Phoenix: Hash-and-Sign with Aborts from Lattice Gadgets

Corentin Jeudy[1,2], Adeline Roux-Langlois[3], and Olivier Sanders[1]

corentin.jeudy@orange.com, adeline.roux-langlois@cnrs.fr, olivier.sanders@orange.com

[1] Orange Labs, Applied Crypto Group, Cesson-Sévigné, France
[2] Univ Rennes, CNRS, IRISA, Rennes, France
[3] Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France

Abstract. Preimage sampling is a fundamental tool in lattice-based cryptography, and its performance directly impacts that of the cryptographic mechanisms relying on it. In 2012, Micciancio and Peikert proposed a new way of generating trapdoors (and an associated preimage sampling procedure) with very interesting features. Unfortunately, in some applications such as digital signatures, the performance may not be as competitive as other approaches like Fiat-Shamir with Aborts. In an effort to improve preimage sampling for Micciancio-Peikert (MP) trapdoors, Lyubashevsky and Wichs (LW) introduced a new sampler which leverages rejection sampling but suffers from strong parameter requirements that hampered performance. As a consequence it seemed to be restricted to theoretical applications and has not been, to our knowledge, considered for real-world applications.

Our first contribution is to revisit the LW sampler by proposing an improved analysis which yields much more compact parameters. This leads to gains on the preimage size of about 60% over the LW sampler, and up to 25% compared to the original MP sampling technique. It thus sheds a new light on the LW sampler, opening promising perspectives for the efficiency of advanced lattice-based constructions relying on such mechanisms. To provide further improvements, we show that it perfectly combines with the approximate trapdoors approach by Chen, Genise and Mukherjee, but with a smaller preimage error.

Building upon those results, we introduce a hash-and-sign signature scheme called Phoenix. The scheme is based on the M-LWE and M-SIS assumptions and features attractive public key and signature sizes which are even smaller than those of the most recent gadget-based construction Eagle of Yu, Jia and Wang (Crypto'23). Moreover, Phoenix is designed to be implementation-friendly, avoiding in particular complex Gaussian samplers that are often hard to protect.

Keywords: Lattice-Based Cryptography · Trapdoors · Preimage Sampling · Signature

1 Introduction

Lattice-based cryptography has proven to be a relatively stable and extensively studied candidate to provide post-quantum secure primitives, and has now shifted towards proposing concretely efficient constructions. The NIST standardization [NISa] perfectly reflects this trend as they recently released the first round of standards, which is dominated by lattice schemes [BDK+18,DKL+18,PFH+20], and are moving to practical deployment discussions. Although they provide a first set of solutions for initiating the post-quantum transition, NIST recently called for additional digital signatures [NISb]. The lattice-based candidates to this new competition, along with some recent publications, e.g., [YJW23], show that there is still room for improvement in this area in terms of optimizing bandwidth, ease of implementation, side-channel protection, etc.

If we set aside schemes designed with very specific applications in mind, e.g., [LNP22a,BLNS23,JRS23], lattice-based signature schemes usually follow one of two main paradigms. The first one, called the *hash-and-sign* paradigm, was instantiated by Gentry, Peikert and Vaikuntanathan [GPV08] (later abbreviated GPV) with lattice preimage sampleable trapdoor functions. In such schemes, the signing key consists of a trapdoor for a publicly computable function which allows one to efficiently find short preimages. Signatures are then preimages of seemingly random (and possibly message-dependent) syndromes. Only the signer is able to compute such preimages, but everyone is able to compute the image to ensure they represent valid signatures. Several schemes rely on variants of the above, e.g., [GPV08,MP12,DM14,DLP14], and were successfully pushed towards concrete practicality [PFH+20,EFG+22,YJW23] using an additional assumption. Trapdoor preimage sampleable functions also represent the most widely used building block in the design of more advanced forms of signatures such as group signatures [dPLS18,LNPS21], blind signatures [AKSY22,dPK22,BLNS23], signatures with efficient protocols [LLM+16,JRS23], etc. In their general use, trapdoor preimage sampling can however be quite computationally intensive, and most solutions are designed to only support Gaussian-distributed preimages.

An alternative, called the *Fiat-Shamir with Aborts* (FSwA) paradigm, was proposed by Lyubashevsky [Lyu12], building signatures on Schnorr-like proofs made non-interactive with the Fiat-Shamir transform. This framework avoids the use of trapdoors, and uses rejection sampling to control the distribution of signatures while making them independent of the signing key. Even though most applications yield Gaussian-distributed signatures, it is possible to tweak the rejection sampling step to get other distributions that can be more suitable depending on the context. Efficient instantiations of this signature paradigm were proposed, such as qTESLA [ABB+20] and Dilithium [DKL+18].

Interestingly, in [LW15], Lyubashevsky and Wichs show that these two approaches may be combined in the case of Micciancio-Peikert trapdoors [MP12].

1.1 Micciancio-Peikert Sampler

In [MP12], Micciancio and Peikert propose a preimage sampling algorithm (later called MP sampler) for matrices $\mathbf{A_H} = [\mathbf{A}|\mathbf{HG} - \mathbf{AR}]$, where $\mathbf{R}$ constitutes the trapdoor. More precisely, $\mathbf{A}$ is a uniform matrix in $R_q^{d\times 2d}$, $\mathbf{H}$ is a tag matrix in $GL_d(R_q)$, $\mathbf{G} \in R^{d\times kd}$ (with $k = \log_b q$) is the base-b gadget matrix, and $\mathbf{R}$ is a short matrix over the ring R, e.g., power-of-two cyclotomic ring. Their algorithm uses the knowledge of $\mathbf{R}$ to sample $\mathbf{v} \in R^{(2+k)d}$ according to a spherical discrete Gaussian of parameter s such that $\mathbf{A_H v} = \mathbf{u} \bmod q$ for an input syndrome $\mathbf{u}$. The technique first relies on the observation that if $\mathbf{z}$ is a Gaussian with width $s_{\mathbf{G}}$ such that $\mathbf{HGz} = \mathbf{u}$, then the vector $\mathbf{v'} = [(\mathbf{Rz})^T|\mathbf{z}^T]^T$ is a valid candidate. This naive approach leaks information on the trapdoor $\mathbf{R}$, which is why the authors perturb this solution $\mathbf{v'}$ into $\mathbf{v} = \mathbf{p} + \mathbf{v'}$, for some suitable perturbation vector $\mathbf{p}$, while adjusting $\mathbf{z}$ to verify $\mathbf{HGz} = \mathbf{u} - \mathbf{A_H p}$. By carefully choosing the covariance of the Gaussian $\mathbf{p}$, one can indeed ensure that $\mathbf{v}$ follows a spherical Gaussian distribution of width s, which in turn does not leak information on the trapdoor.

Although the approach above perfectly fulfils the security expectations of preimage sampling, it remains unsatisfactory in a number of aspects. First, the information on $\mathbf{R}$ in $\mathbf{v'} = [(\mathbf{Rz})^T|\mathbf{z}^T]^T$ that needs to be hidden only affects the first component. One would expect to only have to perturb the first part to ensure security. Additionally, the sampler is quite rigid as it requires sampling perturbations $\mathbf{p}$ from highly non-spherical Gaussian, and is limited to Gaussian preimages.

1.2 Lyubashevsky-Wichs Sampler

To address these problems, Lyubashevsky and Wichs [LW15] break the symmetry between $\mathbf{v}_1 = \mathbf{p}_1 + \mathbf{Rz}$ and $\mathbf{v}_2 = \mathbf{p}_2 + \mathbf{z}$ by setting $\mathbf{p} = [\mathbf{p}_1^T|\mathbf{0}]^T$ and $\mathbf{z} = \mathbf{G}^{-1}(\mathbf{u} - \mathbf{Ap}_1)$ where $\mathbf{G}^{-1}(\cdot)$ is the base-b decomposition. Directly outputting $\mathbf{v}_1 = \mathbf{p}_1 + \mathbf{Rz}$ and $\mathbf{v}_2 = \mathbf{z}$ again leaks information on $\mathbf{R}$ because of $\mathbf{v}_1$ and they thus need to adjust this approach. By identifying $\mathbf{Ap}_1$, $\mathbf{z}$ and $\mathbf{v}_1$ with (respectively) the commitment, the challenge and the response of a zero-knowledge proof of knowledge of $\mathbf{R}$, this problem is very similar to the one of Fiat-Shamir signatures in [Lyu12]. They then resort to the same workaround, namely rejection sampling: before outputting $\mathbf{v}_1 = \mathbf{p}_1 + \mathbf{Rz}$ and $\mathbf{v}_2 = \mathbf{z}$, one performs rejection sampling on $\mathbf{v}_1$ to make its distribution independent of $\mathbf{R}$ and $\mathbf{z}$. We later refer to this sampling method as the LW sampler.

However, to thoroughly show that the preimages do not leak information on $\mathbf{R}$, they provide a simulation result which suffers from parameter constraints that make it less efficient than the MP sampler in terms of preimage size. More concretely, they show that the output distribution of the preimages is statistically close to a distribution that does not depend on the trapdoor $\mathbf{R}$ for an arbitrary (potentially adversarial) syndrome $\mathbf{u}$. Because they deal with an arbitrary $\mathbf{u}$, nothing can be assumed about its distribution which in turn places strong restrictions on the parameters to compensate. Indeed, in their result, they need to

assume that $\mathbf{Av}_1$ (and $\mathbf{Ap}_1$) is *statistically* close to uniform requiring the parameters to be large enough to use a regularity lemma. This requirement in turn prevents them from using a computational instantiation of MP trapdoors. Since computational MP trapdoors lead to much smaller preimages, they are usually more compact than the ones generated by the LW sampler. Concretely, the size of a GPV signature [GPV08] with the LW sampler are about $80 - 90\%$ larger than the ones using the MP sampler, as described in Section 4. This looks like a paradox as one would intuitively expect the method from [LW15] to combine the best of trapdoor-based signatures and Fiat-Shamir with aborts signatures.

