-v)\big(1-\widetilde{p}_v(y-v)\big)}{\rho_g(y)} - 1 \right|.$$

Here $\widetilde{p}_v$ denotes the probability realized by the implementation. Thus ε_v includes the intrinsic distortion of the rejection-free transition together with any error due to truncation and logarithm approximation. We bound these three contributions separately below and combine them multiplicatively. The following lemma gives the composition rule used throughout this section.

Lemma 2 (Composition of relative errors). *Let ρ be an input distribution satisfying $\mathrm{err}_{\rho_g}(\rho) \le \delta$, and let $\widetilde{\mathsf{T}}_v$ be a transition kernel satisfying $\mathrm{err}_{\rho_g}(\widetilde{\mathsf{T}}_v(\rho_g)) \le \varepsilon$. Then $\mathrm{err}_{\rho_g}(\widetilde{\mathsf{T}}_v(\rho)) \le (1+\delta)(1+\varepsilon) - 1$.*

Applying Lemma 2 iteratively, we obtain the following bound for the sampler when the initial input distribution is the ideal discrete Gaussian distribution. Let $\rho_0 = \rho_g$ and $\rho_i = \widetilde{\mathsf{T}}_{v_i}(\rho_{i-1})$ for $i = 1, \ldots, \tau$. Assume each implemented transition satisfies $\mathrm{err}_{\rho_g}(\widetilde{\mathsf{T}}_{v_i}(\rho_g)) \le \varepsilon_{v_i}$. Then the final output distribution ρ_τ satisfies $\mathrm{err}_{\rho_g}(\rho_\tau) \le \prod_{i=1}^{\tau}(1+\varepsilon_{v_i}) - 1$. In particular, if $\varepsilon_{v_i} \le \varepsilon$ for all i, then $\mathrm{err}_{\rho_g}(\rho_\tau) \le (1+\varepsilon)^\tau - 1$.

It remains to translate this final relative-error bound into a bound on the multiplicative Rényi divergence. We use the following relative-error lemma of Prest [18, Lemma 3].

Lemma 3 (Relative-error bound for Rényi divergence [18, Lemma 3]). *Let P and Q be two distributions with common support. Suppose that there exists $0 < \delta < 1$ such that $1 - \delta \le P(x)/Q(x) \le 1 + \delta$ for every x. Then, for every $a > 1$, $R_a(P\|Q) \le (1 + \frac{a(a-1)\delta^2}{2(1-\delta)^{a+1}})^{1/(a-1)} \underset{\delta\to 0}{\approx} 1 + \frac{a\delta^2}{2}$.*

Applying Lemma 3 with $P = \rho_\tau$, $Q = \rho_g$, and $\delta_\tau = \prod_{i=1}^{\tau}(1 + \varepsilon_{v_i}) - 1$ gives

$$R_a(\rho_\tau\|\rho_g) \le \left(1 + \frac{a(a-1)\delta_\tau^2}{2(1-\delta_\tau)^{a+1}}\right)^{1/(a-1)}.$$

We use order $a = 2\lambda$ and require this multiplicative divergence to be at most $1 + 2^{-66}$. It therefore remains to bound the three one-step error contributions.

3.2 Discrete Gaussian Distributions

We now instantiate the target distribution ρ_g as the discrete Gaussian distribution used by the sampler, namely $\rho_g = D_{R^{l+m+1},r}$. Thus $\rho_g(x) = \rho_r(x)/\rho_r(R^{l+m+1})$ for every $x \in R^{l+m+1}$. Throughout this subsection, ρ_g denotes this normalized discrete Gaussian distribution. Since all ratios below are taken between values of the same distribution, the normalizing factor cancels out.

We follow the notation of Gärtner [11] and define $S_v(y) = \sum_{i=0}^{\infty}(-1)^i \frac{\rho_g(y+iv)}{\rho_g(y)}$. Here v denotes the shift used in one iteration. In the full iterative sampler, v will be instantiated as one of the cyclic shifts of the secret vector.

We set

$$p_v(y) = \begin{cases} S_v(y), & \text{if } \langle y, v\rangle \ge 0, \\ 1 - S_v(-y), & \text{if } \langle y, v\rangle < 0 \end{cases}.$$

This is a valid probability. Indeed, when $\langle y, v\rangle \ge 0$, the terms in the alternating sum defining $S_v(y)$ have decreasing absolute values, and therefore $0 \le S_v(y) \le 1$. The case $\langle y, v\rangle < 0$ follows by applying the same argument to $-y$.

We first bound the intrinsic relative error of the exact mathematical transition.

Assume that the input distribution is the ideal discrete Gaussian ρ_g. Let $\rho'_v = \mathsf{T}_v(\rho_g)$ be the output distribution after one transition step. By definition, $\rho'_v(y) = \rho_g(y + v)p_v(y + v) + \rho_g(y - v)\big(1 - p_v(y - v)\big)$. For the above choice of p_v, this output distribution satisfies

$$\rho'_v(y) = \begin{cases} \rho_g(y) - \displaystyle\sum_{i\in\mathbb{Z}}(-1)^i\rho_g(y + iv), & \text{if } -\|v\|_2^2 \le \langle y, v\rangle < \|v\|_2^2, \\ \rho_g(y), & \text{otherwise.} \end{cases}$$

Thus, the output distribution differs from the target discrete Gaussian only in the middle strip $K_v = \{y : -\|v\|_2^2 \le \langle y, v\rangle < \|v\|_2^2\}$; outside this region, the transition preserves the ideal discrete Gaussian distribution exactly. Hence it suffices to bound the relative error on K_v, namely

$$\epsilon_{rel}(v) = \max_{y\in K_v} \left| \frac{\rho'_v(y)}{\rho_g(y)} - 1 \right| = \max_{y\in K_v} \left| \frac{\sum_{i\in\mathbb{Z}} (-1)^i \rho_g(y+iv)}{\rho_g(y)} \right|.$$

The following bound is obtained from the Poisson-summation argument used in Lemma 3 of [11]. Since the normalizing factor of ρ_g cancels in all ratios above, the same bound applies directly to our notation.

Lemma 4 (Lemma 3 of [11]). *Let $v \neq 0$ and let $\beta = r/\|v\|_2$. Then*

$$\epsilon_{rel}(v) \le \frac{2\beta\sqrt{2\pi}\,\exp\left(-\frac{\pi^2\beta^2}{2}\right)}{\exp\left(-\frac{1}{2\beta^2}\right)(1-\exp(-2\pi^2\beta^2))}.$$

To obtain a uniform relative-error bound for all iterations, we use the norm bound of the stretched secret vector. Since $s^{(\alpha_1)} = (\alpha_1, \mathbf{s}_1, \mathbf{s}_2)$ and $\|(1, \mathbf{s}_1, \mathbf{s}_2)\|_2 \le B_k$, define

$$\widetilde{B_k} := \sqrt{B_k^2 + \alpha_1^2 - 1}.$$

In the full signature scheme, each shift v_i is a cyclic shift of $s^{(\alpha_1)}$, and therefore $\|v_i\|_2 \le \widetilde{B_k}$ for all $i = 1, \ldots, \tau$. Let $\widetilde{\beta} = r/\widetilde{B_k}$, and denote the right-hand side of Lemma 4, evaluated at $\widetilde{\beta}$, by $\varepsilon_{\mathrm{id}}(\widetilde{\beta})$. Then every ideal transition step satisfies $\epsilon_{rel}(v_i) \le \varepsilon_{\mathrm{id}}(\widetilde{\beta})$.

For parameter selection, we require the one-step intrinsic error $\varepsilon_{\mathrm{id}}(\widetilde{\beta}) \le 2^{-44}$. Solving the bound of Lemma 4 for $\varepsilon_{\mathrm{id}}(\widetilde{\beta}) \le 2^{-44}$ gives $\widetilde{\beta} = r/\widetilde{B_k} \gtrsim 2.592$, or equivalently $r \ge 2.592\,\widetilde{B_k}$. The complete calculation, including the other two error sources, is given in Appendix A.3.

3.3 Implementing the Transition Probability

We now realize the ideal probability $p_v(y)$ using a finite alternating sum and an integer implementation of a logarithmic comparison.

The first issue already appears in the original iterative rejection sampling framework. We use the same truncation argument. For $S_v(y) = \sum_{i=0}^{\infty}(-1)^i \rho_g(y+iv)/\rho_g(y)$, define its truncation by $S_{v,N}(y) = \sum_{i=0}^{N}(-1)^i \frac{\rho_g(y+iv)}{\rho_g(y)}$. The following bound is from Lemma 2 of [11].

Lemma 5 (Truncation of the alternating sum). *Let v be fixed and assume $\langle y, v\rangle \ge -\|v\|_2^2$. Then $|S_v(y) - S_{v,N}(y)| \le \frac{\rho_g(Nv)}{\rho_g(v)-\rho_g(2v)}$.*

Thus, a fixed N gives an explicit truncation error. We select N by bounding the relative error of both transition branches; the concrete choices and bounds are given in Appendix A.3.

The most direct implementation of the transition samples $u \leftarrow U([0,1))$, computes the truncated probability $p_{v,N}(y)$, and compares u with this value. This requires many evaluations of Gaussian ratios. We instead rewrite $p_{v,N}(y)$ as the length of an interval union, which leads to an equivalent implementation using only one logarithm of the public random sample.

Assume for simplicity that N is odd, and write $N = 2M + 1$. The even case only adds a trivial boundary interval. For every integer t, let $A_t(y, v) := \rho_g(y + tv)/\rho_g(y)$. Define

$$\mathcal{I}_N(y,v) := \begin{cases} \bigcup_{i=0}^{M} [A_{2i+1}(y,v), A_{2i}(y,v)), & \text{if } \langle y, v\rangle \geq 0, \\ \left(\bigcup_{i=0}^{M-1} [A_{-(2i+2)}(y,v), A_{-(2i+1)}(y,v)) \right) \cup [0, A_{-N}(y,v)), & \text{if } \langle y, v\rangle < 0. \end{cases}$$

Set $p_{v,N}(y) := \mathrm{len}(\mathcal{I}_N(y, v))$. Thus N has the same meaning as in the truncated alternating sum $S_{v,N}$: no ratio with shift magnitude larger than N is used.

To avoid evaluating Gaussian ratios inside the transition loop, we take the logarithm of the uniform sample instead. Let $Q_t(y, v) := \|y\|_2^2 - \|y + tv\|_2^2$. Since $A_t(y, v) = \exp(Q_t(y, v)/(2r^2))$, the test $u \in \mathcal{I}_N(y, v)$ is equivalent to testing whether $2r^2 \log u$ lies in the interval union obtained by replacing each endpoint $A_t(y, v)$ with $Q_t(y, v)$.

We define this logarithmic image as

$$\mathcal{J}_N(y,v) := \{\, 2r^2 \log u : u \in \mathcal{I}_N(y,v) \,\},$$

with the convention $2r^2 \log 0 = -\infty$.

$\mathsf{T}_1(y, v)$

1: $u \leftarrow U([0,1))$
2: $p \leftarrow \mathrm{len}(\mathcal{I}_N(y,v))$
3: $b \leftarrow (u < p)$
4: **return** $y + (-1)^b v$

$\mathsf{T}_2(y, v)$

1: $u \leftarrow U([0,1))$
2: $b \leftarrow (u \in \mathcal{I}_N(y,v))$
3: **return** $y + (-1)^b v$

$\mathsf{T}_3(y, v)$

1: $u \leftarrow U([0,1))$
2: $\lambda \leftarrow 2r^2 \log u$
3: $b \leftarrow (\lambda \in \mathcal{J}_N(y,v))$
4: **return** $y + (-1)^b v$

Fig. 2. Three equivalent views of the truncated transition step.

Theorem 2 (Equivalence of the transition implementations). *Assume exact arithmetic and exact evaluation of* $\log u$. *Then* T_1, T_2, *and* T_3 *induce the same output distribution. Moreover,* T_2 *and* T_3 *are pointwise equivalent when given the same value of* u.

The proof is deferred to Appendix A.1.

The advantage of T_3 is that the only transcendental operation is the computation of $\log u$, which depends only on public randomness. It can therefore be generated together with the masking sample, while the secret-dependent transition loop only performs fixed arithmetic comparisons. Appendix A.2 gives a recommended integer-only realization of this logarithm.

Theorem 3 (From absolute logarithm error to relative transition error). *Let $\widehat{\log}$ be the logarithm used by the implementation. Assume that it is nondecreasing on its valid non-tail input domain and that $\widehat{\log}(u) < 0$ for $u < 1$, while $\widehat{\log}(1) = 0$. Let N be odd, and let $p_{b,N}(y)$ and $\widetilde{p}_{b,N}(y)$ denote the probabilities of branch $b \in \{+,-\}$ in the exact and implemented logarithmic tests, respectively. If*

$$\sup_u \left|\widehat{\log}(u) - \log u\right| \le \delta_{\log},$$

then

$$\left|\frac{\widetilde{p}_{b,N}(y)}{p_{b,N}(y)} - 1\right| \le C_{v,r}\big(e^{\delta_{\log}} - 1\big), \qquad C_{v,r} := \frac{1 + \exp(-\|v\|_2^2/(2r^2))}{1 - \exp(-\|v\|_2^2/(2r^2))}.$$

The proof is given in Appendix A.2. We require $\delta_{\log} \le 2^{-54}$; the integer routine of Appendix A.2 satisfies the stronger certified bound $\delta_{\log} < 2^{-55}$.