1.3 Our Contributions

The goal of our paper is to revisit the LW approach [LW15] so as to achieve its full potential. Our first result is a reassessment of the original security analysis showing that we can significantly alleviate the requirements identified in [LW15], at least in the most common applications of preimage sampling. It entails important gains in performance of around 60%, resulting in shorter preimages than the one obtained with the original MP method [MP12] by 25%, thus solving the apparent paradox mentioned above.

In a second step, we leverage the works on approximate trapdoors initiated by Chen, Genise and Mukherjee [CGM19] to further reduce the size of the preimages. Our approach allows to reduce the sampling error, thus yielding either higher security guarantees or better compactness.

Finally, we illustrate the potential of the sampler by designing a hash-and-sign signature scheme, which we call Phoenix. The latter showcases interesting features including small keys and signatures, but also an implementation-friendly design that in particular supports a variety of signature distributions.

We now give more details on these contributions. For the sake of genericity, the contributions in this paper are described over structured lattices but we note that they also apply to standard ones.

Contribution 1: Re-assessing the Lyubashevsky-Wichs Sampler. Our first contribution is a more specific analysis of the LW sampler to get rid of the restrictive requirements mentioned above and thus obtain more compact preimages. Intuitively, our new analysis stems from the observation that the initial assumption of [LW15], namely the fact that the syndrome can be fully controlled by the adversary, is too strong in general. Indeed, in many common situations, the syndrome follows a prescribed distribution, which can be leveraged to simulate preimages in the proof.

For GPV signatures [GPV08] for example, the syndrome $\mathbf{u}$ is the hash output of the message $\mathcal{H}(\mathbf{m})$ where $\mathcal{H}$ is modelled as a random oracle. This means that the syndromes we expect are uniformly distributed and cannot be controlled by the adversary. This allows us to remove this constraint on $\mathbf{Av}_1$ being statistically close to uniform, as we can, at a high level, use the randomness of $\mathbf{u}$ to achieve the same conclusion. As we show in our paper, removing this constraint removes

the need for a large perturbation (either in norm or dimension) and thus leads to improved performances. In the meantime, our result avoids placing restrictions on the underlying algebraic ring R nor the working modulus q, making it suitable for a larger variety of settings and applications.

Compared to the original Micciancio-Peikert sampler, the size of $\mathbf{v}_1$ increases, but $\mathbf{v}_2$ is now in base b which is much smaller (even minimal when $b = 2$ for example). Concretely, the total bit-size of $\mathbf{v}$ for a GPV signature built upon our improved simulation result is reduced by respectively 60% compared to the original[4] LW sampler, and by 25% compared to the MP sampling method. The estimates are detailed in Tables 4.1, 4.2 and 4.3. Along with these estimates, we also analyze the impact of the gadget base b. We show that the intuition of increasing b to reduce the signature size, that was true for the MP sampler (as well as the original LW sampler), should be re-assessed when the sampler changes. More precisely, we explain why the MP sampler and the previous version of the LW sampler perform better with higher bases, and why our new analysis and parameter constraints show that the base leading to the smallest signatures is $b = 2$.

Contribution 2: Leveraging Approximate Trapdoors. At this stage, we have shown that the revisited LW sampler can outperform the MP one but the resulting signature size is still far from competitive compared to, e.g., the future NIST standard Dilithium [DKL+18]. To fully reinstate LW samplers, we thus need to find other means of reducing this size.

As the LW approach inherently leads to signatures where most elements are very small (since $\|\mathbf{v}_2\|_\infty < b$), the remaining target to improve performance is essentially the dimension of those signatures. Thanks to our new analysis above, we have already managed to reduce the one of $\mathbf{A}$, and hence of $\mathbf{v}_1$. When it comes to $\mathbf{v}_2$, the situation is more complex as the dimension seems to be dictated by the one of the gadget matrix $\mathbf{G}$. Fortunately, a study initiated by Chen, Genise and Mukherjee [CGM19] improved the performance of gadget-based constructions through the notion of approximate trapdoors. The idea is to drop the low-order gadget entries and only consider a partial gadget $\mathbf{G}_H = \mathbf{I}_d \otimes \mathbf{g}_H^T$ with $\mathbf{g}_H = [b^\ell | \dots | b^{k-1}]^T$. It not only reduces the dimension of $\mathbf{v}_2$ (and hence the number of elements in the signature), but it also reduces the public and secret key sizes. Additionally, having a secret key $\mathbf{R}$ with fewer columns allows us to reduce $\|\mathbf{Rz}\|_2$ which defines the quality of our sampler, thus reducing the size of $\mathbf{v}_1$ as well.

The removed low-order entries however introduce an error on the preimage which must be taken into account in the security assessment. Intuitively, the more entries are dropped, the larger the error, and in turn the less secure it gets. Reducing the error is thus critical as it leads to better security, or enables to drop more entries to further improve performance. In this regard, we note that our revisited LW sampler lends itself well to approximate trapdoors since

[4] By "original", we mean the LW sampler with the parameters resulting from the original analysis in [LW15].

$\mathbf{v}_2$ is binary and not gaussian. This leads to a sampling error that is smaller than the one from [CGM19] and (almost) as small as that of the recent gadget construction of Yu, Jia and Wang [YJW23].

Contribution 3: Phoenix, a New Hash-and-Sign Scheme. Plugging the previous contributions in the GPV framework leads to a new hash-and-sign signature scheme, which we call Phoenix, which allows to assess the benefits of the LW sampler for concrete applications. The performances of Phoenix are summarized in Table 1.1.

One of the most surprising features of Phoenix is arguably its relatively small signatures sizes |sig|. Given the initial performance of the LW sampler, this was clearly unexpected. An interesting byproduct of having an extremely short $\mathbf{v}_2$ is also that we can apply public key compression as is done in e.g. Dilithium [DKL+18]. This cuts the public key size |pk| in half at almost no cost on the security, allowing us to reach smaller public keys than [YJW23] as well. We give a detailed comparison with the other M-LWE-based signatures Dilithium [DKL+18], Haetae [CCD+23], Raccoon [dPEK+], and Eagle [YJW23] in Section 6.4 and Table 6.2.

NIST-II			NIST-III			NIST-V		
|sk|	|pk|	|sig|	|sk|	|pk|	|sig|	|sk|	|pk|	|sig|
512	1184	2190	648	1490	2897	972	2219	4468

Table 1.1. Performance in bytes of Phoenix for NIST-II, NIST-III and NIST-V security.

Finally, the scheme also benefits from interesting features due to the nature of the LW sampler. The latter can be instantiated with a variety of distributions that are more suited for easy and secure implementations. In particular, Phoenix only involves spherical Gaussians over R which removes the need for complex Gaussian samplers as in previous hash-and-sign schemes (FFO sampler for [PFH+20], hybrid sampler for [EFG+22], perturbation samplers for [CGM19,YJW23]). This makes Phoenix easier to protect against side-channel attacks. We also provide a version of Phoenix which uses uniform distributions over hypercubes to avoid floating points altogether, as described in Appendix B. Our scheme thus combines the benefits of Fiat-Shamir with Aborts schemes and of hash-and-sign schemes, as was originally expected from the LW sampler. This work shows that said sampler is not only of theoretical interest but may have concrete applications that could benefit from its nice performance and implementation features.

1.4 Organization

We start by recalling some notations and standard notions in Section 2. Then, we provide our new preimage sampling analysis in Section 3, and discuss its

performance with respect to the gadget base b in Section 4. We provide in Section 5 an approximate version of the sampler and propose Phoenix as a concrete hash-and-sign signature based on the latter in Section 6.

2 Preliminaries

In this paper, for two integers $a \leq b$, we define $[a, b] = \{k \in \mathbb{Z} : a \leq k \leq b\}$. When $a = 1$, we simply use $[b]$ instead of $[1, b]$. Further, q is a positive integer, and we define $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. We may identify the latter with the set of representatives $(-q/2, q/2] \cap \mathbb{Z} = [-\lfloor (q-1)/2 \rfloor, \lceil (q-1)/2 \rceil]$. Vectors are written in bold lowercase letters $\mathbf{a}$ and matrices in bold uppercase letters $\mathbf{A}$. The transpose of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^T$. The identity matrix of dimension d is denoted by $\mathbf{I}_d$. We use $\|\cdot\|_p$ to denote the ℓ_p norm of $\mathbb{R}^d$, i.e., $\|\mathbf{a}\|_p = (\sum_{i \in [d]} |a_i|^p)^{1/p}$ for any positive integer p, and $\|\mathbf{a}\|_\infty = \max_{i \in [d]} |a_i|$. We also define the spectral norm of a matrix $\mathbf{A}$ by $\|\mathbf{A}\|_2 = \max_{\mathbf{x} \neq \mathbf{0}} \|\mathbf{A}\mathbf{x}\|_2 / \|\mathbf{x}\|_2$.