To conclude, let $\varepsilon_{\mathrm{id}}$ bound the Gaussian-input relative error of the exact rejection-free transition, let $\varepsilon_{\mathrm{trunc}}$ bound the branchwise relative error caused by truncating the alternating sum, and let $\varepsilon_{\log}$ bound the branchwise relative error caused by the logarithm approximation. The implemented one-step transition then satisfies

$$\varepsilon_{\mathrm{step}} \le (1 + \varepsilon_{\mathrm{id}}) \cdot (1 + \varepsilon_{\mathrm{trunc}}) \cdot (1 + \varepsilon_{\log}) - 1,$$

and after at most τ transitions,

$$\delta_\tau \le (1 + \varepsilon_{\mathrm{step}})^\tau - 1.$$

Lemma 3 converts δ_τ into the final multiplicative Rényi-divergence bound. The concrete truncation indices, the three error contributions, their accumulated relative errors, and the resulting security losses for all parameter sets are calculated in Appendix A.3.

4 Our Signature Scheme

At a high level, our construction follows the same design philosophy as the iterative variant of BLISS considered by Gärtner. The main difference lies in the underlying public-key structure. BLISS is based on an NTRU-type relation, while Gärtner's most compact instantiation uses the more recent NTWE assumption in order to minimize the combined size of the public key and the signature. In this work, we instead use an MLWE-based instantiation. This choice leads to a more conservative security foundation, at the cost of a less compact key structure.

4.1 Signature Compression

Our signing equation has a structure similar to that of HAETAE [3]. Hence, we use the same general hint-based compression strategy: the signer commits only

to the high-order information needed to derive the challenge, and includes a hint that allows the verifier to recover the same challenge input.

Let the response vector be $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)$, where $\mathbf{z}_1$ consists of the first $l+1$ ring components and $\mathbf{z}_2$ consists of the last m ring components. Since the last block of the derived public matrix is $2\mathbf{I}_m$, the effect of $\mathbf{z}_2$ on the commitment can be handled through a hint. Thus the signature contains $\mathbf{z}_1$ and $\mathbf{h}$, instead of the full vector $\mathbf{z}$.

Compared with the NTWE-based instantiation of Gärtner, an MLWE-based instantiation has an inherent size disadvantage, since the extended secret vector $s = (1, \mathbf{s}_1, \mathbf{s}_2)$ also contains the error part. Moreover, the first component is fixed to 1 and contributes no secret entropy, but it still requires one full masking component in $\mathbf{y}$.[7] This directly increases the response size.

We reduce this overhead by exploiting the asymmetry of s. Since the first component is much shorter than the remaining secret components, we stretch only this component during sampling and compress the corresponding response component before output. This reduces the size of the first response component, while increasing the effective secret norm only mildly. Since the masking parameter r is mainly determined by the secret norm, the required masking width increases only modestly.

Formally, define the stretched secret by

$$s^{(\alpha_1)} = \mathsf{StretchS}_{\alpha_1}(s) = (\alpha_1, \mathbf{s}_1, \mathbf{s}_2), \qquad s = (1, \mathbf{s}_1, \mathbf{s}_2).$$

Let $\mathsf{CompressY} = \mathsf{CompressY}_{\alpha_1}$ be the map that compresses the first ring component by a factor α_1, using coefficient-wise round-half-up, and leave all other components unchanged. The required property is shift compatibility: adding $\alpha_1 t$ before compression is equivalent to adding t after compression. For example, coefficient-wise maps $x_0 \mapsto \mathrm{RoundHU}(x_0/\alpha_1)$ satisfy this property.

In the signing algorithm, the sampler is run with $s^{(\alpha_1)}$, the commitment is computed from $\mathsf{CompressY}(\mathbf{y})$, and the response included in the signature is $\mathbf{z} = \mathsf{CompressY}(\widetilde{\mathbf{z}})$. The following theorem formalizes the correctness of this compression step.

Theorem 4 (Correctness of stretching compression). *Let $s = (1, \mathbf{s}_1, \mathbf{s}_2)$ and $s^{(\alpha_1)} = (\alpha_1, \mathbf{s}_1, \mathbf{s}_2)$. Suppose that the deterministic map $\mathsf{CompressY}$ satisfies the shift-compatibility condition*

$$\mathsf{CompressY}(x + s^{(\alpha_1)} c') = \mathsf{CompressY}(x) + sc'$$

for every relevant x and every signed challenge c' with $c' \equiv c \pmod 2$.

If the sampler outputs $\widetilde{\mathbf{z}} = \mathbf{y} + s^{(\alpha_1)} c'$, the signature response is $\mathbf{z} = \mathsf{CompressY}(\widetilde{\mathbf{z}})$, and the commitment is computed as $\mathbf{w} = \mathbf{A}\mathsf{CompressY}(\mathbf{y}) \pmod{2q}$, then

$$\mathbf{Az} - qc\mathbf{j} = \mathbf{w} \pmod{2q}.$$

[7] This masking component also hides the signs of the signed challenge: the first response component satisfies $z_0 = \mathrm{RoundHU}(y_0/\alpha_1) + c'$, whereas setting $y_0 = 0$ and $\alpha_1 = 1$ would reveal c'.

Moreover, for every $a > 1$, applying CompressY *cannot increase the Rényi divergence from the corresponding compressed target distribution.*

The proof and the concrete construction of CompressY are deferred to Appendix A.4.

4.2 Final Signature Algorithm

KeyGen()

repeat
  $\mathbf{A_{gen}} \leftarrow U(R_q^{m\times l})$
  $\mathbf{s}_1 \leftarrow \chi_\eta^l$, $\mathbf{s}_2 \leftarrow \chi_\eta^m$
  $b \leftarrow \mathbf{A_{gen}}\mathbf{s}_1 + \mathbf{s}_2 \pmod q$
until $B'_k < \|(1, \mathbf{s}_1, \mathbf{s}_2)\|_2 \le B_k$
return $\mathsf{pk} = (\mathbf{A_{gen}}, b)$, $\mathsf{sk} = (\mathsf{pk}, \mathbf{s}_1, \mathbf{s}_2)$

Verify$((\mathbf{A_{gen}}, b), M, \sigma)$

$\sigma \leftarrow \mathsf{sigDecode}(\sigma)$
if $\sigma = \perp$ **then**
  return reject
end if
Parse σ as $(\mathbf{z}_1, \mathbf{h}, c)$
$\widetilde{\mathbf{w}} \leftarrow [\, q\mathbf{j} - 2b \mid 2\mathbf{A_{gen}} \,] \cdot \mathbf{z}_1 - qc\mathbf{j} \pmod{2q}$
$\mathbf{w}'_h \leftarrow \mathsf{HighBits}(\widetilde{\mathbf{w}}) + \mathbf{h} \pmod{2(q-1)}$
Let z_0 be the first element of $\mathbf{z}_1$
$\mathbf{w}'_0 \leftarrow \mathsf{LSB}(z_0 - c)\mathbf{j}$
$\widetilde{c} \leftarrow \mathsf{H}(\mathbf{w}'_h \parallel \mathbf{w}'_0 \parallel M \parallel \mathbf{A_{gen}} \parallel b)$
$\mathbf{z}'_2 \leftarrow (\mathbf{w}'_h - \widetilde{\mathbf{w}} + \mathbf{w}'_0)/2 \pmod{\pm q}$
if $c = \widetilde{c}$ and $\|(\mathbf{z}_1, \mathbf{z}'_2)\|_2 \le B_v$ **then**
  return accept
else
  return reject
end if

Sign$(((\mathbf{A_{gen}}, b), \mathbf{s}_1, \mathbf{s}_2), M)$

repeat
  $\mathbf{y} \leftarrow D_r^{l+m+1}$
  $\mathbf{w} \leftarrow [\, q\mathbf{j} - 2b \mid 2\mathbf{A_{gen}} \mid 2\mathbf{I}_m \,] \cdot \mathsf{CompressY}(\mathbf{y}) \pmod{2q}$
  $\mathbf{w}_h \leftarrow \mathsf{HighBits}(\mathbf{w})$
  $\mathbf{w}_0 \leftarrow \mathsf{LSB}(\mathbf{w})$
  $c \leftarrow \mathsf{H}(\mathbf{w}_h \parallel \mathbf{w}_0 \parallel M \parallel \mathbf{A_{gen}} \parallel b)$
  $\mathbf{z} \leftarrow \mathsf{CompressY}(\mathsf{IterSample}(c, \mathbf{y}, (\alpha_1, \mathbf{s}_1, \mathbf{s}_2)))$
  Write $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)$
  $\widetilde{\mathbf{w}} \leftarrow \mathbf{w} - 2\mathbf{z}_2 \pmod{2q}$
  $\mathbf{h} \leftarrow \mathsf{HighBits}(\mathbf{w}) - \mathsf{HighBits}(\widetilde{\mathbf{w}}) \pmod{2(q-1)}$
  $\sigma \leftarrow \perp$
  if $\|\mathbf{z}\|_2 \le B_s$ **then**
    $\sigma \leftarrow \mathsf{sigEncode}(\mathbf{z}_1, \mathbf{h}, c)$
  end if
until $\sigma \neq \perp$
return σ

HighBits$(\mathbf{w})$

return $\mathrm{RoundHU}(\mathbf{w}/\alpha_h) \cdot \alpha_h \pmod{2(q-1)}$

CompressY$(\mathbf{y}^{(0)}, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(l+m)})$

return $\left(\mathrm{RoundHU}\left(\mathbf{y}^{(0)}/\alpha_1\right), \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(l+m)}\right)$

Fig. 3. Simplified description of SHUTTLE.

We now give the final signature scheme. For clarity, the algorithms are written in a simplified mathematical form: internal signature values are shown in decoded form, while seed expansion, deterministic generation of $\mathbf{A_{gen}}$, and NTT-domain arithmetic are omitted. Signing explicitly encodes its output, and verification explicitly decodes its signature input. The decoded hint belongs to $R^m_{2(q-1)}$, and each of its coefficients is represented by the unique integer in $[0, 2(q-1)) \subset [0, 2q)$. This range is enforced by the signature data structure: an out-of-range encoding does not decode to a hint, and is never reduced modulo $2(q-1)$. The

Algorithm 1 $\mathsf{IterSample}(c, \mathbf{y}, s^{(\alpha_1)})$.

Input: Challenge $c = \sum_{i=1}^{\tau} X^{j_i}$, masking vector $\mathbf{y}$,
stretched secret $s^{(\alpha_1)} = (\alpha_1, \mathbf{s}_1, \mathbf{s}_2)$.
Output: Uncompressed response $\widetilde{\mathbf{z}}$.
Let $j_1 < \cdots < j_\tau$ be the nonzero positions of c
$\widetilde{\mathbf{z}}_0 \leftarrow \mathbf{y}$
for $i = 1, \ldots, \tau$ **do**
  $v_i \leftarrow X^{j_i} s^{(\alpha_1)}$
  $\widetilde{\mathbf{z}}_i \leftarrow \mathsf{T}_3(\widetilde{\mathbf{z}}_{i-1}, v_i)$
end for
return $\widetilde{\mathbf{z}}_\tau$

map $\mathsf{sigEncode}$ is deterministic and injective, and $\mathsf{sigDecode}(\sigma) = \perp$ unless σ is the unique canonical encoding of the decoded tuple. We write $pk = (\mathbf{A}_{\mathbf{gen}}, b)$ throughout the algorithms. The matrix used for signing is the derived matrix $\mathbf{A} = [\, q\mathbf{j} - 2b \mid 2\mathbf{A}_{\mathbf{gen}} \mid 2\mathbf{I}_m \,]$, and the extended secret vector is $s = (1, \mathbf{s}_1, \mathbf{s}_2)$. By construction, we have the relation $\mathbf{A}s = q\mathbf{j} \pmod{2q}$, which is the algebraic core of the verification equation.

The signer first samples $\mathbf{y}$ from the target discrete Gaussian and computes $\mathbf{w} = \mathbf{A}\mathsf{CompressY}(\mathbf{y}) \pmod{2q}$. The challenge is derived from the high bits and the least significant bits of this commitment. Before hashing, each coefficient of $\mathbf{w}_h$, represented canonically in $[0, 2(q-1))$, is encoded as an unsigned 16-bit little-endian integer. The response vector is then computed using the rejection-free iterative sampler in Algorithm 1, which returns the uncompressed response $\widetilde{\mathbf{z}} = \mathsf{IterSample}(c, \mathbf{y}, s^{(\alpha_1)})$. Signing then sets $\mathbf{z} = \mathsf{CompressY}(\widetilde{\mathbf{z}})$, as in Figure 3. To reduce the signature size, the response is split as $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)$, and only $\mathbf{z}_1$ is included in the signature. The omitted part $\mathbf{z}_2$ is represented implicitly through the hint $\mathbf{h}$, which allows the verifier to reconstruct the challenge input.

In Algorithm 1, multiplication by X^{j_i} is performed component-wise in $R = \mathbb{Z}[X]/(X^n + 1)$. Each invocation of T_3 in Figure 2 uses fresh independent randomness; r and N are fixed by the parameter set. The sampler performs exactly τ transitions and returns

$$\widetilde{\mathbf{z}} = \mathbf{y} + s^{(\alpha_1)} c', \qquad c' = \sum_{i=1}^{\tau} \epsilon_i X^{j_i}, \quad \epsilon_i \in \{-1, +1\},$$

where $c' \equiv c \pmod 2$. The signs ϵ_i are implicit in the transitions and are not returned. The signing algorithm applies $\mathsf{CompressY}$ to this response; norm checking and encoding remain part of its outer restart loop.

Figure 3 summarizes the three algorithms. In verification, the verifier first computes $\widetilde{\mathbf{w}} = [\, q\mathbf{j} - 2b \mid 2\mathbf{A}_{\mathbf{gen}} \,] \cdot \mathbf{z}_1 - qc\mathbf{j} \pmod{2q}$. This is the commitment reconstructed without the omitted component $\mathbf{z}_2$. The hint $\mathbf{h}$ corrects the high-bit information, giving $\mathbf{w}'_h$, while the least significant bit part $\mathbf{w}'_0$ is recovered from the first component of $\mathbf{z}_1$. The verifier then recomputes the challenge and reconstructs a candidate omitted component $\mathbf{z}'_2$. The signature is accepted only

if the recomputed challenge matches and the reconstructed response satisfies the verification norm bound B_v.

5 Security Proof

5.1 UF-CMA and sUF-CMA Security

Since our scheme does not impose a maximum number of rejections, we first prove the security of the scheme under bounded rejections. Then, by invoking Theorem 1 of [4], we extend the security proof to the unbounded case. Finally, we demonstrate in Appendix A that our signature scheme satisfies the necessary conditions for this theorem, thereby establishing a security reduction from CMA to NMA for our scheme.

In the reduction from CMA to NMA, the inability to sample from a perfect Gaussian distribution necessitates an intermediate step in our zero-knowledge proof. Let $\mathcal{T}_1$ be the real signing distribution (from the initial games), $\mathcal{T}_2$ be the hypothetical distribution assuming perfect Gaussian sampling (as in Game 2.1), and $\mathcal{T}_3$ be the simulator's distribution.

Both $\mathcal{T}_1$ and $\mathcal{T}_2$ rely on the secret key, whereas $\mathcal{T}_3$ is independent. We define the simulator such that we bypass the difficulty of bounding the Rényi divergence $R_a(\mathcal{T}_1\|\mathcal{T}_3)$ via the triangle inequality, as we lack a direct composition of $R_a(\mathcal{T}_1\|\mathcal{T}_2)$ and $R_a(\mathcal{T}_2\|\mathcal{T}_3)$.

At Rényi order $a > 1$, we use the bounds

$$R_a(\mathcal{T}_1\|\mathcal{T}_2) \le M_1, \qquad R_a(\mathcal{T}_2\|\mathcal{T}_3) \le M_2.$$

The factor M_1 accounts for replacing the actual discrete-Gaussian input sampler by the ideal one, whereas M_2 accounts for replacing the output of the iterative sampler by the simulator's target distribution.

The bound on the Rényi divergence $R_a(\mathcal{T}_2\|\mathcal{T}_3)$ has been established in Section 3. Hence, it remains only to control the divergence between $\mathcal{T}_1$ and $\mathcal{T}_2$. The latter two distributions are obtained by applying the same iterative procedure to different initial inputs: $\mathcal{T}_2$ uses an ideal discrete Gaussian input, while $\mathcal{T}_1$ uses the actual output of the sampler. Consequently, by Lemma 1, applying the same iterative procedure to the actual sampler output and to the ideal discrete Gaussian input cannot increase their Rényi divergence. Hence, the divergence between $\mathcal{T}_1$ and $\mathcal{T}_2$ is bounded by the Rényi divergence of the sampler itself.

Definition 5 (Rényi Honest-Verifier Zero-Knowledge with Intermediate Distribution). *Let $a > 1$, $M_1 \ge 1$, $M_2 \ge 1$, and $T_{\mathrm{Sim}} \ge 0$. A Σ-protocol is $(M_1, M_2, T_{\mathrm{Sim}})$-Rényi-HVZK of order a if there exists an intermediate distribution sampler $\mathsf{Sampler}_{\mathsf{int}}$ (utilizing the secret key sk) and a simulator Sim with total runtime at most T_{Sim}, such that for every instance x:*

Let (w', c', z') denote an honestly generated transcript, let $(w_{\mathsf{int}}, c_{\mathsf{int}}, z_{\mathsf{int}})$ denote the output of $\mathsf{Sampler}_{\mathsf{int}}(x, \mathsf{sk})$, and let (w, c, z) denote the output of $\mathsf{Sim}(x)$. The protocol satisfies:

$$R_a\big((w', c', z') \,\big\|\, (w_{\mathsf{int}}, c_{\mathsf{int}}, z_{\mathsf{int}})\big) \le M_1 \quad \textit{and} \quad R_a\big((w_{\mathsf{int}}, c_{\mathsf{int}}, z_{\mathsf{int}}) \,\big\|\, (w, c, z)\big) \le M_2.$$

This includes aborting transcripts (where the components are $\perp$).

Theorem 5. *(Adapted from [4]) Let $a > 1$, $M_1, M_2 \geq 1$, $\alpha, T_{\mathrm{Sim}} \geq 0$, $B \geq 0$, and let H be a hash function modeled as a random oracle. Assume that $\Sigma = ((P_1, P_2), (V_1, V_2))$ is an $(M_1, M_2, T_{\mathrm{Sim}})$-Rényi-HVZK of order a public-coin identification protocol, and that the commitment message of the prover has min-entropy α. Let $\mathcal{A}$ be any adversary against the UF-CMA security of $\mathrm{SIG}_B = \mathsf{FS}_B[\Sigma, H]$ that makes at most Q_H queries to the random oracle H and Q_S classical queries to the signing oracle. Consider a scenario with an array of N classical strings $x_1, \ldots, x_N$. Quantum Random Access Classical Memory (QRACM) allows loading this data into a quantum register in superposition, where a QRACM operation is defined by $U_{\mathrm{QRACM}} : |i\rangle|y_i\rangle \mapsto |i\rangle|y_i \oplus x_i\rangle$. Then there exists a quantum adversary $\mathcal{B}$ against the UF-NMA security of SIG_B such that:*

- *In the ROM, algorithm $\mathcal{B}$ runs in time $\mathrm{Time}(\mathcal{A}) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \log(B \cdot Q_S + Q_H))$, and in advantage $\mathrm{Adv}^{\mathrm{CMA}}_{\mathrm{SIG}_B}(\mathcal{A}) \leq M_1^{B \cdot Q_S} \cdot M_2^{B \cdot Q_S \cdot \frac{a-1}{a}} \cdot (\mathrm{Adv}^{\text{UF-NMA}}_{\mathrm{SIG}_B}(\mathcal{B}))^{(a-1)^2/a^2} + 2^{-\alpha} \cdot B \cdot Q_S \cdot (B \cdot Q_S + Q_H + 1)$.*
- *In the Quantum ROM (QROM), algorithm $\mathcal{B}$ runs in time $\mathrm{Time}(\mathcal{A}) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \log(B \cdot Q_S))$ with QRACM, or $\mathrm{Time}(\mathcal{A}) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \cdot (B \cdot Q_S))$ without QRACM, and in advantage $\mathrm{Adv}^{\mathrm{CMA}}_{\mathrm{SIG}_B}(\mathcal{A}) \leq M_1^{B \cdot Q_S} \cdot M_2^{B \cdot Q_S \cdot \frac{a-1}{a}} \cdot (\mathrm{Adv}^{\text{UF-NMA}}_{\mathrm{SIG}_B}(\mathcal{B}))^{(a-1)^2/a^2} + 2^{-\alpha/2} \cdot 3B \cdot Q_S \cdot \sqrt{B \cdot Q_S + Q_H + 1}$.*

Proof. The proof is similar to that of Theorem 10 of [4].

Game G_0. The first game is the UF-CMA security game.

Game G_1. In this game, the challenges of the transcripts are no longer computed by the random oracle, but are instead sampled independently and uniformly each time. Then, the random oracle is reprogrammed according to the new challenges.

To bound the distance between Game G_0 and Game G_1, we construct a wrapper $\mathcal{D}$ around $\mathcal{A}$ that uses $\mathcal{A}$ to solve a reprogramming game. It works as in Figure 16 of [4]. Note that if $b = 0$ in that figure, then $\mathcal{D}$ perfectly simulates G_0, and otherwise it perfectly simulates G_1. Therefore,

$$\left|\Pr\left[1 \Leftarrow G_0^{\mathcal{A}}\right] - \Pr\left[1 \Leftarrow G_1^{\mathcal{A}}\right]\right| \leq \left|\Pr\left[1 \Leftarrow \mathrm{Reprogram}_0^{\mathcal{D}}\right] - \Pr\left[1 \Leftarrow \mathrm{Reprogram}_1^{\mathcal{D}}\right]\right|.$$

During the game, the distinguisher $\mathcal{D}$ makes $B \cdot Q_S$ reprogramming queries and $B \cdot Q_S + Q_H + 1$ random oracle queries. In the ROM, the advantage of $\mathcal{D}$ is bounded by

$$B \cdot Q_S \cdot (B \cdot Q_S + Q_H + 1)\, 2^{-\alpha}.$$

In the QROM, it follows that the advantage of $\mathcal{D}$ is bounded by

$$3B \cdot Q_S^2 \cdot \sqrt{(B \cdot Q_S + Q_H + 1)\, 2^{-\alpha}}.$$

Game $G_{2.1}$. Let $\mathsf{Sampler}_{\mathsf{int}}$ be the intermediate distribution sampler. In this game, we modify GetTrans such that the transcripts are produced by

$\mathsf{Sampler}_{\mathsf{int}}$ using the secret key sk, effectively bridging the gap between the honest signing distribution and the simulator's output by utilizing an ideal Gaussian distribution for the sampling process. By the statistical HVZK property of the Σ-protocol, it follows that

$$\Pr[1 \Leftarrow G_1^{\mathcal{A}}] \leq M_1^{B \cdot Q_S} \cdot (\Pr[1 \Leftarrow G_{2.1}^{\mathcal{A}}])^{(a-1)/a}.$$

Game $G_{2.2}$. Let Sim be the zero-knowledge simulator for Σ. In this game we modify GetTrans such that the transcripts are now produced by Sim and without the secret key.

By the statistical HVZK property of the Σ-protocol, it follows that

$$\Pr[1 \Leftarrow G_{2.1}^{\mathcal{A}}] \leq M_2^{B \cdot Q_S} \cdot (\Pr[1 \Leftarrow G_{2.2}^{\mathcal{A}}])^{(a-1)/a}.$$

Thus,

$$\Pr[1 \leftarrow G_1^{\mathcal{A}}] \leq M_1^{B \cdot Q_S} \cdot M_2^{B \cdot Q_S \cdot \frac{a-1}{a}} \cdot \left(\Pr[1 \leftarrow G_{2.2}^{\mathcal{A}}]\right)^{\left(\frac{a-1}{a}\right)^2}$$

Games G_3 and G_4. These games are the same as in [4]. Combining the differences between each game, we obtain the final inequality.

For the reduction proof of strong unforgeability, our analysis differs substantially from that in the reference. In the reference, because the statistical error is symmetric, one can estimate the probability of event 2 in game 1 in order to bound the probability of event 2 in game 2. The setting considered here is different, since Rényi divergence is not symmetric. Therefore, we introduce a new notion, called a verifiable simulator. We prove that the simulator of our algorithm satisfies this property, and then carry out the analysis based on it.

Definition 6 (β-Verifiable Simulator). *Let $\Pi = (P, V)$ be an identification protocol. A probabilistic polynomial-time algorithm Sim is called a* β-verifiable simulator *if, on input a statement x, it outputs a transcript (w, c, z) such that $\Pr[z = \perp] \leq \beta$, and for all outputs with $z \neq \perp$, the transcript is accepting for the verifier, i.e., $V(x, w, c, z) = 1$.*

Definition 7 (Simulator Commitment Min-Entropy). *For $\alpha \geq 0$, we say that an identification scheme $\Sigma = ((P_1, P_2), (V_1, V_2))$ with instance generator Gen has* simulator commitment min-entropy α *if $H_\infty[w \mid (w, \mathsf{st}) \leftarrow \mathsf{Sim}_1(x)] \geq \alpha$ for all $(x, y) \leftarrow \mathsf{Gen}(1^\lambda)$, where Sim_1 denotes the first-message simulator of the honest-verifier zero-knowledge simulator.*

For Shuttle, both the honest prover's commitment and the simulator's commitment have min-entropy conservatively lower-bounded by more than 254.93 bits. The derivation, including the rounded discrete-Gaussian low-bit estimate and the transfer from the implemented sampler, is given in Appendix A.7.