2.1 Lattices

A full-rank *lattice* $\mathcal{L}$ of rank d is a discrete subgroup of $(\mathbb{R}^d, +)$. The *dual lattice* of $\mathcal{L}$ is defined by $\mathcal{L}^* = \{\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(\mathcal{L}) : \forall \mathbf{y} \in \mathcal{L}, \mathbf{x}^T\mathbf{y} \in \mathbb{Z}\}$. For d, m, q positive integers, we consider the family of q-ary lattices $\{\mathcal{L}_q^\perp(\mathbf{A}); \mathbf{A} \in \mathbb{Z}_q^{d \times m}\}$, where $\mathcal{L}_q^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod q\mathbb{Z}\}$. For any $\mathbf{A} \in \mathbb{Z}_q^{d \times m}$ and $\mathbf{u} \in \mathbb{Z}_q^d$, we define $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{u} \bmod q\mathbb{Z}\}$ which is a coset of $\mathcal{L}_q^\perp(\mathbf{A})$.

2.2 Probabilities

For a finite set S, we define $U(S)$ to be the uniform probability distribution over S. We use $x \hookleftarrow P$ to describe the action of sampling $x \in S$ according to the probability distribution P. In contrast, we use $x \sim P$ to mean that the random variable x follows P. The *statistical distance* between two discrete distributions P, Q over a countable set S is defined as $\Delta(P, Q) = \frac{1}{2} \sum_{x \in S} |P(x) - Q(x)|$. Later, $\mathscr{D}_s, \mathscr{D}_t$ denote arbitrary distributions called source and target distributions respectively. Let P, Q be two discrete distributions such that the support of P, denoted by S is a subset of that of Q. The Rényi divergence of order $a \in (1, +\infty]$ from P to Q is defined by

$$RD_a(P\|Q) = \left(\sum_{\mathbf{x} \in S} \frac{P(\mathbf{x})^a}{Q(\mathbf{x})^{a-1}} \right)^{\frac{1}{a-1}}.$$

We also use the smooth Rényi divergence from P to Q, parameterized by $\varepsilon \geq 0$, defined in [DFPS22] as

$$RD_\infty^\varepsilon(P\|Q) = \inf\{M > 0 : \mathbb{P}_{\mathbf{x} \sim P}[M \cdot Q(\mathbf{x}) \geq P(\mathbf{x})] \geq 1 - \varepsilon\}.$$

For a center $\mathbf{c} \in \mathbb{R}^d$ and positive definite $\mathbf{S} \in \mathbb{R}^{d \times d}$, we define the Gaussian function $\rho_{\sqrt{\mathbf{S}}, \mathbf{c}} : \mathbf{x} \in \mathbb{R}^d \mapsto \exp(-\pi(\mathbf{x} - \mathbf{c})^T \mathbf{S}^{-1} (\mathbf{x} - \mathbf{c}))$. For a countable set

$A \subseteq \mathbb{R}^d$, we define the *discrete Gaussian distribution* $\mathcal{D}_{A,\sqrt{\mathbf{S}},\mathbf{c}}$ of support A, covariance $\mathbf{S}$ and center $\mathbf{c}$ by its density $\mathcal{D}_{A,\sqrt{\mathbf{S}},\mathbf{c}} : \mathbf{x} \in A \mapsto \rho_{\sqrt{\mathbf{S}},\mathbf{c}}(\mathbf{x})/\rho_{\sqrt{\mathbf{S}},\mathbf{c}}(A)$, where $\rho_{\sqrt{\mathbf{S}},\mathbf{c}}(A) = \sum_{\mathbf{x}\in A} \rho_{\sqrt{\mathbf{S}},\mathbf{c}}(\mathbf{x})$. When $\mathbf{c} = \mathbf{0}$, we omit it from the notations. When $\mathbf{S} = s^2\mathbf{I}_d$, we use s as subscript instead of $\sqrt{\mathbf{S}}$. As coined by Micciancio and Regev [MR07], we define the *smoothing parameter* of a lattice $\mathcal{L}$, parameterized by $\varepsilon > 0$, by $\eta_\varepsilon(\mathcal{L}) = \inf\{s > 0 : \rho_{1/s}(\mathcal{L}^*) \leq 1 + \varepsilon\}$.

We also give the standard tail bounds for the discrete Gaussian distribution from [Ban93,Pei08]. Notice that when $\mathbf{c} = \mathbf{0}$, the smoothing requirement $s \geq \eta_\varepsilon(\mathcal{L})$ in the following is not needed.

Lemma 2.1 ([Ban93, Lem. 1.5][Pei08, Cor. 5.3]). *Let $\mathcal{L} \subset \mathbb{R}^d$ be a lattice of rank d, and $s > 0$. It holds that for $c \geq 1$, $\mathbb{P}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{L},s}}[\|\mathbf{x}\|_2 > cs\sqrt{d/2\pi}] < (c\sqrt{e}e^{-c^2/2})^d$. Additionally, for $t \geq 0$, we have $\mathbb{P}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{L},s}}[\|\mathbf{x}\|_\infty > ts/\sqrt{2\pi}] < 2de^{-t^2/2}$.*

Finally, we give the rejection sampling results from [DFPS22, Lem. 2.2, Lem. 4.1], which we slightly specify to our context.

Lemma 2.2 (Adapted from [DFPS22, Lem. 2.2, Lem. 4.1]). *Let d, m be positive integers. Let $\mathscr{D}_s, \mathscr{D}_t, \mathscr{D}_r, \mathscr{D}_z$ be distributions on $\mathbb{R}^d, \mathbb{R}^d, \mathbb{R}^{d\times m}, \mathbb{R}^m$ respectively. Let $\mathbf{R}$ be drawn from $\mathscr{D}_r$. Then, let $Y \subseteq \mathbb{R}^d$ be the support of the distribution of $\mathbf{R} \cdot \mathscr{D}_z$. We assume they are such that $Supp(\mathscr{D}_t) \subseteq Supp(\mathscr{D}_s^{+\mathbf{Rz}})$ for all $\mathbf{Rz} \in Y$, where $\mathscr{D}_s^{+\mathbf{Rz}}$ is the distribution corresponding to sampling $\mathbf{p}$ from $\mathscr{D}_s$ and outputting $\mathbf{p} + \mathbf{Rz}$. Let $M > 1$ and $\varepsilon \in [0, 1/2]$ such that $\max_{\mathbf{Rz}\in Y} RD^\varepsilon_\infty(\mathscr{D}_t \| \mathscr{D}_s^{+\mathbf{Rz}}) \leq M$. We then define two distributions*

$\mathcal{P}_1$: *Sample $\mathbf{z} \hookleftarrow \mathscr{D}_z$, $\mathbf{p} \hookleftarrow \mathscr{D}_s$ and set $\mathbf{v} \leftarrow \mathbf{p} + \mathbf{Rz}$. Then sample a continuous $u \hookleftarrow U([0, 1])$. If $u > \min(1, \mathscr{D}_t(\mathbf{v})/(M \cdot \mathscr{D}_s(\mathbf{p})))$, restart, otherwise output $(\mathbf{v}, \mathbf{z})$.*

$\mathcal{P}_2$: *Sample $\mathbf{z} \hookleftarrow \mathscr{D}_z$, $\mathbf{v} \hookleftarrow \mathscr{D}_t$. Then sample a continuous $u \hookleftarrow U([0, 1])$. If $u > 1/M$, restart, otherwise output $(\mathbf{v}, \mathbf{z})$.*

Then, $\Delta(\mathcal{P}_1, \mathcal{P}_2) \leq \varepsilon$ and for all $a \in (1, +\infty]$, $RD_a(\mathcal{P}_1 \| \mathcal{P}_2) \leq 1/(1 - \varepsilon)^{a/(a-1)}$.

To perform rejection sampling in the Gaussian case, we use the following bound on the smooth Rényi divergence between shifted Gaussians.

Lemma 2.3 ([DFPS22, Lem. C.2]). *Let d be a positive integer, $\mathbf{y}$ in $\mathbb{R}^d$, $\varepsilon \in (0, 1)$, and $s > 0$. Then, $RD^\varepsilon_\infty(\mathcal{D}_{\mathbb{Z}^d,s} \| \mathcal{D}_{\mathbb{Z}^d,s,\mathbf{y}}) \leq \exp(\pi\frac{\|\mathbf{y}\|_2^2}{s^2} + 2\frac{\|\mathbf{y}\|_2}{s}\sqrt{\pi \ln \varepsilon^{-1}})$. For $M > 1$, the bound is less than M if $s > \|\mathbf{y}\|_2 \cdot \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}})$.*

2.3 Algebraic Number Theory

We now give the necessary background in algebraic number theory. A number field $K = \mathbb{Q}(\zeta)$ is an extension field of $\mathbb{Q}$ of finite degree n obtained by adjoining an algebraic number ζ. The unique monic polynomial $f \in \mathbb{Q}[X]$ of smallest

degree that vanishes at ζ is called the minimal polynomial of K. Its degree is the degree of K. The set of algebraic integers in K defines a ring R called the ring of integers of K, sometimes denoted by $\mathcal{O}_K$. We also define $R_q = R/qR$ for any modulus $q \geq 2$. Although most of our result apply to general number fields, the rest of the paper focuses on cyclotomic fields. For $\nu \not\equiv 2 \bmod 4$, the cyclotomic field of conductor ν is $K = \mathbb{Q}(\zeta_\nu)$ where ζ_ν is a primitive ν-th root of unity. Its degree is $n = \varphi(\nu)$, where φ is the Euler totient function, and its ring of integers is $R = \mathbb{Z}[\zeta_\nu] \cong \mathbb{Z}[X]/\langle \Phi_\nu \rangle$ where Φ_ν is the ν-th cyclotomic polynomial. A particularly popular case is when $\nu = 2^{\mu+1}$, which we call power-of-two cyclotomic field, as it results in $n = 2^\mu$ and $\Phi_\nu = X^n + 1$.

Embeddings. Field and ring elements can be naturally embedded into $\mathbb{R}^n$ by their coefficient vector when seen as polynomials in ζ or X. We call τ the coefficient embedding of R, i.e., for all $r = \sum_{i \in [0,n-1]} r_i \zeta^i \in R$, $\tau(r) = [r_0 | \dots | r_{n-1}]^T$. One can extend τ to vectors of R^d by concatenating the coefficient embeddings of each vector entry. For an integer η, we define $S_\eta = \tau^{-1}([-\eta, \eta]^n)$ We also define the usual norms $\|\cdot\|_p$ over R by $\|r\|_p := \|\tau(r)\|_p$.

Multiplication Matrices. For all $r, s \in R$, $\tau(rs) = M_\tau(r)\tau(s)$, where $M_\tau(r)$ is the multiplication matrix of $\mathbb{R}^{n \times n}$ associated to r with respect to τ. In the field of minimal polynomial $f = X^n + \sum_{i \in [0,n-1]} f_i X^i$, we have $M_\tau(r) = \sum_{i \in [0,n-1]} r_i \mathbf{C}^i$ where

$$\mathbf{C} = \begin{bmatrix} 0 & & 0 & -f_0 \\ & & & -f_1 \\ & \mathbf{I}_{n-1} & & \vdots \\ & & & -f_{n-1} \end{bmatrix}$$

We naturally extend M_τ to matrices $\mathbf{A} = [a_{i,j}]_{i,j} \in R^{d \times m}$ entrywise by $M_\tau(\mathbf{A}) = [M_\tau(a_{i,j})]_{i,j} \in \mathbb{R}^{nd \times nm}$. Then, we define $\|\mathbf{A}\|_2$ as $\|M_\tau(\mathbf{A})\|_2$.

Lattices. For any $\mathbf{A} \in R_q^{d \times m}$, we define $\mathcal{L}_q^\perp(\mathbf{A}) = \{\mathbf{x} \in R^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod qR\}$. For any $\mathbf{u} \in R_q^d$, we similarly define $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A}) = \{\mathbf{x} \in R^m : \mathbf{A}\mathbf{x} = \mathbf{u} \bmod qR\}$.

Gaussians. For a positive definite matrix $\mathbf{S} \in \mathbb{R}^{nd \times nd}$, we define the discrete Gaussian distribution over R^d by $\tau^{-1}(\mathcal{D}_{\tau(R^d), \sqrt{\mathbf{S}}})$, which we denote by $\mathcal{D}_{R^d, \sqrt{\mathbf{S}}}$. Since $\tau(R^d) = \mathbb{Z}^{nd}$ in cyclotomic fields, the distribution corresponds to sampling an integer vector according to $\mathcal{D}_{\mathbb{Z}^{nd}, \sqrt{\mathbf{S}}}$ which thus defines a vector of R^d via τ^{-1}.

2.4 Hardness Assumptions

We now recall the relevant hardness assumptions for our work, namely the (resp. *Inhomogeneous*) *Module Short Integer Solution* M-SIS (resp. M-ISIS) and *Module Learning With Errors* M-LWE problems [LS15]. We consider the problems in their Hermite Normal Form, i.e., we specify the identity in the M-SIS and M-ISIS matrix, and we use the same distribution for the M-LWE secret and error. We also consider a version of M-ISIS and M-SIS which imposes an infinity norm constraint on the solution for our signature scheme Phoenix.

Definition 2.1 (M-ISIS and M-SIS). *Let K be a number field of degree n and R its ring of integers. Let d, m, q be positive integers and $\beta, \beta_\infty > 0$ with $m > d$. The Module Inhomogeneous Short Integer Solution problem in Hermite Normal Form* $\text{M-ISIS}_{n,d,m,q,\beta,\beta_\infty}$ *asks to find* $\mathbf{x} \in \mathcal{L}_q^{\mathbf{u}}([\mathbf{I}_d|\mathbf{A}'])$ *such that* $\|\mathbf{x}\|_2 \leq \beta$ *and* $\|\mathbf{x}\|_\infty \leq \beta_\infty$*, given* $\mathbf{A}' \hookleftarrow U(R_q^{d\times m-d})$ *and* $\mathbf{u} \hookleftarrow U(R_q^d)$*. When* $\mathbf{u} = \mathbf{0}$ *we call it* M-SIS *and expect* $\mathbf{x}$ *to be non-zero as well. When only considering the Euclidean norm, we remove the subscript* β_∞*.*

The advantage of a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ against $\text{M-ISIS}_{n,d,m,q,\beta,\beta_\infty}$ is defined by

$$\text{Adv}_{\text{M-ISIS}}[\mathcal{A}] = \mathbb{P}\left[\mathbf{x} \in \mathcal{L}_q^{\mathbf{u}}([\mathbf{I}_d|\mathbf{A}']) \wedge \|\mathbf{x}\|_2 \leq \beta \wedge \|\mathbf{x}\|_\infty \leq \beta_\infty : \mathbf{x} \leftarrow \mathcal{A}(\mathbf{A}', \mathbf{u})\right],$$

where the probability is over the randomness of $(\mathbf{A}', \mathbf{u})$ and the random coins of $\mathcal{A}$. When the parameters are clear from the context, we define the hardness bound as $\varepsilon_{\text{M-ISIS}} = \sup_{\mathcal{A}\ \text{PPT}} \text{Adv}_{\text{M-ISIS}}[\mathcal{A}]$. We define these quantities similarly for M-SIS. We now present the M-LWE problem in its variant with multiple secrets which we use throughout the paper.

Definition 2.2 (M-LWE). *Let K be a number field of degree n and R its ring of integers. Let d, m, k, q be positive integers and $\mathscr{D}_r$ a distribution on R. The decision Module Learning With Errors problem* $\text{M-LWE}^k_{n,d,m,q,\mathscr{D}_r}$ *asks to distinguish between the following distributions: (1)* $(\mathbf{A}', [\mathbf{I}_m|\mathbf{A}']\mathbf{R} \bmod qR)$*, where* $\mathbf{A}' \sim U(R_q^{m\times d})$ *and* $\mathbf{R} \sim \mathcal{D}_r^{d+m\times k}$*, and (2)* $(\mathbf{A}', \mathbf{B})$*, where* $\mathbf{A}' \sim U(R_q^{m\times d})$ *and* $\mathbf{B} \sim U(R_q^{m\times k})$*. The search variant asks to find* $\mathbf{R}$ *given a sample from (1).*

The advantage of a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ against decision $\text{M-LWE}^k_{n,d,m,q,\mathcal{D}_r}$ is defined by

$$\text{Adv}_{\text{M-LWE}}[\mathcal{A}] = |\mathbb{P}\left[\mathcal{A}(\mathbf{A}', [\mathbf{I}_m|\mathbf{A}']\mathbf{R}) = 1\right] - \mathbb{P}\left[\mathcal{A}(\mathbf{A}', \mathbf{B}) = 1\right]|,$$

When the parameters are clear from the context, we define the hardness bound as $\varepsilon_{\text{M-LWE}} = \sup_{\mathcal{A}\ \text{PPT}} \text{Adv}_{\text{M-LWE}}[\mathcal{A}]$. Additionally, a standard hybrid argument shows that $\text{M-LWE}^k_{n,d,m,q,\mathcal{D}_r}$ is at least as hard as $\text{M-LWE}^1_{n,d,m,q,\mathcal{D}_r}$ at the expense of a loss factor k in the reduction. The same holds for the search variant.

3 Revisiting Trapdoor Sampling

We here focus on the trapdoor preimage sampling procedure proposed by Lyubashevsky and Wichs [LW15] for Micciancio-Peikert trapdoors [MP12] (which we late abbreviate MP trapdoors). We start by recalling the structure of MP trapdoors and describe the LW sampler from [LW15] which suffers from very restrictive constraints on the parameters, as explained in Section 1. We provide an improved analysis of the LW sampler which gets rid of those parameter constraints, leading to a performance improvement of about 60% over the sampler analysis from [LW15], and of around 25% over the MP sampler. Our result places however moderate constraints on the applications, albeit easily met in practice as we discuss.

3.1 Micciancio-Peikert Preimage Sampling

The notion of trapdoors introduced by Micciancio and Peikert [MP12] is very versatile and has been extensively used in cryptographic constructions, including many advanced lattice-based primitives. These trapdoors involve matrices $\mathbf{A_H}$ of the form

$$\mathbf{A_H} = [\mathbf{A}|\mathbf{HG} - \mathbf{AR}] \bmod qR \in R_q^{d\times d(2+k)},$$

where $\mathbf{H} \in R_q^{d\times d}$ is an invertible tag matrix, $\mathbf{G} \in R^{d\times dk}$ a primitive gadget matrix, and $\mathbf{R} \in R^{2d\times dk}$ a short matrix corresponding to the trapdoor. In what follows, we consider the gadget matrix of [MP12] in base $b \geq 2$, i.e., $\mathbf{G} = \mathbf{I}_d \otimes [1|b|\dots|b^{k-1}] \in \mathbb{Z}^{d\times dk} \subseteq R^{d\times dk}$ where[5] $k = \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$, but any other gadget matrix would work, as long as it enables to easily compute short preimages.