Theorem 6. *(adapted from [4]) Let $a > 1$, $M_1, M_2 \geq 1$, $\alpha, \beta, \gamma, T_{\mathrm{Sim}} \geq 0$, $B \geq 0$, and H be a hash function modeled as a random oracle. Assume that*

$\Sigma = ((P_1, P_2), (V_1, V_2))$ is an $(M_1, M_2, T_{\mathrm{Sim}})$-Rényi-HVZK of order a public-coin identification protocol and (γ, β)-correct public-coin identification protocol, and that its zero-knowledge simulator is also β-verifiable and has min-entropy α, and that the commitment message of the prover has min-entropy α. Let $\mathcal{A}$ be any arbitrary adversary against sUF-CMA security of $\mathrm{SIG}_B = \mathrm{FS}_B[\Sigma, H]$ that issues at most Q_H queries to the random oracle H and Q_S classical queries to the signing oracle. There exist quantum adversaries $\mathcal{B}$ and $\mathcal{C}$ respectively against UF-NMA security of SIG_B and the computational unique response property of Σ as follows:

- *In the ROM, both $\mathcal{B}$ and $\mathcal{C}$ have runtime $\mathrm{Time}(A) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \log(B \cdot Q_S + Q_H))$, and the advantage is bounded by $\mathrm{Adv}^{\mathrm{CMA}}_{\mathrm{SIG}_B}(\mathcal{A}) \leq M_1^{B \cdot Q_S} \cdot M_2^{B \cdot Q_S \cdot \frac{a-1}{a}} \cdot (\mathrm{Adv}^{\text{UF-NMA}}_{\mathrm{SIG}_B}(\mathcal{B}) + \mathsf{Adv}^{\mathsf{CUR}}_{\Sigma}(\mathcal{C}) + Q_S(\beta^B + B \cdot 2^{-\alpha}))^{(a-1)^2/a^2} + 2^{-\alpha} \cdot B \cdot Q_S \cdot (B \cdot Q_S + Q_H + 1)$.*
- *In the QROM, the runtimes of $\mathcal{B}$ and $\mathcal{C}$ are $\mathrm{Time}(A) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \log(B \cdot Q_S))$ with QRACM, or $\mathrm{Time}(A) + O((T_{\mathrm{Sim}} \cdot B \cdot Q_S + Q_H) \cdot (B \cdot Q_S))$ without QRACM, and the advantage is bounded by $\mathrm{Adv}^{\mathrm{CMA}}_{\mathrm{SIG}_B}(\mathcal{A}) \leq M_1^{B \cdot Q_S} \cdot M_2^{B \cdot Q_S \cdot \frac{a-1}{a}} \cdot (\mathrm{Adv}^{\text{UF-NMA}}_{\mathrm{SIG}_B}(\mathcal{B}) + \mathsf{Adv}^{\mathsf{CUR}}_{\Sigma}(\mathcal{C}) + Q_S(\beta^B + B \cdot 2^{-\alpha}))^{(a-1)^2/a^2} + 2^{-\alpha/2} \cdot 3B \cdot Q_S \cdot \sqrt{B \cdot Q_S + Q_H + 1}$.*

Proof. The proof is similar to that of Theorem 11 of [4].

Here, the difficulty in our analysis lies in evaluating the probability that Event 2 occurs in G_3.

Event 2. For every signature query in $\mathcal{G}$, if the value of w^* has appeared in the final transcript, then the final transcript is rejected by the verifier. Let $\mathcal{G}_{\mathrm{final}}$ be the subset corresponding to these queries. We reverse the hybrid games back to G_1, where the transcripts are honestly generated. The only difference lies in our analysis of the probability of Event 2 in Game 3. Since our simulator is perfectly accepting in all non-aborting executions, Event 2 in Game 3 can occur in only two cases: (i) the simulator outputs an aborting transcript; or (ii) the commitment w^* does not appear in the final transcript but instead appears during the intermediate iterations. The probability of the latter event is bounded by the high-entropy guarantee of the simulator. Consequently,

$$\Pr[\mathrm{Event}_2 \mid G_3^A] \leq Q_S\big(\beta^B + B \cdot 2^{-\alpha}\big). \tag{1}$$

Finally, using the union bound, it holds that:

$$\begin{aligned}\big|\Pr[1 \leftarrow G_3^{\mathcal{A}}] - \Pr[1 \leftarrow G_4^{\mathcal{A}}]\big| &\leq \Pr[\mathrm{Event}_1 \mid G_3^{\mathcal{A}}] + \Pr[\mathrm{Event}_2 \mid G_3^{\mathcal{A}}] \\ &\leq \mathsf{Adv}^{\mathsf{CUR}}_{\Sigma}(\mathcal{C}) + \beta^B + B \cdot 2^{-\alpha}\ .\end{aligned}$$

The reduction from the UF-NMA game to the sUF-CMA game is similar to that of Theorem 3.

For the ROM case, we reduce the UF-NMA game to G_3 as follows. One can use the random oracle of the UF-NMA game to answer the direct hash queries of the adversary in G_3. This preserves the view of the adversary. Therefore, a similar argument and upper bound, as in the QROM, holds here.

5.2 UF-NMA Security

The UF-NMA security of the uncompressed version of our scheme is almost directly given by the assumed hardnesses of the decisional MLWE problem and the BimodalSelfTargetMSIS problem. For the compressed version of our scheme, the UF-NMA security similarly relies on the assumed hardness of these problems, but not quite as obviously. For both versions of our scheme, we argue for their UF-NMA security by the following lemma.

Lemma 6. *Let $\mathcal{A}$ be an adversary against the UF-NMA security of our signature scheme. Then, there exist two adversaries $\mathcal{B}$ and $\mathcal{B}'$, each using $\mathcal{A}$ once and performing negligible additional work, such that*

$$\mathsf{Adv}^{UF\text{-}NMA}(\mathcal{A}) \leq 1/\delta \cdot \mathsf{Adv}^{\mathsf{MLWE}}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}(\mathcal{B}) + \mathsf{Adv}^{\mathsf{BimodalSelfTargetMSIS}}_{\mathsf{H},q,m,\ell,B_v}(\mathcal{B}'),$$

where δ is the probability of an accepted secret key occurring.

Proof. We define two games, Game_0 and Game_1. Game_0 represents the UF-NMA security experiment for our scheme. The advantage of an adversary $\mathcal{A}$ in Game_0 corresponds directly to the advantage $\mathsf{Adv}^{\text{UF-NMA}}(\mathcal{A})$ against the real scheme. For Game_1, $\mathbf{A}_0$ and $\mathbf{b}$ in the public key are uniformly random instead of given by an MLWE instance. The advantage of the adversary $\mathcal{A}$ in Game_1 is defined as G_1.

For the proof, we consider that if $\mathcal{A}$ has different advantages against the two games, this can be used to solve the decisional MLWE problem. Formally, the adversary $\mathcal{B}$ is given an instance $(\mathbf{A}_0, \mathbf{b})$. $\mathcal{B}$ sets $(\mathbf{A}_0, \mathbf{b})$ as the verification key of our scheme for $\mathcal{A}$ with probability δ. The probability δ exists due to the key rejection. Under the condition that $(\mathbf{A}, \mathbf{b})$ is successfully embedded, if a forgery is created by $\mathcal{A}$ under $(\mathbf{A}, \mathbf{b})$, then $\mathcal{B}$ outputs 1; otherwise, it outputs 0. In the case where $(\mathbf{A}, \mathbf{b})$ is not successfully embedded, $\mathcal{B}$ outputs 1 with probability $1/2$.

Next, we analyze the advantage of $\mathcal{B}$ in solving the decisional MLWE problem. If $(\mathbf{A}, \mathbf{b})$ is an MLWE instance, the probability that $\mathcal{B}$ outputs 1 is $\delta \cdot \mathsf{Adv}^{\text{UF-NMA}}(\mathcal{A}) + (1 - \delta) \cdot 1/2$. If $(\mathbf{A}, \mathbf{b})$ is sampled uniformly at random, the probability that $\mathcal{B}$ outputs 1 is $\delta \cdot G_1 + (1 - \delta) \cdot 1/2$. Therefore, we conclude that the advantage of $\mathcal{B}$ in solving the MLWE problem is $\mathsf{Adv}^{\mathsf{MLWE}}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}(\mathcal{B}) = \delta \cdot |\mathsf{Adv}^{\text{UF-NMA}}(\mathcal{A}) - G_1|$.

The algorithm $\mathcal{B}'$ instead relies on whatever advantage $\mathcal{A}$ has in Game_1. To achieve this, the input to the target BimodalSelfTargetMSIS problem is used to construct the verification key that is provided to $\mathcal{A}$. This verification key is thus constructed from uniformly random $\mathbf{A}_0$ and $\mathbf{b}$. For the uncompressed version of our signature scheme, a forgery produced by $\mathcal{A}$ directly corresponds to a solution for the BimodalSelfTargetMSIS problem of $\mathcal{B}'$, which proves the statement for the uncompressed version.

For the compressed version of our signature scheme, the random oracle that $\mathcal{B}'$ provides when simulating $\mathcal{A}$ is given by $\mathsf{H}'(\mathbf{w}', w_0, \mu) = \mathsf{H}(\mathbf{w}' + w_0, \mu)$ where H is the random oracle for the BimodalSelfTargetMSIS instance. A forgery

from the signature scheme consists of a signature $(\mathbf{z}_1, \mathbf{h}, c)$ and a message μ such that $\mathsf{H}'(\mathbf{w}', w_0, \mu) = \mathsf{H}(\mathbf{w}' + w_0, \mu) = c$ for $\mathbf{w}'$ and w_0 defined as in $\mathrm{Verify}((\mathbf{z}_1, \mathbf{h}, c), \mathbf{A}, \mu)$. A valid forgery additionally corresponds to a vector $\mathbf{z}' = [\mathbf{z}_1^T, \mathbf{z}_2^T]^T$ such that $\|\mathbf{z}'\|_2 \leq B_v$, where $\mathbf{z}_2' = (\mathbf{w}' + w_0 - \tilde{\mathbf{w}}')/2$. Given such a forgery, we claim that $(\mathbf{z}', c, \mu)$ solves the target BimodalSelfTargetMSIS instance. This holds because $\mathbf{A}\mathbf{z}' = \mathbf{A}_1\mathbf{z}_1 + 2\mathbf{z}_2' = \mathbf{w}' + w_0 + qc\mathbf{j}$ and thus $\mathsf{H}(\mathbf{A}\mathbf{z}' - qc\mathbf{j}, \mu) = \mathsf{H}(\mathbf{w}' + w_0, \mu) = c$ as claimed. Furthermore, valid signatures satisfy that $\|\mathbf{z}'\|_2 \leq B_v$ and therefore the norm bound for the BimodalSelfTargetMSIS instance is satisfied. That is $G_1 \leq \mathsf{Adv}^{\mathsf{BimodalSelfTargetMSIS}}_{\mathsf{H},q,m,\ell,B_v}(\mathcal{B}')$.

6 Concrete Parameters

6.1 Selecting Parameters

Table 1 gives our concrete parameter sets, targeting approximately 120-bit, 180-bit, and 255-bit classical security. The parameters are chosen so that the key-generation and signing tail probabilities are negligible, and the output distribution of the iterative sampler remains close to the target discrete Gaussian distribution under the Rényi-divergence analysis above.

In contrast to other constructions, where rejection typically comes from an excessive Euclidean norm or from rejection sampling itself, our scheme makes the former negligible and removes the latter by design. The dominant source of signing restarts is failure of the fixed-capacity rANS encoding [8]. The concrete encoding-capacity and signature-norm estimates are given in Appendix A.5. The signature sizes in Table 1 use an encoding-overflow target of 2^{-30}.

We estimate the concrete hardness of the underlying MLWE and MSIS instances using the Lattice Estimator [1], and report the corresponding Core-SVP estimates in Table 1. These estimates serve as concrete-security guidance; the proof itself is stated in terms of the underlying MLWE and MSIS-type assumptions.

BimodalSelfTargetMSIS and MSIS security. As in previous Fiat–Shamir lattice signatures, unforgeability is related to a bimodal self-target MSIS-type problem induced by the hash challenge. As a concrete-security heuristic, we assume that this problem has comparable hardness to the standard MSIS problem with the same parameters. We therefore estimate $\mathrm{MSIS}_{q,m,l,B_v}$ with length bound B_v.

In Table 1, the MSIS value outside parentheses uses length bound B_v, while the value inside parentheses corresponds to the strong-unforgeability analysis with length bound $2B_v$. See Appendix A.6 for details.

MLWE security. Secret-key recovery from the public key is modeled as an MLWE instance with secret dimension ln, sample dimension mn, modulus q, and secret and error distributions induced by χ_η. We estimate this instance with the Lattice Estimator [1], and report the resulting Core-SVP estimates in Table 1.

Table 1. Parameter sets with no secret-dependent rejection in the iterative sampler.

Security	120	180	255	Description
n	256	256	256	Ring dimension
q	32257	15361	7681	Modulus
l	3	4	5	Number of polynomial components in $\mathbf{s}_1$
m	2	3	4	Number of polynomial components in $\mathbf{s}_2$
η	1	1	1	Centered binomial distribution parameter of $\mathbf{s}_1$ and $\mathbf{s}_2$
r	70	82	99	Parameter of y
τ	58	80	128	Hamming weight of the challenge c
α_h	128	128	128	Compression parameter for h
α_1	8	8	16	Compression parameter for the first component
B_k	25.79	30.4	34.4	Secret-key rejection bound
B_s	3233	4312	5556	Signature rejection bound
B_v	4000	5250	6638	Verification bound
pk size	992	1376	1696	Public key size
Sign size	1175	1589	2013	Signature size
Combined	2167	2965	3709	Total size
MLWE(Core-SVP)	121.7	185.4	258.1	Estimated MLWE security
MSIS(Core-SVP)	156.5 (124.6)	225.4 (182.5)	278.9 (227.8)	Estimated MSIS security

Choice of the bounds. The secret-key bound B_k is set slightly above the typical norm of $(1, \mathbf{s}_1, \mathbf{s}_2)$, so key generation succeeds with high probability while excluding rare large-norm keys; the lower bound B'_k only rules out degenerate unusually small keys. The signing bound B_s makes the signature-norm tail a lower-order contribution to the signing-restart probability, which is dominated by the fixed-capacity encoding failure. A complete numerical accounting is given in Appendix A.5. The verification bound B_v is chosen according to the target MSIS security level.

Choice of the Gaussian parameter. The bound B_k applies to the key-generation vector $(1, \mathbf{s}_1, \mathbf{s}_2)$. The iterative sampler uses the stretched vector $(\alpha_1, \mathbf{s}_1, \mathbf{s}_2)$, whose norm is bounded by $\widetilde{B_k} := \sqrt{B_k^2 + \alpha_1^2 - 1}$. We choose the masking parameter r so that $\frac{r}{\widetilde{B_k}} \geq 2.592$. For the three parameter sets, $\widetilde{B_k}$ is approximately 26.984, 31.419, and 37.926, giving ratios $r/\widetilde{B_k}$ of approximately 2.594, 2.610, and 2.610, respectively. By Lemma 4 and the concrete accounting in Appendix A.3, this choice makes the final multiplicative Rényi-divergence bound at order 2λ smaller than $1 + 2^{-66}$ for every parameter set.