The sampling algorithm relies on the link between such matrices $\mathbf{A_H}$ and the gadget matrix $\mathbf{G}$, that is

$$\mathbf{A_H} \begin{bmatrix} \mathbf{R} \\ \mathbf{I}_{dk} \end{bmatrix} = \mathbf{HG} \bmod qR.$$

Thence, if $\mathbf{z}$ is a short vector in $\mathcal{L}_q^{\mathbf{u}}(\mathbf{HG})$, then $\mathbf{v} = [(\mathbf{Rz})^T|\mathbf{z}^T]^T$ is a short vector in $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A_H})$, i.e., verifying $\mathbf{A_H v} = \mathbf{u} \bmod qR$, that is $\mathbf{v}$ is a preimage of $\mathbf{u}$ by $\mathbf{A_H}$. Unfortunately, $\mathbf{v}$ leaks information about the trapdoor $\mathbf{R}$ which is undesirable in cryptographic applications as $\mathbf{R}$ usually corresponds to the long-term secret key. To circumvent this issue, the authors use the Gaussian convolution theorem [Pei10, Thm. 3.1] to perturb $\mathbf{v}$ in order to make the final samples independent of $\mathbf{R}$. In more details, they sample a (highly) non-spherical Gaussian perturbation $\mathbf{p} = [\mathbf{p}_1^T|\mathbf{p}_2^T]^T \sim \mathcal{D}_{R^{d(2+k)},\sqrt{\mathbf{S}}}$ with

$$\mathbf{S} = M_\tau \left(s^2 \mathbf{I}_{d(2+k)} - s_{\mathbf{G}}^2 \begin{bmatrix} \mathbf{R}\mathbf{R}^T & \mathbf{R} \\ \mathbf{R}^T & \mathbf{I}_{dk} \end{bmatrix} \right),$$

and then compensate this perturbation by sampling $\mathbf{z} \sim \mathcal{D}_{\mathcal{L}_q^{\mathbf{x}}(\mathbf{G}),s_{\mathbf{G}}}$ with $\mathbf{x} = \mathbf{H}^{-1}(\mathbf{u} - \mathbf{A_H p}) \bmod qR$. The output sample is then $\mathbf{v}' = [(\mathbf{p}_1 + \mathbf{Rz})^T|(\mathbf{p}_2 + \mathbf{z})^T]^T$. By the convolution theorem, $\mathbf{v}'$ is statistically close to a Gaussian distribution over $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A_H})$ with parameter s, which no longer depends on $\mathbf{R}$.

Therefore, from the security standpoint, the approach above perfectly addresses the problem of preimage sampling for cryptographic applications. However, if we reconsider the unperturbed vector $\mathbf{v} = [(\mathbf{Rz})^T|\mathbf{z}^T]^T$, we note that the convolution is now applied to both parts in the same way. This does not seem optimal as the bottom section of $\mathbf{v}$ is independent of $\mathbf{R}$ and as $\mathbf{Rz}$ is always larger than $\mathbf{z}$. Unfortunately, this seems inherent to the approach stated in [Pei10, Sec. 1.3] which only considers covariance matrices of the form $s^2\mathbf{I} - \mathbf{S}_1$ for some covariance matrix $\mathbf{S}_1$. Ideally, we would like to select a perturbation that only affects the top component, typically:

[5] See Remark 3.2 for details on the definition of k.

$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{0} \end{bmatrix} \sim \mathcal{D}_{R^{d(2+k)},\sqrt{\mathbf{S}}}, \text{ with } \mathbf{S} = M_\tau \left(\begin{bmatrix} s^2\mathbf{I}_{2d} - s_{\mathbf{G}}^2 \mathbf{R}\mathbf{R}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \right).$$

However, when sampling $\mathbf{z}$ and outputting $\mathbf{p} + [\mathbf{R}^T | \mathbf{I}_{dk}]^T \mathbf{z}$, we end up with a joint probability of covariance (up to applying M_τ)

$$\begin{bmatrix} s^2\mathbf{I}_{2d} - s_{\mathbf{G}}^2 \mathbf{R}\mathbf{R}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + s_{\mathbf{G}}^2 \begin{bmatrix} \mathbf{R}\mathbf{R}^T & \mathbf{R} \\ \mathbf{R}^T & \mathbf{I}_{dk} \end{bmatrix} = \begin{bmatrix} s^2\mathbf{I}_{2d} & s_{\mathbf{G}}^2 \mathbf{R} \\ s_{\mathbf{G}}^2 \mathbf{R}^T & s_{\mathbf{G}}^2 \mathbf{I}_{dk} \end{bmatrix},$$

which again leaks information about $\mathbf{R}$. This highlights the need to hide both $\mathbf{Rz}$ and $\mathbf{z}$ to rely on the convolution technique. Intuitively, the first component $\mathbf{v}_1 = \mathbf{p}_1 + \mathbf{Rz}$ can be seen as a Gaussian distribution with a secret center $\mathbf{Rz}$. Looking at its marginal distribution, one could use standard techniques to hide this secret center, namely convolution when $\mathbf{z}$ is Gaussian or noise flooding (based on either the statistical distance or the Rényi divergence) if $\mathbf{z}$ is non-Gaussian. However, giving $\mathbf{v}_2 = \mathbf{z}$ provides side information on this secret center which explains why $\mathbf{z}$ also has to be perturbed for the convolution technique to be meaningful. We therefore need a middle way between this efficient, but insecure, approach and the one from [MP12] that does not seem optimal for the type of asymmetric vectors we have to perturb.

From the implementation standpoint, the MP approach also leads to some specific problems. It indeed requires the sampling of a perturbation vector $\mathbf{p}$ from a (highly) non-spherical Gaussian distribution. Such a perturbation sampling is rather costly and represents the most part of the computation time of preimage sampling. The gadget sampling step (sampling $\mathbf{z} \hookleftarrow \mathcal{D}_{\mathcal{L}_q^{\mathbf{x}}(\mathbf{G}),s_{\mathbf{G}}}$) also requires the sampling of non-spherical Gaussian perturbations when q is not a power of the gadget base b. The latter case has been considered in several works [GM18,ZY22] which show how to leverage specific structures in the basis of $\mathcal{L}_q^\perp(\mathbf{G})$ to enable more efficient sampling over $\mathcal{L}_q^\perp(\mathbf{G})$. Unfortunately, this does not work for the perturbation $\mathbf{p}$ and the covariance matrix $\mathbf{S}$ we consider because $\mathbf{R}$ is random. Another downside is that this convolution method is seemingly limited to Gaussian distributions, which limits the possible preimage distributions.

3.2 A More Flexible Preimage Sampler

To circumvent these shortcomings, Lyubashevsky and Wichs [LW15] proposed a more flexible preimage sampling procedure which only perturbs the top component.

3.2.1 Description. The approach from [LW15] can be seen as combining the features of tag-friendly gadget-based preimage sampling with rejection sampling that is extensively used in Fiat-Shamir with Aborts (FSwA) signatures. Let $\mathbf{G}^{-1}(\cdot)$ be the entry-wise base-b decomposition of vectors of R_q^d. As we explain below in Remark 3.2, we consider a centered representation of $\mathbb{Z}_q$ which results

in a signed base-b decomposition. Hence, $\mathbf{G}^{-1}$ maps to vectors of S_{b-1}^{dk}. The intuition is to sample a perturbation $\mathbf{p}_1 \in R^m$ from a source distribution $\mathscr{D}_s$. Further, instead of using Gaussian $\mathbf{G}$-sampling, we simply use $\mathbf{G}^{-1}$ and obtain $\mathbf{v}_2 = \mathbf{G}^{-1}(\mathbf{H}^{-1}(\mathbf{u} - \mathbf{A}\mathbf{p}_1))$. Then, we can define $\mathbf{v}_1 = \mathbf{p}_1 + \mathbf{R}\mathbf{v}_2$ so that the relation $\mathbf{A}_\mathbf{H}\mathbf{v} = \mathbf{u}$ is verified, and apply rejection sampling to make $\mathbf{v}_1$ independent of $\mathbf{R}\mathbf{v}_2$ and in turn $\mathbf{R}$. This setting is reminiscent of lattice-based zero-knowledge arguments or Lyubashevsky's signature scheme [Lyu12], where $\mathbf{R}$ is the witness, $\mathbf{p}_1$ is the mask, $\mathbf{A}\mathbf{p}_1$ is a commitment to the mask, $\mathbf{v}_2$ is the challenge, and $\mathbf{v}_1$ is the response to the challenge. We slightly modify the presentation of the sampler from [LW15] by taking the matrix $\mathbf{A}$ in Hermite Normal Form. Concretely, throughout the rest of the paper, $\mathbf{A} = [\mathbf{I}_d|\mathbf{A}']$ for a matrix $\mathbf{A}'$ of dimension $d \times (m-d)$.