Choice of the compression parameters. The compression parameter α_h is chosen so that the compressed verification condition $B_v \geq B_s + \sqrt{nm}(\alpha_h/4 + 1)$ holds [11, Lemma 5]. The choice of α_1 is heuristic: after stretching, the trivial norm contribution of the first component should be comparable to the expected norm contribution of each other secret component.

6.2 Comparison to Other Schemes

We also provide a constant-execution-time reference implementation of Shuttle. The discrete Gaussian sampler follows the isochronous techniques of Howe et al. [13], while the approximations of the exp and log functions follow the

Table 2. Comparison of security levels and sizes

Scheme	Security	Signature size	Pk size	Total
Falcon-512 [19]	120	555	897	1563
Gärtner [11]	120	775	928	1703
Gärtner-WR [11]	121	1059	1056	2115
Ours	121	1175	992	2167
HAETAE-120 [3]	119	1474	992	2466
Dilithium-2 [6]	123	2420	1312	3732
Gärtner [11]	180	1184	1056	2240
Ours	182	1589	1376	2965
Gärtner-WR [11]	180	1475	1568	3043
HAETAE-180 [3]	180	2349	1472	3821
Dilithium-3 [6]	180	3309	1952	5261
Falcon [19]	273	1280	1793	3073
Gärtner [11]	257	1694	1568	3262
Ours	255	2013	1696	3709
Gärtner-WR [11]	255	2161	2080	4241
HAETAE-260 [3]	256	2948	2080	5028
Dilithium-5 [6]	252	4627	2592	7219

Note. For the parameter sets of Gärtner [11]-WR, the compression parameter α_h (denoted by τ in the original paper) is chosen to be larger than $2r$. In this regime, the high-bit truncation is often larger than the corresponding z_2 component itself, and the excess part does not actually provide compression. Consequently, the actual sizes of their 120-bit and 180-bit parameter sets are larger than the estimated values reported in the original paper. The Dilithium sizes are measured from the reference implementation.

GALACTICS methodology of Barthe et al. [2]. We report benchmarks of the reference implementations of Shuttle, Dilithium [6], and HAETAE [3] on an 11th Gen Intel Core i7-11700K CPU at 3.60 GHz, with Turbo Boost and Hyper-Threading disabled. The updated Shuttle measurements use Ubuntu 26.04 LTS (Linux 7.0.0-30-generic) and GCC 15.2.0 with `-O3`. The Dilithium and HAETAE results are retained from the earlier Ubuntu 24.04 and GCC 13.3.0 measurements. Thus, the hardware is shared, but the software environments differ; the effects of the OS and compiler changes have not been isolated.

Table 3. Performance comparison of the reference implementations of Shuttle, Dilithium[a], and HAETAE[b]. We report average sign and verify costs in k cycles over 1000 runs on the same CPU, rounded to the nearest integer.

Security	Shuttle		Dilithium		HAETAE	
	Sign	Verify	Sign	Verify	Sign	Verify
120	1406k	277k	1331k	320k	4466k	210k
180	2117k	439k	2025k	495k	5689k	370k
255	3156k	609k	2559k	823k	8472k	462k

[a] `https://github.com/pq-crystals/dilithium`, commit 6e00625.
[b] `https://github.com/kpqc-cryptocraft/KpqClean_ver2.git`, commit 0896ebe.

As Table 3 shows, Shuttle is substantially faster than HAETAE for signing at all three security levels. Compared with Dilithium, Shuttle has a moderately higher signing cost, but a lower verification cost throughout. Overall, the reference implementation matches the design goal of Shuttle: compact signatures with balanced signing and verification costs.

Performance with AVX2. We do not yet provide an AVX2 implementation, so we only give a reasonable estimate. For both Shuttle and HAETAE, the main signing bottleneck is discrete Gaussian sampling. On our platform, the AVX2 implementation of HAETAE-120 improves signing performance by about 2.7×, from 4466k cycles to 1641k cycles. We therefore expect an AVX2 implementation of Shuttle to improve over the reference implementation by roughly 2–3×. Moreover, Shuttle should also benefit from recent SIMD optimizations for discrete Gaussian sampling [20].

Dilithium has a different bottleneck, namely NTT-based polynomial multiplication. On our platform, the AVX2 implementation of Dilithium-2 improves signing performance by about 5.8×, from 1331k cycles to 231k cycles. A reasonable takeaway is therefore the following: under AVX2, Shuttle should remain about 2–3× faster than HAETAE in signing, while Dilithium should be about 2–3× faster than Shuttle.

References

1. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015). https://doi.org/10.1515/jmc-2015-0016
2. Barthe, G., Belaïd, S., Espitau, T., Fouque, P.A., Rossi, M., Tibouchi, M.: GALACTICS: Gaussian sampling for lattice-based constant-time implementation of cryptographic signatures, revisited. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 2147–2164 (2019). https://doi.org/10.1145/3319535.3363223
3. Cheon, J.H., Choe, H., Devevey, J., Güneysu, T., Hong, D., Krausz, M., Land, G., Möller, M., Stehlé, D., Yi, M.: HAETAE: Shorter lattice-based Fiat-Shamir signatures. IACR Transactions on Cryptographic Hardware and Embedded Systems **2024**(3), 25–75 (2024)
4. Devevey, J., Fallahpour, P., Passelègue, A., Stehlé, D.: A detailed analysis of Fiat-Shamir with aborts. In: Annual International Cryptology Conference. pp. 327–357. Springer (2023)
5. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 40–56. Springer (2013)
6. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268
7. Ducas, L., Postlethwaite, E.W., Pulles, L., Yu, Y.: Hawk: Module LIP makes lattice signatures fast, compact and simple. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13792, pp. 65–94. Springer (2022). https://doi.org/10.1007/978-3-031-22966-4_3
8. Duda, J.: Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding. arXiv preprint arXiv:1311.2540 (2013)
9. Espitau, T., Fouque, P.A., Gérard, B., Tibouchi, M.: Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1857–1874 (2017)
10. Espitau, T., Fouque, P., Gérard, F., Rossi, M., Takahashi, A., Tibouchi, M., Wallet, A., Yu, Y.: Mitaka: A simpler, parallelizable, maskable variant of Falcon. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 222–253. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_8
11. Gärtner, J.: Compact lattice signatures via iterative rejection sampling. In: Annual International Cryptology Conference. pp. 514–547. Springer (2025)
12. Groot Bruinderink, L., Hülsing, A., Lange, T., Yarom, Y.: Flush, gauss, and reload–a cache attack on the BLISS lattice-based signature scheme. In: Cryptographic Hardware and Embedded Systems–CHES 2016. pp. 323–345. Springer (2016)

13. Howe, J., Prest, T., Ricosset, T., Rossi, M.: Isochronous Gaussian sampling: From inception to implementation. Cryptology ePrint Archive, Paper 2019/1411 (2019), https://eprint.iacr.org/2019/1411
14. Lyubashevsky, V.: Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In: Advances in Cryptology–ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15. pp. 598–616. Springer (2009)
15. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Advances in Cryptology–EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings 31. pp. 738–755. Springer (2012)
16. National Institute of Standards and Technology (NIST): FIPS 204: Module-Lattice-Based Digital Signature Standard. Tech. rep., US Department of Commerce (August 2024). https://doi.org/10.6028/NIST.FIPS.204
17. Pessl, P.: Analyzing the shuffling side-channel countermeasure for lattice-based signatures. In: Progress in Cryptology–INDOCRYPT 2016: 17th International Conference on Cryptology in India, Kolkata, India, December 11-14, 2016, Proceedings 17. pp. 153–170. Springer (2016)
18. Prest, T.: Sharper bounds in lattice-based cryptography using the Rényi divergence. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 347–374. Springer (2017)
19. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON: Fast-fourier lattice-based compact signatures over NTRU. Tech. rep., National Institute of Standards and Technology (NIST) (2020), submission to the NIST Post-Quantum Cryptography Standardization Project
20. Zhang, J., Zhang, J.: Vectorized Falcon-Sign implementations using SSE2, AVX2, AVX-512F, NEON, and RVV. IACR Transactions on Cryptographic Hardware and Embedded Systems **2026**(1), 302–324 (2026). https://doi.org/10.46586/TCHES.V2026.I1.302-324

A Missing proofs

A.1 Proof of Theorem 2

We give the details for the equivalence between the three implementations of the truncated transition step. Throughout this subsection, we assume exact arithmetic and exact evaluation of $\log u$. Endpoint conventions are immaterial, since all endpoints have probability zero under a continuous uniform sample.

Let N be odd and write $N = 2M + 1$. For every integer t, define

$$A_t(y, v) := \frac{\rho_g(y + tv)}{\rho_g(y)}.$$

When $\langle y, v\rangle \geq 0$, define

$$\mathcal{I}_N^+(y, v) := \bigcup_{i=0}^{M} [A_{2i+1}(y, v), A_{2i}(y, v)).$$

When $\langle y, v\rangle < 0$, define

$$\mathcal{I}_N^-(y,v) := \left(\bigcup_{i=0}^{M-1} [A_{-(2i+2)}(y,v), A_{-(2i+1)}(y,v)) \right) \cup [0, A_{-N}(y,v)).$$

Set

$$\mathcal{I}_N(y,v) := \begin{cases} \mathcal{I}_N^+(y,v), & \text{if } \langle y, v\rangle \geq 0, \\ \mathcal{I}_N^-(y,v), & \text{if } \langle y, v\rangle < 0, \end{cases}$$

and define

$$p_{v,N}(y) := \mathrm{len}(\mathcal{I}_N(y,v)).$$

By construction, the interval endpoints only use Gaussian ratios with shift magnitude at most N. Moreover, along the selected ray these Gaussian ratios are monotone, so the intervals in $\mathcal{I}_N(y,v)$ are disjoint.

For every integer t, define

$$Q_t(y,v) := \|y\|_2^2 - \|y + tv\|_2^2.$$

Since

$$A_t(y,v) = \frac{\rho_g(y+tv)}{\rho_g(y)} = \exp\left(\frac{Q_t(y,v)}{2r^2}\right),$$

we define the logarithmic image of $\mathcal{I}_N(y,v)$ by

$$\mathcal{J}_N(y,v) := \{\, 2r^2 \log u : u \in \mathcal{I}_N(y,v) \,\},$$

with the convention $2r^2 \log 0 = -\infty$.

Theorem 2 (Equivalence of the transition implementations). *Assume exact arithmetic and exact evaluation of* $\log u$. *Then* T_1, T_2, *and* T_3 *induce the same output distribution. Moreover,* T_2 *and* T_3 *are pointwise equivalent when given the same value of* u.

Proof. By definition,

$$p_{v,N}(y) = \mathrm{len}(\mathcal{I}_N(y,v)).$$

Therefore, T_1 returns $y - v$ with probability $p_{v,N}(y)$ and returns $y + v$ with probability $1 - p_{v,N}(y)$.

For T_2, the event that the output is $y - v$ is precisely

$$u \in \mathcal{I}_N(y,v).$$

Since the intervals in $\mathcal{I}_N(y,v)$ are disjoint, this event has probability

$$\mathrm{len}(\mathcal{I}_N(y,v)) = p_{v,N}(y).$$

Thus T_1 and T_2 return $y - v$ with the same probability, and hence induce the same output distribution.

It remains to compare T_2 and T_3. For every integer t,

$$A_t(y,v) = \exp\left(\frac{Q_t(y,v)}{2r^2}\right).$$

Since the logarithm is strictly increasing, membership in any interval

$$A_{t_L}(y,v) \le u < A_{t_U}(y,v)$$

is equivalent to

$$Q_{t_L}(y,v) \le 2r^2 \log u < Q_{t_U}(y,v).$$

The same equivalence also holds for intervals whose lower endpoint is 0, using the convention $2r^2 \log 0 = -\infty$.

By definition, $\mathcal{J}_N(y,v)$ is exactly the image of $\mathcal{I}_N(y,v)$ under the map $u \mapsto 2r^2 \log u$. Hence, for the same uniform sample u,

$$u \in \mathcal{I}_N(y,v) \quad \Longleftrightarrow \quad 2r^2 \log u \in \mathcal{J}_N(y,v).$$

Therefore T_2 and T_3 set the same bit b and return the same output pointwise. Consequently, all three procedures induce the same output distribution.

A.2 Recommended Integer Approximation of the Logarithm

Exponent–mantissa representation. Let $R_{\exp} \in \{0,1\}^{\kappa_{\exp}}$ and $m \in \{0,\dots,2^{\kappa_b}-1\}$ be independent uniform blocks. Set

$$e_u = 1 + \mathsf{CLZ}_{\kappa_{\exp}}(R_{\exp}), \qquad b = 1 + \frac{m}{2^{\kappa_b}},$$

where $\mathsf{CLZ}_{\kappa_{\exp}}$ returns $\kappa_{\exp}$ on the all-zero input. For $e_u \le \kappa_{\exp}$, the pair (e_u, m) identifies a bin of a continuous uniform input whose left endpoint is $u_0 = 2^{-e_u} b$. The all-zero exponent block represents the tail $u < 2^{-\kappa_{\exp}}$; $\kappa_{\exp}$ is chosen so that this tail lies below every nonzero boundary on the supported range, and collapsing it therefore changes no comparison. We take $\kappa_b = 57$. Since $\log_2 u_0 = \log_2 b - e_u$, only $\log_2 b$ on $[1,2)$ is approximated.

This choice of $\kappa_{\exp}$ can be checked without evaluating any exponential. Let

$$T_{\max} := \max |\langle y, v\rangle|, \qquad V_{\max} := \max \|v\|_2,$$

where both maxima are over the supported inputs to a transition. For $0 \le k \le N$, every nonzero endpoint is at least

$$\exp\left(-\frac{2NT_{\max} + N^2 V_{\max}^2}{2r^2}\right).$$

Consequently, it is sufficient to choose

$$\kappa_{\exp} \ge \left\lceil \frac{2NT_{\max} + N^2 V_{\max}^2}{2r^2 \ln 2} \right\rceil.$$

For our three parameter sets, the sampler support and the stretched-secret bound give

$$T_{\max} \le 920968,\ 1442420,\ 2392449, \qquad V_{\max}^2 \le 728,\ 987,\ 1438,$$

respectively. Substitution gives the sufficient lower bounds $20940, 28383, 19690$ on $\kappa_{\exp}$. We use the common value $\kappa_{\exp} = 32720$. The implementation reads the exponent string

lazily in 80-bit blocks, stopping at the first nonzero block or after 32720 bits. This is distributionally identical to sampling the whole string at once, while a second block is needed only with probability 2^{-80}. The collapsed tail is then handled exactly by the comparison logic, so the absolute-error bound below is required only on the non-tail bins.

Four-segment Q62 approximation. The two most significant mantissa bits select one of four equal segments:

$$j = \left\lfloor \frac{m}{2^{55}} \right\rfloor, \qquad \beta_j = 1 + \frac{j}{4}, \qquad x = \frac{m \bmod 2^{55}}{2^{55}}.$$

Thus $b = \beta_j + x/4$ with $x \in [0, 1)$. On segment j, a degree-13 polynomial

$$P_j(x) = 2^{-62} \sum_{i=0}^{13} c_{j,i} x^i \approx \log_2\Big(\beta_j + \frac{x}{4}\Big)$$

is used, with the 4×14 signed integer coefficient table given in Table 4. The coefficients are independent of the signature parameters; only the final scale $2r^2 \ln 2$ depends on the parameter set.

Table 4. Signed Q62 coefficients $c_{j,i}$ for $P_j(x) = 2^{-62} \sum_{i=0}^{13} c_{j,i} x^i$.

i	$c_{0,i}$	$c_{1,i}$	$c_{2,i}$	$c_{3,i}$
0	0	1484631294131014398	2697663385880076776	3723267405961586381
1	1663314137230540005	1330651309784432187	1108876091487026868	950465221274594463
2	−207914267153782834	−133065130978439107	−92406340957251847	−67890372948185266
3	34652377857794422	17742017463685572	10267371217462318	6465749804587722
4	−6497320829889967	−2661302618120493	−1283421402046772	−692758907616243
5	1299464001399016	425808407352253	171122852511038	79172446438655
6	−270720747985221	−70968007646487	−23767057140815	−9425290482070
7	58008183734025	12165731010280	3395273701197	1154114515161
8	−12680634058615	−2128477788302	−495094418506	−144257705268
9	2802591328231	377485364884	73261264270	18306992888
10	−611697204761	−66839897685	−10884612082	−2339854304
11	123141961189	11228353229	1561954161	292286931
12	−19530076391	−1554229086	−191070343	−31958889
13	1717255301	124985134	14130118	2187745

For an integer w, write $\mathrm{RoundShift}_{62}(w) = \mathrm{RoundHU}\big(w/2^{62}\big)$. The local coordinate has the exact Q62 encoding $x_Q = (m \bmod 2^{55})2^7$. Starting with $z = c_{j,13}$, Horner evaluation uses

$$z \leftarrow c_{j,i} + \mathrm{RoundShift}_{62}(z\, x_Q), \qquad i = 12, 11, \ldots, 0.$$

The integer $\widehat{\ell}_Q = z - e_u 2^{62}$ represents $\widehat{\ell} \approx \log_2 u_0$. For each parameter set, precompute $C_r = \mathrm{RoundHU}\big(2^{62}(2r^2 \ln 2)\big)$ and return $\widehat{\lambda}_Q = \mathrm{RoundShift}_{62}(C_r \widehat{\ell}_Q)$. To realize this using signed 128-bit arithmetic, write $C_r = C_{r,\mathrm{hi}} 2^{62} + C_{r,\mathrm{lo}}$ and compute

$$\widehat{\lambda}_Q = C_{r,\mathrm{hi}} z + \mathrm{RoundShift}_{62}(C_{r,\mathrm{lo}} z) - e_u C_r,$$

r	$C_{r,\mathrm{hi}}$	$C_{r,\mathrm{lo}}$
70	6792	3884743587687722514
82	9321	2044287409645675025
99	13587	327583442694609146

This represents $2^{62}(2r^2 \widehat{\log}(u))$ and is compared with the exact boundaries $2^{62} Q_t(y, v)$, using a sufficiently wide signed accumulator.

The coefficient row is selected by a masked scan of all four rows. Thus the routine uses solely integer arithmetic and has an input-independent instruction and memory-access pattern. For Table 4, $c_{0,0} = 0$; direct integer-difference analysis of the four Horner recurrences also verifies that the output is nondecreasing for $\kappa_b = 57$, including at segment and exponent boundaries, and is negative for every represented input below 1.

Proof of Theorem 3. Orient the selected ray so that $\langle y, v\rangle \geq 0$; when the original inner product is negative, replace v by $-v$, which only swaps the two branch labels. Define

$$L_k := \frac{\|y\|_2^2 - \|y + kv\|_2^2}{2r^2}, \qquad \xi_k := e^{L_k}.$$

Let $\widetilde{\xi}_k$ be the generalized threshold induced by comparing $\widehat{\log}(u)$ with L_k. Monotonicity and $|\widehat{\log}(u) - \log u| \leq \delta_{\log}$ imply

$$e^{-\delta_{\log}}\xi_k \leq \widetilde{\xi}_k \leq e^{\delta_{\log}}\xi_k, \qquad |\widetilde{\xi}_k - \xi_k| \leq (e^{\delta_{\log}} - 1)\xi_k.$$

Indeed, inputs below $e^{-\delta_{\log}}\xi_k$ must compare below L_k, whereas inputs above $e^{\delta_{\log}}\xi_k$ must compare above it. Moreover, $\xi_0 = \widetilde{\xi}_0 = 1$, and the threshold order is preserved.

For odd N, the exact probabilities on the oriented ray are

$$p_{+,N} = \sum_{k=0}^{N}(-1)^k\xi_k, \qquad p_{-,N} = \sum_{k=1}^{N}(-1)^{k+1}\xi_k,$$

and the implemented probabilities have the same expressions with ξ_k replaced by $\widetilde{\xi}_k$. Let

$$\rho_{v,r} := e^{-\|v\|_2^2/(2r^2)}.$$

Since

$$\frac{\xi_{k+1}}{\xi_k} = \exp\left(-\frac{2\langle y, v\rangle + (2k+1)\|v\|_2^2}{2r^2}\right) \leq \rho_{v,r},$$

every adjacent pair $x \geq y$ appearing in either alternating sum satisfies

$$x + y \leq \frac{1 + \rho_{v,r}}{1 - \rho_{v,r}}(x - y) = C_{v,r}(x - y).$$

An unpaired positive term satisfies the same inequality because $C_{v,r} \geq 1$. Therefore, if $T_{b,N}$ denotes the ordinary sum of all thresholds appearing in branch b, then $T_{b,N} \leq C_{v,r}p_{b,N}$. Consequently,

$$\begin{aligned}|\widetilde{p}_{b,N} - p_{b,N}| &\leq (e^{\delta_{\log}} - 1)T_{b,N} \\ &\leq C_{v,r}(e^{\delta_{\log}} - 1)p_{b,N},\end{aligned}$$

which proves the claimed relative-error bound for both branches.

Absolute-error verification. For the concrete four-segment Q62 approximation of Table 4, define the exact-polynomial error

$$E_j(x) := \log_2\left(\beta_j + \frac{x}{4}\right) - P_j(x).$$

The stationary points of E_j are the roots in $[0, 1]$ of the degree-13 equation

$$\left(\beta_j + \frac{x}{4}\right)P_j'(x) - \frac{1}{4\ln 2} = 0.$$

Isolating all real roots and evaluating E_j at them and at both endpoints gives

$$\delta_{\mathrm{poly}} := \max_{0\le j<4} \sup_{x\in[0,1]} |E_j(x)| < 1.148 \cdot 10^{-18} < 2^{-59.59}.$$

Each of the 13 rounded Horner products contributes at most half a Q62 unit. Since $0 \le x < 1$, these errors accumulate to at most

$$\delta_{\mathrm{Horner}} \le 13 \cdot 2^{-63} < 2^{-59.29}.$$

Hence the implemented base-two logarithm satisfies the conservative bound

$$\delta_2 \le \delta_{\mathrm{poly}} + \delta_{\mathrm{Horner}} < 2.558 \cdot 10^{-18} < 2^{-58.44}.$$

The mantissa bin contributes at most $\log((b + 2^{-\kappa_b})/b) \le 2^{-\kappa_b}$ to the natural-logarithm error. After division by $2r^2$, the Q62 rounding of C_r and the last product add at most $(\kappa_{\exp} + 3)2^{-63}/(2r^2)$. Thus, on every non-tail input,

$$\delta_{\log} \le 2^{-\kappa_b} + (\ln 2)\delta_2 + \frac{(\kappa_{\exp} + 3)2^{-63}}{2r^2}.$$

For $\kappa_b = 57$, $r \ge 70$, and any $\kappa_{\exp} \le 2^{20}$, this gives $\delta_{\log} < 2^{-55} < 2^{-54}$, as required by Theorem 3.

A.3 Concrete Sampler Error Accounting

Using the transcript distributions $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$ defined in the security proof, we establish at order $a = 2\lambda$ that

$$R_a(\mathcal{T}_1 \| \mathcal{T}_2) \le M_1, \qquad R_a(\mathcal{T}_2 \| \mathcal{T}_3) \le M_2.$$

The first factor accounts for replacing the concrete discrete-Gaussian input sampler by the ideal one. The second factor is obtained by instantiating the symbolic bounds of Section 3 for the iterative sampler. These are two consecutive hybrids rather than an application of a triangle inequality.

Discrete-Gaussian input sampler (M_1). We evaluate the distribution produced by the integer implementation directly. This calculation specializes the standard convolution sampler analysis of [13] to the parameters and fixed-point tables used here. Let

$$g_r(z) = \frac{\exp(-z^2/(2r^2))}{Z_r}, \qquad Z_r = \sum_{t\in\mathbb{Z}} \exp(-t^2/(2r^2))$$

be the ideal one-dimensional discrete Gaussian. The concrete sampler uses $k = 64$, a $\theta = 93$ bit reverse CDT, and the small width $s = r/k$. For table length L, its integer thresholds are

$$T_i = \left\lfloor 2^{\theta} \frac{\sum_{j=i+1}^{\infty} \exp(-j^2/(2s^2))}{\sum_{j=0}^{\infty} \exp(-j^2/(2s^2))} \right\rfloor, \qquad 0 \le i < L, \tag{2}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer. Thus the probability mass function implemented by the reverse CDT is exactly

$$\widehat{q}_r(x) = \begin{cases} 1 - T_0/2^\theta, & x = 0, \\ (T_{x-1} - T_x)/2^\theta, & 1 \le x < L, \\ T_{L-1}/2^\theta, & x = L. \end{cases} \tag{3}$$

For $x \in \{0, \ldots, L\}$ and $y \in \{0, \ldots, k-1\}$, set

$$u_{x,y} = \left\lfloor \frac{y(y+2kx)}{2^{20}} \left\lfloor \frac{2^{80}}{2r^2} \right\rfloor \right\rfloor.$$

Let $A_r(x,y) \in \{0, \ldots, 2^{63}\}$ be the Q63 integer returned by the fixed-point ApproxExp routine on the Q60 input $u_{x,y}$. Hence the implemented acceptance probability is exactly $A_r(x,y)/2^{63}$. Writing $z = kx + y$, define the unnormalized probability of emitting the magnitude z by

$$\widehat{w}_r(z) = \frac{\widehat{q}_r(x)}{k} \frac{A_r(x,y)}{2^{63}} 2^{-\mathbf{1}[z=0]}, \qquad \widehat{C}_r = \sum_{z=0}^{kL+k-1} \widehat{w}_r(z). \tag{4}$$

The last factor is the zero-fold correction. The independent sign bit then gives the exact one-coordinate output distribution

$$\widehat{g}_r(0) = \frac{\widehat{w}_r(0)}{\widehat{C}_r}, \qquad \widehat{g}_r(\pm z) = \frac{\widehat{w}_r(z)}{2\widehat{C}_r} \quad (z > 0). \tag{5}$$

Equations (2)–(5) include the CDT rounding, the Q80 reciprocal and Q60 input truncation, the Q63 exponential approximation, rejection-sampler normalization, the zero fold, and the sign bit.