Algorithm 3.1: SamplePreRej($\mathbf{R}; \mathbf{A}', \mathbf{H}, \mathbf{u}, \mathscr{D}_s, \mathscr{D}_t$)

Input (offline phase)**:** Matrix $\mathbf{A}' \in R_q^{d\times(m-d)}$, Source distribution $\mathscr{D}_s$ over R^m.
Input (online phase)**:** Trapdoor $\mathbf{R} \in R^{m\times dk}$, Tag $\mathbf{H} \in GL_d(R_q)$, Syndrome $\mathbf{u} \in R_q^d$, Target distribution $\mathscr{D}_t$ over R^m such that rejection sampling can be performed with respect to the source distribution $\mathscr{D}_s$.

Offline phase
1. $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$.
2. $\mathbf{w} \leftarrow [\mathbf{I}_d|\mathbf{A}']\mathbf{p}_1 \bmod qR$.

Online phase
3. $\mathbf{x} \leftarrow \mathbf{H}^{-1}(\mathbf{u} - \mathbf{w}) \bmod qR$. ▷ Syndrome correction
4. $\mathbf{v}_2 \leftarrow \mathbf{G}^{-1}(\mathbf{x}) \in S_{b-1}^{dk}$. ▷ Deterministic
5. $\mathbf{v}_1 \leftarrow \mathbf{p}_1 + \mathbf{R}\mathbf{v}_2$.
6. Sample a continuous $u \hookleftarrow U([0,1])$.
7. **if** $u > \min\left(1, \frac{\mathscr{D}_t(\mathbf{v}_1)}{M\cdot\mathscr{D}_s(\mathbf{p}_1)}\right)$ **then** go back to 1. ▷ Rejection

Output: $\mathbf{v} = \begin{bmatrix}\mathbf{v}_1\\ \mathbf{v}_2\end{bmatrix}$.

3.2.2 Current Limitations. At first glance, the approach from [LW15] seems to fully achieve what we wanted to do in Section 3.1, namely to completely break the symmetry between $\mathbf{v}_1$ and $\mathbf{v}_2$ to reduce the size of $\mathbf{v}_2$. However, in practice, the choice of parameters and suitable distributions $\mathscr{D}_s, \mathscr{D}_t$ is conditioned by the security requirements coming from the simulation result of [LW15, Thm. 3.1]. Unfortunately, the latter is too restrictive in most cases, which explains why it does not lead to improvements on the preimage size, as we explain below.

Concretely, in [LW15, Thm. 3.1], it is shown that the output distribution of SamplePreRej is statistically close to some ideal distribution that does not depend on the trapdoor $\mathbf{R}$ for an arbitrary (potentially adversarial) syndrome $\mathbf{u}$. It means that a preimage $\mathbf{v}$ of $\mathbf{u}$ can be simulated without resorting to the trapdoor $\mathbf{R}$, and thus does not leak information on $\mathbf{R}$. There are however some challenges to overcome in order to prove this result. The first one is to identify this ideal distribution that must additionally be close to the one of actual preimages. If

we focus on the $\mathbf{v}_2$ component of these preimages, we indeed note that the SamplePreRej algorithm above generates them as $\mathbf{G}^{-1}(\mathbf{H}^{-1}(\mathbf{u}-\mathbf{w}))$ where $\mathbf{w} = \mathbf{A}\mathbf{p}_1$. If $\mathbf{w}$ is non-uniform, then so is $\mathbf{H}^{-1}(\mathbf{u}-\mathbf{w})$, which makes the distribution of $\mathbf{v}_2$ complex to define when $\mathbf{u}$ is arbitrary.

It therefore seems necessary to assume that $\mathbf{A}\mathbf{p}_1$ is close to uniform, but at this stage one could still wonder whether a computational argument is sufficient. Unfortunately we here face a second challenge which is due to the very nature of the perturbation $\mathbf{p}_1$. Indeed, $\mathbf{p}_1$ does not only affect the syndrome (through $\mathbf{A}\mathbf{p}_1$) but also the preimage as it is eventually added to its upper component to form $\mathbf{v}_1$. In a computational argument, one would end up with an intermediate game where $\mathbf{A}\mathbf{p}_1$ would be replaced by some random vector $\mathbf{r}$, but then how to generate $\mathbf{v}_1$? The syndrome would indeed be $\mathbf{u}+\mathbf{r}$, which seems impossible to invert without resorting to the trapdoor since the reduction does not control $\mathbf{u}$.

This is why the authors of [LW15] need to assume that $\mathbf{A}\mathbf{p}_1$ is *statistically* close to uniform requiring $\mathbf{p}_1$ to have a high entropy in order to use a regularity lemma, which in turn leads to large parameters (either in the dimension m of $\mathbf{p}_1$, or in the size of its entries). This in particular prevents them from using a (much more efficient) computational instantiation of MP trapdoors where $m = 2d$. This results in significant performance losses which cancel out the benefits of having smaller $\mathbf{v}_2$. In addition, regularity lemmas generally require the modulus q to be prime and with low splitting in the ring R, which may be undesirable for concrete applications.

We give concrete parameter and performance estimates in Table 4.2 following the original result and parameter selection from [LW15, Sec 3.2] in the Gaussian case, i.e., when $\mathscr{D}_s = \mathscr{D}_t = \mathcal{D}_{R^m,s}$ is a spherical discrete Gaussian of width s. Their simulation result leads to choosing $m = dk = d\lceil \log_b q \rceil$ and $s = \alpha \cdot (b-1)\sqrt{ndk}(\sqrt{ndk} + \sqrt{ndk} + t)$, for a constant factor $\alpha \approx 8$. Overall it yields a signature of around 20 KB whereas the original MP sampler yields signature of approximately 10.2 KB.

3.3 Improved Simulatability of Preimages.

We now explain how to get rid of these requirements when the syndrome follows a prescribed uniform distribution. This is for example the case for GPV signatures [GPV08], where the syndrome $\mathbf{u}$ is the hash output $\mathcal{H}(\mathbf{m})$ of the message $\mathbf{m}$, where $\mathcal{H}$ is modelled as a random oracle. With this assumption, we can drastically change the proof strategy. Indeed, we first note that we no longer have to study the distribution of $\mathbf{v}$ conditioned on some arbitrary $\mathbf{u}$ as we can now consider the joint distribution of $\mathbf{v}$ and $\mathbf{u}$. Put differently, we can now manipulate these two vectors as long as their joint distribution is correct, which offers a lot more flexibility in the proof. In particular, this allows to circumvent the challenges faced in the proof of [LW15] because we can now leverage the randomness of $\mathbf{u}$ to compensate the one introduced by the computational assumption. More precisely, this allows us to specify the expected distribution of $\mathbf{v}_2$ as $\mathbf{H}^{-1}(\mathbf{u}-\mathbf{A}\mathbf{p}_1)$ is now uniform because $\mathbf{u}$ is uniform and independent of $\mathbf{A}\mathbf{p}_1$.

This alleviates the restriction on $\mathbf{Ap}_1$ being statistically uniform, while still being able to simulate the pairs $(\mathbf{v}, \mathbf{u})$ without resorting to the trapdoor $\mathbf{R}$. Note that $\mathbf{p}_1$ still needs to have a sufficient entropy so as to hide $\mathbf{Rv}_2$, which is given by the rejection sampling condition. This trapdoor-independence property of the preimages is necessary for cryptographic applications, e.g., signatures, as an adversary can usually have access to many such preimages (and syndromes) for a single key. As a consequence, we no longer need a large perturbation (either in norm or dimension), which leads to improved performances, as illustrated by the tables in Section 4.

We provide our new simulation result in Theorem 3.1, which we instantiate for Gaussian distributions in Corollary 3.1.

Theorem 3.1. *Let K be a number field, and R its ring of integers. Let d, q, b be positive integers with $b \geq 2$, and let $k = \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$. Let $\mathscr{D}_r, \mathscr{D}_s, \mathscr{D}_t$ be three distributions over $R^{2d \times dk}$, R^{2d} and R^{2d} respectively. Let $\mathbf{A}' \sim U(R_q^{d \times d})$, $\mathbf{R} \sim \mathscr{D}_r$, $\mathbf{H} \in GL_d(R_q)$ and $\mathbf{A} = [\mathbf{I}_d | \mathbf{A}'] \in R_q^{d \times 2d}$. Then, let $Y \subseteq R^{2d}$ be the support of the distribution of $\mathbf{R} \cdot \mathbf{G}^{-1}(U(R_q^d))$. Let $M > 1, \varepsilon \in [0, 1/2]$ such that $\max_{\mathbf{Rv}_2 \in Y} RD_\infty^\varepsilon(\mathscr{D}_t \| \mathscr{D}_s^{+\mathbf{Rv}_2}) \leq M$. We then define two distributions*

$\mathcal{P}_1$:
$\mathbf{u} \hookleftarrow U(R_q^d)$, *and* $\mathbf{v} \leftarrow \mathsf{SamplePreRej}(\mathbf{R}; \mathbf{A}', \mathbf{H}, \mathbf{u}, \mathscr{D}_s, \mathscr{D}_t)$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{P}_2$:
1. $\mathbf{v}_1 \hookleftarrow \mathscr{D}_t$, $\mathbf{v}_2 \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$.
2. $\mathbf{v} \leftarrow [\mathbf{v}_1^T | \mathbf{v}_2^T]^T$.
3. $\mathbf{u} \leftarrow [\mathbf{A} | \mathbf{HG} - \mathbf{AR}]\mathbf{v} \bmod qR$.
4. *With probability* $1 - 1/M$ *go back to 1.*

Output: $(\mathbf{v}, \mathbf{u})$.

Then, $\Delta(\mathcal{P}_1, \mathcal{P}_2) \leq \varepsilon$ and for all $a \in (1, +\infty]$, $RD_a(\mathcal{P}_1 \| \mathcal{P}_2) \leq 1/(1-\varepsilon)^{a/(a-1)}$.

Proof. We define the following hybrid distributions from $\mathcal{H}_1$ to $\mathcal{H}_5$, where $\mathcal{H}_1 = \mathcal{P}_1$ and $\mathcal{H}_5 = \mathcal{P}_2$.

$\mathcal{H}_1$:
$\mathbf{u} \hookleftarrow U(R_q^d)$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{x}' \leftarrow \mathbf{u} - \mathbf{Ap}_1 \bmod qR$, $\mathbf{v}_2 \leftarrow \mathbf{G}^{-1}(\mathbf{H}^{-1}\mathbf{x}')$, $\mathbf{v}_1 \leftarrow \mathbf{p}_1 + \mathbf{Rv}_2$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1)/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1^T | \mathbf{v}_2^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_2$:
$\mathbf{x}' \hookleftarrow U(R_q^d)$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{u} \leftarrow \mathbf{x}' + \mathbf{Ap}_1 \bmod qR$, $\mathbf{v}_2 \leftarrow \mathbf{G}^{-1}(\mathbf{H}^{-1}\mathbf{x}')$, $\mathbf{v}_1 \leftarrow \mathbf{p}_1 + \mathbf{Rv}_2$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1)/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1^T | \mathbf{v}_2^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_3$:
$\mathbf{v}_2 \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$, $\mathbf{x}' \leftarrow \mathbf{HGv}_2 \bmod qR$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{u} \leftarrow \mathbf{x}' + \mathbf{Ap}_1 \bmod qR$, $\mathbf{v}_1 \leftarrow \mathbf{p}_1 + \mathbf{Rv}_2$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1)/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1^T | \mathbf{v}_2^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_4$ $\mathbf{v}_2 \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{v}_1 \leftarrow \mathbf{p}_1 + \mathbf{R}\mathbf{v}_2$, $\mathbf{v} \leftarrow [\mathbf{v}_1^T|\mathbf{v}_2^T]^T$, $\mathbf{u} \leftarrow [\mathbf{A}|\mathbf{HG} - \mathbf{AR}]\mathbf{v} \bmod qR$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1)/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. If not go to output.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_5$ $\mathbf{v}_2 \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$, $\mathbf{v}_1 \hookleftarrow \mathscr{D}_t$, $\mathbf{v} \leftarrow [\mathbf{v}_1^T|\mathbf{v}_2^T]^T$, $\mathbf{u} \leftarrow [\mathbf{A}|\mathbf{HG} - \mathbf{AR}]\mathbf{v} \bmod qR$. Then $u \hookleftarrow U([0,1])$ and restart if $u > 1/M$. If not go to output.
Output: $(\mathbf{v}, \mathbf{u})$.

Let us now show that these distributions are statistically close to each other.

$\mathcal{H}_1$ - $\mathcal{H}_2$: Here we just change the sampling order of $\mathbf{u}$ and $\mathbf{x}'$. In $\mathcal{H}_2$ the vector $\mathbf{x}'$ is uniform and independent of $\mathbf{A}\mathbf{p}_1$ implying that $\mathbf{u}$ is also uniform, as in $\mathcal{H}_1$. Hence $\mathcal{H}_1$ and $\mathcal{H}_2$ are identically distributed.

$\mathcal{H}_2$ - $\mathcal{H}_3$: We now change the way $\mathbf{x}'$ is generated. Notice that, for correctness, once $\mathbf{x}'$ is fixed then so is $\mathbf{v}_2$ and vice-versa. In $\mathcal{H}_2$, since $\mathbf{H}$ is in $GL_d(R_q)$, $\mathbf{H}^{-1}\mathbf{x}'$ also follows the uniform distribution over R_q^d. As a result, $\mathbf{v}_2$ follows exactly $\mathbf{G}^{-1}(U(R_q^d))$ as in $\mathcal{H}_3$. Also, $\mathbf{x}'$ is coherently set in $\mathcal{H}_3$. Indeed, in $\mathcal{H}_2$, we have $\mathbf{HG}\mathbf{v}_2 = \mathbf{H}(\mathbf{H}^{-1}\mathbf{x}') \bmod qR = \mathbf{x}' \bmod qR$. Thence, $\mathcal{H}_2$ and $\mathcal{H}_3$ are identically distributed as well.

$\mathcal{H}_3$ - $\mathcal{H}_4$: $\mathcal{H}_4$ is merely a re-writing of $\mathcal{H}_3$. Indeed, in $\mathcal{H}_3$, $\mathbf{x}'$ only acts as an intermediate vector to define $\mathbf{u}$. Defining $\mathbf{R}' = [\mathbf{R}^T|\mathbf{I}_{m_2}]^T$, we have $[\mathbf{A}|\mathbf{HG} - \mathbf{AR}]\mathbf{R}' = \mathbf{HG} \bmod qR$. In $\mathcal{H}_3$, this yields

$$\begin{aligned}\mathbf{u} = \mathbf{HG}\mathbf{v}_2 + \mathbf{A}\mathbf{p}_1 \bmod qR &= [\mathbf{A}|\mathbf{HG} - \mathbf{AR}]\mathbf{R}'\mathbf{v}_2 + \mathbf{A}\mathbf{p}_1 \bmod qR \\ &= [\mathbf{A}|\mathbf{HG} - \mathbf{AR}]\mathbf{v} \bmod qR,\end{aligned}$$

as $\mathbf{v} = [\mathbf{p}_1^T|\mathbf{0}]^T + \mathbf{R}'\mathbf{v}_2$. Again, $\mathcal{H}_3$ and $\mathcal{H}_4$ are identical.

$\mathcal{H}_4$ - $\mathcal{H}_5$: We now change the way $\mathbf{v}_1$ is generated by using the rejection sampling result. In $\mathcal{H}_4$, $\mathbf{R}\mathbf{v}_2$ is distributed according to $\mathbf{R} \cdot \mathbf{G}^{-1}(U(R_q^d))$ with support Y as defined in the theorem statement. By our assumptions on Y, $\mathscr{D}_s$, $\mathscr{D}_t$, the rejection sampling result from Lemma 2.2 yields that

$$\Delta((\mathbf{v}_1, \mathbf{v}_2)_{\mathcal{H}_4}, (\mathbf{v}_1, \mathbf{v}_2)_{\mathcal{H}_5}) \leq \varepsilon \quad \text{and} \quad RD_a((\mathbf{v}_1, \mathbf{v}_2)_{\mathcal{H}_4} \| (\mathbf{v}_1, \mathbf{v}_2)_{\mathcal{H}_5}) \leq \frac{1}{(1-\varepsilon)^{\frac{a}{a-1}}},$$

for all $a > 1$. By the data processing inequality of the statistical distance and Rényi divergence, it holds

$$\Delta(\mathcal{H}_4, \mathcal{H}_5) \leq \varepsilon \quad \text{and} \quad RD_a(\mathcal{H}_4 \| \mathcal{H}_5) \leq \frac{1}{(1-\varepsilon)^{\frac{a}{a-1}}}.$$

Since $\mathcal{H}_1 = \mathcal{P}_1$ and $\mathcal{H}_5 = \mathcal{P}_2$, combining the above gives the result.

□

Theorem 3.1 also provides the simulation in Rényi divergence because, as noted for example in [Pre17], it usually leads to tighter constructions. One can indeed take a much larger ε for (almost) the same security guarantees, which in turn relaxes the constraints on other parameters.

Because our new analysis does not require any regularity result to argue that $\mathbf{Ap}_1$ is statistically uniform, we do not need to place any restrictions on the modulus q, nor the field K. Typically, regularity lemmas in the module setting are usually restricted to monogenic fields and/or to prime modulus with low splitting, i.e., such that the ideal qR factors into a small number of distinct prime ideals. Our simulation avoids these constraints altogether. Additionally, the result specifies to the integers by choosing the field $K = \mathbb{Q}$, and thus the ring $R = \mathbb{Z}$, of degree $n = 1$.

Remark 3.1. Our proof strategy would still work if the syndrome $\mathbf{u}$ is statistically uniform and not necessarily a hash output. This is for example the case in the recent signature with efficient protocols of [JRS23] where they simulate one signature query $\mathbf{v}$ along with the public key syndrome $\mathbf{u}$.

Remark 3.2. Note that when working with centered modular arithmetic, the gadget needs to invert possibly negative elements. For $w \in (-q/2, q/2] \cap \mathbb{Z}$, the gadget inversion thus takes the base-b decomposition of $|w|$ and multiplies all coefficients by the sign of w. Additionally, the elements have magnitude at most $\lceil (q-1)/2 \rceil$ and not $q-1$. The base-b decomposition thus requires k entries where $b^k - 1 \geq \lceil (q-1)/2 \rceil$ which indeed leads to $k = \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$ instead of $k = \lceil \log_b q \rceil$. This almost never differs for large bases and moduli except for rare corner cases, but when $b = 2$ for example this saves one dimension in the gadget length and thus d columns for $\mathbf{R}$.