For Rényi order a, the one-coordinate multiplicative divergence is therefore the following finite sum:

$$m_1(r,a) = \left(\frac{\widehat{g}_r(0)^a}{g_r(0)^{a-1}} + 2 \sum_{z=1}^{kL+k-1} \frac{\widehat{g}_r(z)^a}{g_r(z)^{a-1}} \right)^{1/(a-1)}. \tag{6}$$

The mask contains $d = n(1 + l + m)$ independently sampled coefficients. Multiplicativity of Rényi divergence for product distributions and data processing through the common remainder of the signing algorithm give

$$M_1 = m_1(r, 2\lambda)^d. \tag{7}$$

We evaluated the integer quantities in Equations (2)–(7) exactly and the exponential sums with 600-bit interval arithmetic. For the ideal normalizer Z_r, summing through $K = 2600$ leaves the explicit tail bound

$$2 \sum_{t=K+1}^{\infty} e^{-t^2/(2r^2)} \le \frac{2e^{-(K+1)^2/(2r^2)}}{1 - e^{-(2K+3)/(2r^2)}} < 2^{-494}$$

even for the largest width $r = 99$. Table 5 reports outward-rounded upper bounds.

Table 5. Concrete calculation of the discrete-Gaussian sampler factor M_1.

Security level	120	180	255
r	70	82	99
L	12	14	17
$a = 2\lambda$	240	360	510
$d = n(1 + l + m)$	1536	2048	2560
$\widehat{C}_r$	$0.7327364214\ldots$	$0.7625615323\ldots$	$0.7949744385\ldots$
$m_1(r, a) - 1$	$< 2^{-97.36}$	$< 2^{-96.45}$	$< 2^{-99.16}$
$M_1 - 1 = m_1(r, a)^d - 1$	$< 2^{-86.78}$	$< 2^{-85.45}$	$< 2^{-87.83}$

In particular, every parameter set satisfies the uniform bound

$$M_1 \le 1 + 2^{-66}. \tag{8}$$

Iterative sampler (M_2). The remaining calculation separates the intrinsic error of the exact rejection-free transition, truncation of the alternating sum, and approximation of the logarithm, so that every contribution is counted exactly once.

Intrinsic transition error. For each parameter set, let

$$\widetilde{B}_k = \sqrt{B_k^2 + \alpha_1^2 - 1}, \qquad \widetilde{\beta} = \frac{r}{\widetilde{B}_k}.$$

Lemma 4 gives the uniform one-step bound

$$\varepsilon_{\mathrm{id}} \le \frac{2\widetilde{\beta}\sqrt{2\pi}\exp(-\pi^2\widetilde{\beta}^2/2)}{\exp(-1/(2\widetilde{\beta}^2))(1 - \exp(-2\pi^2\widetilde{\beta}^2))}. \tag{9}$$

Truncation error. Orient the ray so that $\langle y, v\rangle \ge 0$, and set

$$\xi_k = \frac{\rho_g(y + kv)}{\rho_g(y)}, \qquad \rho = \exp\left(-\frac{\|v\|_2^2}{2r^2}\right).$$

Then $\xi_{k+1}/\xi_k \le \rho^{2k+1}$. For odd N, the two exact branch probabilities are

$$p_+ = \sum_{k=0}^{\infty}(-1)^k\xi_k, \qquad p_- = \sum_{k=1}^{\infty}(-1)^{k+1}\xi_k.$$

Alternation gives $p_+ \ge 1 - \rho$ and $p_- \ge \xi_1(1 - \rho^3)$, while the absolute remainder after index N is at most ξ_{N+1}. Consequently, the branchwise relative truncation error is bounded by

$$\varepsilon_{\mathrm{trunc}}(v, N) \le \max\left\{\frac{\rho^{(N+1)^2}}{1 - \rho}, \frac{\rho^{(N+1)^2-1}}{1 - \rho^3}\right\}. \tag{10}$$

Every shift contains the stretched first component, so $\|v\|_2 \geq \alpha_1$. The right-hand side of (10) is therefore at most its value at

$$\rho_* := \exp\left(-\frac{\alpha_1^2}{2r^2}\right).$$

We use the odd truncation indices $N = 75, 89, 55$ for the 120-, 180-, and 255-bit parameter sets, respectively.

Logarithm-approximation error. The certified implementation in Appendix A.2 satisfies $\delta_{\log} < 2^{-55}$. Since $\|v\|_2 \geq \alpha_1$, Theorem 3 gives

$$\varepsilon_{\log} \leq C_{\alpha_1,r}\big(e^{2^{-55}} - 1\big), \qquad C_{\alpha_1,r} = \frac{1+\rho_*}{1-\rho_*}. \tag{11}$$

Composition and Rényi loss. Combining the three one-step bounds gives

$$1 + \varepsilon_{\text{step}} \leq (1+\varepsilon_{\text{id}}) \cdot (1+\varepsilon_{\text{trunc}}) \cdot (1+\varepsilon_{\log}). \tag{12}$$

After τ steps,

$$\delta_\tau \leq (1+\varepsilon_{\text{step}})^\tau - 1.$$

For Rényi order $a = 2\lambda$, Lemma 3 then gives

$$M_2 := R_a(\rho_\tau \| \rho_g) \leq \left(1 + \frac{a(a-1)\delta_\tau^2}{2(1-\delta_\tau)^{a+1}}\right)^{1/(a-1)}. \tag{13}$$

Table 6 reports certified upper bounds obtained by direct substitution into (9)–(13).

Table 6. Concrete relative-error and Rényi-divergence accounting for the iterative sampler.

Security level	120	180	255
$\widetilde{\beta}$	2.5941	2.6098	2.6103
τ	58	80	128
N	75	89	55
ε_{id}	$< 2^{-44.10}$	$< 2^{-44.67}$	$< 2^{-44.69}$
$\varepsilon_{\text{trunc}}$	$< 2^{-47.15}$	$< 2^{-47.89}$	$< 2^{-52.81}$
$\varepsilon_{\log}$	$< 2^{-46.74}$	$< 2^{-46.28}$	$< 2^{-47.74}$
$\varepsilon_{\text{step}}$	$< 2^{-43.74}$	$< 2^{-44.15}$	$< 2^{-44.52}$
δ_τ	$< 2^{-37.88}$	$< 2^{-37.83}$	$< 2^{-37.52}$
$M_2 - 1$	$< 2^{-68.86}$	$< 2^{-68.17}$	$< 2^{-67.05}$
$2^{64}\log_2 M_2$ (bits)	< 0.050	< 0.080	< 0.174

In particular, every set satisfies $M_2 < 1 + 2^{-66}$. To express the effect directly in the security reduction, let $Q = BQ_S \leq 2^{64}$ be the total number of transcript samples in Theorem 5. The iterative-distribution factor costs at most $Q\log_2 M_2$ bits, as shown in the last row of the table. Using the discrete-Gaussian sampler

bound in (8) and the exponent $((a-1)/a)^2$ in Theorem 5, the loss in the first multiplicative term, relative to an ideal advantage $2^{-\lambda}$, is

$$L_{\text{total}} = \lambda\left(1-\left(\frac{a-1}{a}\right)^2\right) + Q\log_2 M_1 + Q\frac{a-1}{a}\log_2 M_2$$
$$< 1.41, \quad 1.44, \quad 1.54 \text{ bits}$$

for the 120-, 180-, and 255-bit parameter sets, respectively. Thus the concrete Rényi accounting remains below the stated three-bit budget for all three sets. The additive min-entropy and restart terms in the security theorem are separate from this distributional calculation.

A.4 Proof of Theorem 4

We first restate the theorem for convenience.

Theorem 4 (Correctness of stretching compression). *Let $s = (1, \mathbf{s}_1, \mathbf{s}_2)$ and $s^{(\alpha_1)} = (\alpha_1, \mathbf{s}_1, \mathbf{s}_2)$. Suppose that the deterministic map* CompressY *satisfies the shift-compatibility condition*

$$\mathsf{CompressY}(x + s^{(\alpha_1)}c') = \mathsf{CompressY}(x) + sc'$$

for every x in the relevant support and every signed challenge c' with $c' \equiv c \pmod 2$. If the sampler outputs $\widetilde{\mathbf{z}} = \mathbf{y} + s^{(\alpha_1)}c'$, the signature response is $\mathbf{z} = \mathsf{CompressY}(\widetilde{\mathbf{z}})$, and the commitment is computed as $\mathbf{w} = \mathbf{A}\mathsf{CompressY}(\mathbf{y}) \pmod{2q}$, then

$$\mathbf{Az} - qc\mathbf{j} = \mathbf{w} \pmod{2q}.$$

Moreover, for every $a > 1$, applying CompressY *cannot increase the Rényi divergence from the corresponding compressed target distribution.*

Proof. By the definition of $\widetilde{\mathbf{z}}$ and the shift compatibility of CompressY, we have

$$\begin{aligned}\mathbf{z} &= \mathsf{CompressY}(\widetilde{\mathbf{z}})\\ &= \mathsf{CompressY}(\mathbf{y} + s^{(\alpha_1)}c')\\ &= \mathsf{CompressY}(\mathbf{y}) + sc'.\end{aligned}$$

Using the defining relation

$$\mathbf{A}s = q\mathbf{j} \pmod{2q},$$

we obtain

$$\begin{aligned}\mathbf{Az} - qc\mathbf{j} &= \mathbf{A}\mathsf{CompressY}(\mathbf{y}) + \mathbf{A}sc' - qc\mathbf{j} \pmod{2q}\\ &= \mathbf{A}\mathsf{CompressY}(\mathbf{y}) + qc'\mathbf{j} - qc\mathbf{j} \pmod{2q}\\ &= \mathbf{A}\mathsf{CompressY}(\mathbf{y}) + q(c'-c)\mathbf{j} \pmod{2q}.\end{aligned}$$

Since $c' \equiv c \pmod 2$, the polynomial $c' - c$ has even coefficients. Hence

$$q(c' - c)\mathbf{j} = 0 \pmod{2q}.$$

Therefore,

$$\mathbf{Az} - qc\mathbf{j} = \mathbf{A}\mathsf{CompressY}(\mathbf{y}) = \mathbf{w} \pmod{2q}.$$

This proves correctness of the verification equation.

It remains to show that the compression step does not increase the distributional distance. Let P be the distribution of $\widetilde{\mathbf{z}}$, and let Q be the corresponding target distribution before compression. Let

$$P_{\mathrm{c}} = (\mathsf{CompressY})_{\#}P, \qquad Q_{\mathrm{c}} = (\mathsf{CompressY})_{\#}Q$$

be their pushforward distributions under CompressY. Since CompressY is deterministic, the data-processing inequality for Rényi divergence gives

$$R_a(P_{\mathrm{c}} \| Q_{\mathrm{c}}) \le R_a(P \| Q)$$

for every $a > 1$. Thus the stretching compression cannot increase the Rényi divergence.

A.5 Concrete Signing-Restart Probabilities

Encoding-capacity restart. The implementation places the merged rANS stream in a fixed-size byte slot. The response and hint coefficients have different marginals: the response coefficients are obtained by round-half-up compression of a discrete Gaussian, whereas the hints follow the bucket-crossing distribution induced by MakeHint. We weight the two distributions by their coefficient counts and quantize the resulting mixture to the rANS frequency table. From this model, we compute the mean μ_L and standard deviation σ_L of the payload length L, including the four-byte terminal-state flush. The reserved capacity is

$$R_p = \left\lceil \mu_L + \Phi^{-1}(1-p)\sigma_L \right\rceil,$$

where Φ is the standard-normal cumulative distribution function. Table 7 gives the capacities used for the reported signature sizes, with target $p = 2^{-30}$.

Table 7. Fixed-capacity rANS overflow estimates.

Security	Symbols	M_{voc}	μ_L	σ_L	$R_{2^{-30}}$	Signature size
120	1536	13	439.19	4.86	469	1175
180	2048	16	640.59	5.63	675	1589
255	2560	20	885.49	6.19	923	2013

The encoder also restarts if a symbol lies outside the configured vocabulary. Under the same ideal discrete-Gaussian and bucket-crossing models, a union bound over the merged stream puts this probability below 2^{-100}, 2^{-110}, and 2^{-117}, respectively. It is therefore a lower-order term relative to the 2^{-30} capacity-overflow target.

Signature-norm restart. We conservatively estimate the probability that the signer restarts because the compressed response exceeds the fixed bound B_s. The calculation below concerns the ideal target distribution; the final pointwise sampler error can be included multiplicatively as explained at the end of this subsection.

Write $d = n(l + m)$ for the number of coefficients in the uncompressed response components. The first ring component contains n additional coefficients and is compressed by α_1. For the corresponding continuous-Gaussian proxy, define

$$X = \sum_{j=1}^{d} G_j^2 + \frac{1}{\alpha_1^2} \sum_{j=d+1}^{d+n} G_j^2, \qquad G_j \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1).$$

Thus

$$\mu := \mathbb{E}[X] = d + \frac{n}{\alpha_1^2}, \qquad V := d + \frac{n}{\alpha_1^4}.$$

For completeness, the standard moment-generating-function argument for a weighted chi-square variable gives, for every $t > 0$,

$$\Pr\Big[X \geq \mu + 2\sqrt{Vt} + 2t\Big] \leq e^{-t}.$$

Indeed, for weights $a_j \in \{1, \alpha_1^{-2}\}$ and $0 < s < 1/2$,

$$\log \mathbb{E}\Big[e^{s(X-\mu)}\Big] = \frac{1}{2} \sum_j \big(-\log(1 - 2sa_j) - 2sa_j\big) \leq \frac{s^2 V}{1 - 2s},$$

and optimizing the resulting Chernoff bound yields the stated inequality.

The coefficient-wise round-half-up in $\mathsf{CompressY}$ contributes an error vector of Euclidean norm at most $\sqrt{n}/2$. Consequently,

$$\|\mathsf{CompressY}(\widetilde{\mathbf{z}})\|_2 \leq r\sqrt{X} + \frac{\sqrt{n}}{2}.$$

Therefore, with

$$T := \left(\frac{B_s - \sqrt{n}/2}{r}\right)^2, \qquad \Delta := T - \mu,$$

the event $\|\mathsf{CompressY}(\widetilde{\mathbf{z}})\|_2 > B_s$ implies $X > T$. When $\Delta > 0$, solving $\Delta = 2\sqrt{Vt} + 2t$ gives

$$t = \left(\frac{\sqrt{V + 2\Delta} - \sqrt{V}}{2}\right)^2,$$

and hence

$$\Pr[\|\mathsf{CompressY}(\widetilde{\mathbf{z}})\|_2 > B_s] \leq e^{-t} + \varepsilon_{\text{disc}}.$$

Here $\varepsilon_{\text{disc}}$ accounts for replacing the discrete Gaussian by its continuous proxy. Applying Poisson summation to the one-dimensional discrete-Gaussian moment-generating function shows that, for $r \geq 70$ and total dimension at most 2560, this correction is far below 2^{-1000} and is immaterial for the bounds below.

Table 8. Conservative bounds on the signature-norm restart probability.

Security	T	μ	t	Restart probability
120	2122.58	1284	86.51	$< 2^{-124.8}$
180	2754.97	1796	86.28	$< 2^{-124.4}$
255	3140.53	2305	56.61	$< 2^{-81.6}$

For the three concrete parameter sets, the following conservative bounds are obtained.

The signature-norm restart probabilities are therefore below $2^{-124.8}$, $2^{-124.4}$, and $2^{-81.6}$ for the 120-bit, 180-bit, and 255-bit parameter sets, respectively. Hence the reported parameter sets have overall signing-restart probability approximately 2^{-30}, dominated by the fixed-capacity encoding overflow.

Alternatively, enlarging the vocabulary radii to 15, 18, and 20 and setting the capacity-overflow target below one half of the corresponding signature-norm bound makes the signature-norm event the dominant restart source. The resulting capacities are 503, 715, and 950 bytes and the signature sizes are 1209, 1629, and 2040 bytes, at the cost of increasing the reported signatures by 34, 40, and 27 bytes, respectively.

Verification norm bounds. The compression analysis requires

$$B_v \geq B_s + \sqrt{nm}\left(\frac{\alpha_h}{4} + 1\right).$$

For the 120-bit, 180-bit, and 255-bit parameter sets, respectively, the right-hand side is

$$3979.70 < 4000 = B_v, \qquad 5226.52 < 5250 = B_v, \qquad 6612 < 6638 = B_v.$$

Hence, all three parameter sets satisfy the required verification ℓ_2-norm bound.

Finally, if the actual compressed sampler output distribution P satisfies the pointwise bound $\mathrm{err}_Q(P) \leq \delta_{\mathrm{samp}}$ with respect to the ideal compressed target distribution Q, then

$$\Pr_P[\|\mathbf{z}\|_2 > B_s] \leq (1 + \delta_{\mathrm{samp}})\left(e^{-t} + \varepsilon_{\mathrm{disc}}\right).$$

The sampler and implementation errors can therefore be incorporated without changing the displayed exponents at the reported precision.

A.6 CID

The identification scheme corresponding to our signature algorithm and its zero-knowledge simulator are shown in the figure.

CID scheme

$\mathbf{P}(\mathbf{A}, \mathbf{s})$		$\mathbf{V}(\mathbf{A})$
$\mathbf{y} \leftarrow \mathbf{SampleY}$		
$\mathbf{w} = \mathbf{A}\mathbf{y} \pmod{2q}$		
$w_0 \leftarrow LSB(\mathbf{w})$		
	$\xrightarrow{\mathbf{w}}$	$c \leftarrow \mathcal{U}(\mathcal{C})$
$\mathbf{z}_0 = \mathbf{y}$	$\xleftarrow{c}$	
for each non-zero c_i of c		
$\quad \mathbf{z}_i = R_{sc_i}(\mathbf{z}_{i-1})$		
$k = \|\mathbf{c}\|_1$		
$\quad$ **if** $\mathbf{z}_k = \perp$: **Abort**		
$\mathbf{z} = \mathbf{z}_k$		
$(\mathbf{z}_1, \mathbf{z}_2) \leftarrow \mathbf{z}$		
$\mathbf{h} \leftarrow \text{MakeHint}(\mathbf{z}_2, \mathbf{w})$		
$\sigma \leftarrow (\mathbf{z_1}, \mathbf{h})$	$\xrightarrow{\sigma}$	Reject if $\text{Verify}(\sigma, c) = 0$

CID simulator

1 : $\mathbf{z} \leftarrow \mathcal{D}^{\ell+m+1}_{\mathcal{R},t,r}$
2 : $c \leftarrow \mathcal{U}(\mathcal{C})$
3 : $\mathbf{w} = \mathbf{A}\mathbf{z} - qc$
$\pmod{2q}$,$w_0 \leftarrow LSB(\mathbf{w})$
4 : $\mathbf{z} = \mathbf{z}_k$,
$(\mathbf{z}_1, \mathbf{z}_2) \leftarrow \mathbf{z}$,$\mathbf{h} \leftarrow \text{MakeHint}(\mathbf{z}_2, \mathbf{w})$
5 : if $\|\mathbf{z}\|_2 > B$, **return**
$(\perp, c, (\mathbf{w}, w_0))$
6 : **return** $((\mathbf{z_1}, \mathbf{h}), c, (\mathbf{w}, w_0))$

Fig. 4. Canonical identification scheme and simulator corresponding to the unencoded version of our signature scheme.

A.7 Conservative Estimate of the Commitment Min-Entropy

We give a conservative lower bound rather than assuming that the commitment is uniform. The argument follows the standard reduction modulo $2\alpha_1$: the parity of a rounded discrete Gaussian is close to uniform because a wide discrete Gaussian is close to uniform modulo $2\alpha_1$.

Lemma 7 (Min-entropy of rounded discrete-Gaussian low bits). *Let $A \in \mathbb{Z}_{>0}$ and let $X_1, \ldots, X_n$ be independent samples from the discrete Gaussian on $\mathbb{Z}$ with standard deviation r, i.e.,*

$$\Pr[X_i = x] = \frac{\exp(-x^2/(2r^2))}{\sum_{z\in\mathbb{Z}} \exp(-z^2/(2r^2))}.$$

Define $Y_i = \mathrm{RoundHU}(X_i/A)$, $B_i = Y_i \bmod 2$, and $B = (B_1, \ldots, B_n)$. Set

$$\Lambda_{A,r} = \frac{\pi^2 r^2}{2A^2}, \qquad \varepsilon_{A,r} = \frac{2e^{-\Lambda_{A,r}}}{1 - e^{-3\Lambda_{A,r}}}.$$

Then

$$H_\infty(B) \geq n\big(1 - \log_2(1 + 2\varepsilon_{A,r})\big).$$

Proof. For one coordinate, let $R = X_i \bmod 2A$. Round-half-up is invariant under integer translation, so replacing X_i by $X_i + 2A$ leaves B_i unchanged, whereas replacing it by $X_i + A$ flips B_i. Thus exactly A of the $2A$ residue classes give each bit value.

Poisson summation applied to the discrete Gaussian modulo $2A$ gives

$$\Delta_{\mathrm{TV}}(R, U_{2A}) \leq 2\sum_{m=1}^{\infty} \exp(-\Lambda_{A,r} m^2),$$

where U_{2A} is uniform on $\mathbb{Z}/(2A)\mathbb{Z}$. Since $m^2 \geq 1 + 3(m-1)$ for $m \geq 1$, the right-hand side is at most $\varepsilon_{A,r}$. Therefore

$$\max_{b\in\{0,1\}} \Pr[B_i = b] \leq \frac{1}{2} + \varepsilon_{A,r}.$$

The B_i are independent, and hence

$$\max_{b\in\{0,1\}^n} \Pr[B = b] \leq (1/2 + \varepsilon_{A,r})^n.$$

Taking the negative base-two logarithm proves the claim.

We also record the transfer step needed for the implemented sampler.

Lemma 8 (Min-entropy transfer from rooted Rényi divergence). *Let $a > 1$ and let P, Q be distributions on the same discrete domain. If $R_a(P\|Q) \leq M$ and $H_\infty(Q) \geq \alpha_0$, then*

$$H_\infty(P) \geq \frac{a-1}{a}\big(\alpha_0 - \log_2 M\big).$$

Proof. For every outcome x, the corresponding nonnegative summand in the definition of rooted Rényi divergence satisfies

$$P(x)^a Q(x)^{1-a} \le M^{a-1}.$$

Since $Q(x) \le 2^{-\alpha_0}$, it follows that

$$P(x) \le \left(M 2^{-\alpha_0}\right)^{(a-1)/a}.$$

Taking the maximum over x proves the stated min-entropy bound.

We now apply these lemmas to the compressed commitment. Let $\mathbf{y}'_0 = \mathrm{RoundHU}(\mathbf{y}_0/\alpha_1)$ be the first component of $\mathsf{CompressY}(\mathbf{y})$. In the matrix

$$\mathbf{A} = [\, q\mathbf{j} - 2b \mid 2\mathbf{A}_{\mathbf{gen}} \mid 2\mathbf{I}_m \,],$$

all columns other than the first are even and q is odd. Consequently,

$$\mathsf{LSB}(\mathbf{w}) = \mathsf{LSB}(\mathbf{y}'_0)\mathbf{j}.$$

The low-bit vector is part of the commitment, so

$$H_\infty(\mathbf{w}_h, \mathbf{w}_0) \ge H_\infty(\mathbf{w}_0) = H_\infty(\mathsf{LSB}(\mathbf{y}'_0)).$$

For an ideal discrete-Gaussian mask, Lemma 7 applies with $n = 256$, $A = \alpha_1$, and the masking width r. For the implemented sampler, let P be the distribution of $\mathsf{LSB}(\mathbf{y}'_0)$ and Q its ideal counterpart. By data processing, the sampler bound in Table 5 also bounds $R_{2\lambda}(P\|Q)$ and, in particular, satisfies $M_1 \le 1 + 2^{-66}$. Applying Lemma 8 gives the following conservative lower bounds. All displayed decimal values are rounded in the direction that weakens the resulting min-entropy bound.

Table 9. Conservative lower bounds on the commitment min-entropy.

Security level	120	180	255
r	70	82	99
α_1	8	8	16
$a = 2\lambda$	240	360	510
$\Lambda_{\alpha_1,r}$	377.82	518.46	188.92
$\varepsilon_{\alpha_1,r}$	$< 2^{-544}$	$< 2^{-746}$	$< 2^{-271}$
Ideal $H_\infty(\mathbf{w}_0)$	$> 256 - 2^{-534}$	$> 256 - 2^{-737}$	$> 256 - 2^{-262}$
Implemented lower bound	> 254.93	> 255.28	> 255.49

For the simulator, conditioning on a fixed challenge c changes the relevant low-bit vector from $\mathsf{LSB}(\mathbf{z}_0)$ to $\mathsf{LSB}(\mathbf{z}_0 - c)$, which is a permutation of its bit strings. Hence the same lower bound applies to the simulator commitment. Thus, in the worst case, both the honest and simulated commitments have min-entropy strictly greater than 254.93 bits under this conservative estimate. The explicit value in Table 9, rather than an assumption of uniformity, is the value to be substituted for α in Theorems 5 and 6.

A.8 Divergence and Computational Uniqueness of the CID

Divergence, verifiability, and min-entropy of the HVZK simulator. By the data-processing inequality for Rényi divergence, Table 5 gives $M_1 \leq 1+2^{-66}$, while Table 6 gives the sharper concrete bounds on $M_2 < 1 + 2^{-66}$. Verifiability follows directly, and the min-entropy bound follows from the preceding argument.
CUR. The adversary $\mathcal{A}$ would instead produce $w_0, c, z_{a,1}, h_a, z_{b,1}, h_b$, with $(z_{a,1}, h_a) \neq (z_{b,1}, h_b)$. We show that this guarantees that the reconstructed vectors z'_a and z'_b differ. Since the reconstructed vector z' is given by

$$z' = \left[z_1^T,\ z_2'^T\right]^T,$$

the equality $z'_a = z'_b$ implies that $z_{a,1} = z_{b,1}$. Additionally, given $z_{a,1} = z_{b,1}$, in order for $z_{a,2} = z_{b,2}$ to hold, it is required that

$$w_{0,a} \equiv w_{0,b} \pmod{2(q-1)},$$

which is only possible if

$$h_a \equiv h_b \pmod{2(q-1)}.$$

By definition of the signature data structure, every decoded coefficient of h is the unique representative in $[0, 2(q-1)) \subset [0, 2q)$; an out-of-range encoding is not reduced modulo $2(q-1)$, but fails to decode. Consequently, the above congruence implies $h_a = h_b$. It follows that two distinct accepting transcripts for the compressed CID scheme imply two distinct reconstructed vectors z'_a and z'_b, both of which have norm at most B_v. This directly yields a solution to the MSIS variant with bound $2B_v$.

B Usage of AI

In accordance with the ACM Policy on the Use of AI Tools, we disclose the use of generative AI systems during the preparation of this work.

We used the following AI tools during the development of the code and implementation aspects of this paper:

- GPT-5.4
- GPT-5.6 sol
- Opus 4.7
- Gemini 3.1 Pro

These tools were primarily used to assist with programming-related tasks, including:

- generation of initial code templates and boilerplate implementations,
- debugging and correction of implementation errors,
- refactoring and optimization of existing code,
- translation of algorithmic descriptions into executable code,

– assistance in writing and reviewing auxiliary scripts for experiments.

All AI-generated suggestions were manually reviewed, tested, and potentially modified by the authors before inclusion in the final implementation. The authors take full responsibility for the correctness, integrity, and appropriateness of all submitted material, including any content assisted by AI tools.