3.3.1 Gaussian Instantiation. We instantiate Theorem 3.1 with a Gaussian distribution on $\mathbf{v}_1$ for a fair comparison with previous results. We thus choose $\mathscr{D}_r = U(S_1^{2d \times dk})$ for the trapdoor distribution, and we select $\mathscr{D}_s = \mathscr{D}_t = \mathcal{D}_{R^{2d},s}$ for the source and target distributions. For convenience, we write $\mathsf{SamplePreRej}(\mathbf{R}; \mathbf{A}', \mathbf{H}, \mathbf{u}, s)$ instead of specifying $\mathscr{D}_s$ and $\mathscr{D}_t$.

In order to set s, we need to derive a bound T on $\|\mathbf{R}\mathbf{v}_2\|_2$ to use Lemma 2.3. For that, we bound it by $\|\mathbf{R}\|_2 \|\mathbf{v}_2\|_2$, and apply the standard bound $\|\mathbf{R}\|_2 \leq \sqrt{2nd} + \sqrt{ndk} + t =: B$. We simply note that as the matrix is structured, this bound, which could be proven by [Ver12] in the unstructured case, is only verified empirically as in several works using lattice gadget trapdoors. To thoroughly match the conditions of the rejection sampling, we need to enforce this spectral bound on $\|\mathbf{R}\|_2$ before the sampling procedure. Since $\mathbf{R}$ represents the secret key, this bound should be enforced during key generation[6]. As it is verified with overwhelming probability, this only discards a negligible fraction of all the possible keys. We therefore actually apply Theorem 3.1 on $\mathscr{D}_r =$ "$U(S_1^{2d \times dk})$ conditioned on $\|\mathbf{R}\|_2 \leq B$". We then choose a repetition rate $M > 1$ and a loss ε, which both define the minimal slack $\alpha > 0$ so that $s = \alpha T$. This leads to the following corollary, which will be more convenient to use later.

[6] It is also the case for the original MP sampler as it may happen (albeit with negligible probability) that the sampler fails if $\mathbf{R}$ has norm larger than the bound used to set the Gaussian width s.

Corollary 3.1. *Let R be the power-of-two cyclotomic ring of degree n. Let d, q, b be positive integers with $b \geq 2$, and define the gadget dimension $k = \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$. Let $t > 0$, and $T = (b-1)\sqrt{ndk}(\sqrt{2nd} + \sqrt{ndk} + t)$. Let $M > 1$, $\varepsilon \in (0, 1/2]$ and define $\alpha = \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}})$. Finally $s = \alpha T$. Let $\mathbf{A}' \sim U(R_q^{d \times d})$, $\mathbf{R} \sim U(S_1^{2d \times dk})$ conditioned on $\|\mathbf{R}\|_2 \leq \sqrt{2nd} + \sqrt{ndk} + t$, and $\mathbf{H} \in GL_d(R_q)$. We define $\mathcal{P}_1$ and $\mathcal{P}_2$ the same way as in Theorem 3.1 but where $\mathscr{D}_s, \mathscr{D}_t$ are replaced with $\mathcal{D}_{R^{2d},s}$. Then, it holds that $\Delta(\mathcal{P}_1, \mathcal{P}_2) \leq \varepsilon$ and $RD_a(\mathcal{P}_1 \| \mathcal{P}_2) \leq 1/(1-\varepsilon)^{a/(a-1)}$ for all $a \in (1, +\infty]$.*

Proof. We simply have to verify that the smooth Rényi divergence condition of Theorem 3.1 holds. In our context, we restrict the matrices $\mathbf{R}$ to have a bounded spectral norm. Following the notations of Theorem 3.1, the distribution $\mathscr{D}_r$ consists in sampling $\mathbf{R}$ from $U(S_1^{2d \times dk})$ such that $\|\mathbf{R}\|_2 \leq B$, where $B = \sqrt{2nd} + \sqrt{ndk} + t$. The set Y is the support of $\mathbf{R} \cdot \mathbf{G}^{-1}(U(R_q^d))$. Hence, for all $\mathbf{R}\mathbf{v}_2$ in Y, we have $\|\mathbf{R}\mathbf{v}_2\|_2 \leq \|\mathbf{R}\|_2 \|\mathbf{v}_2\|_2 \leq B \cdot (b-1)\sqrt{ndk} = T$. We note that since $Y \subset R^{2d}$, we have $\mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{v}_2} = \mathcal{D}_{R^{2d},s,\mathbf{R}\mathbf{v}_2}$ for all $\mathbf{R}\mathbf{v}_2 \in Y$. Using Lemma 2.3, it thus holds that

$$\begin{aligned} RD_\infty^\varepsilon(\mathcal{D}_{R^{2d},s} \| \mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{v}_2}) &\leq \exp\left(\pi \frac{\|\mathbf{R}\mathbf{v}_2\|_2^2}{s^2} + 2\frac{\|\mathbf{R}\mathbf{v}_2\|_2}{s}\sqrt{\pi \ln \varepsilon^{-1}}\right) \\ &\leq \exp\left(\pi \frac{T^2}{s^2} + 2\frac{T}{s}\sqrt{\pi \ln \varepsilon^{-1}}\right) \\ &\leq M, \end{aligned}$$

where the last inequality follows from the fact that $s = \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}}) \cdot T$. Thence, $\max_{\mathbf{R}\mathbf{v}_2 \in Y} RD_\infty^\varepsilon(\mathcal{D}_{R^{2d},s} \| \mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{v}_2}) \leq M$. Theorem 3.1 then yields the result. □

In this specific instantiation of [LW15] and Theorem 3.1 with Gaussian distributions, we only reach widths s which are larger than the ones from [MP12]. Indeed, in the latter, $\mathbf{v}_1$ was distributed according to a discrete Gaussian of width $s = \Theta(b\|\mathbf{R}\|_2) = \Theta(b(\sqrt{2nd} + \sqrt{ndk}))$, while here we obtain a width $s = \Theta(b\sqrt{ndk}(\sqrt{2nd} + \sqrt{ndk}))$. However, in the meantime, we drastically reduce the size of $\mathbf{v}_2$, which largely compensate for the increase in size of $\mathbf{v}_1$ for typical parameters, as shown in Section 4.

4 Optimal Gadget Base and Sampler Performances

In the computational instantiation of MP trapdoors, the gadget base b is an important parameter to optimize over. Since the base defines the length of the gadget matrix $dk = d\lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$, choosing a larger base results in lower dimensional vectors, at the expense of a larger norm. As the norm only impacts the bitsize logarithmically while the dimension impacts it linearly, one could think that the optimal choice for b is around $\sqrt{q}$, thus resulting in $k = 2$, smaller

preimages and in turn smaller signatures. The goal of this section is to show that the optimal base actually depends on the preimage sampler. We illustrate our discussion with the instructive example of GPV signatures [GPV08]. We compare signatures generated using the original sampler of [MP12] (thereafter called MP signatures) with those generated using the original sampler (recalled in Algorithm 3.1) of [LW15] (called LW signatures) and those resulting from our new simulation result in Corollary 3.1 (later called LW* signatures). In the process, we demonstrate interesting improvement factors on the size of preimages resulting from our new analysis. This represents a step towards concrete practicality of constructions based on MP trapdoors.

GPV Signature. We briefly describe the signature framework from [GPV08] with MP trapdoors. The secret key $\mathbf{R}$ is drawn from $U(S_1^{m\times dk})$, and the public key is composed of $\mathbf{A} = [\mathbf{I}_d|\mathbf{A}'] \in R_q^{d\times m}$ and $\mathbf{B} = \mathbf{AR} \bmod qR$. As described before, the signature of a message $\mathbf{m} \in \{0,1\}^*$ consists of a short preimage $\mathbf{v} = [\mathbf{v}_1^T|\mathbf{v}_2^T]^T \in R^{m+dk}$ satisfying $[\mathbf{A}|\mathbf{G}-\mathbf{B}]\mathbf{v} = \mathcal{H}(\mathbf{m}) \bmod qR$. Since the matrix $\mathbf{A}$ has $\mathbf{I}_d$ as its first block, we can use similar tricks as for example [PFH+20,EFG+22,ETWY22] to reduce the signature size. The GPV signature now consists of $(\mathbf{v}_{1,2}, \mathbf{v}_2)$, where $\mathbf{v}_1 = [\mathbf{v}_{1,1}^T|\mathbf{v}_{1,2}^T]^T$, because $\mathbf{v}_{1,1}$ is determined by the verification equation as $\mathbf{v}_{1,1} = \mathcal{H}(\mathbf{m}) - \mathbf{A}'\mathbf{v}_{1,2} - (\mathbf{G}-\mathbf{B})\mathbf{v}_2 \bmod qR$. Below, the signature size is thus computed as $|\mathsf{sig}| = |\mathbf{v}_{1,2}| + |\mathbf{v}_2|$. The bitsize of Gaussian vectors is estimated by the entropy bound, which can be achieved using the rANS encoding as discussed in [ETWY22]. More precisely, for a discrete Gaussian vector of dimension N and width s, the entropy bound is close to $N/2 \cdot (1 + \log_2 s^2) = N(1/2 + \log_2 s)$.

Choosing the Gadget Base. The main difficulty when determining the optimal base for a given sampler is that b impacts both the bitsize evaluation of the signature and the hardness of the underlying computational assumptions. As the latter in turn affects the parameters (and hence the bitsize), this may lead to some counterintuitive situations.

For a given base, the minimal Gaussian parameter needed for MP signatures $\mathbf{v}$ is[7] $s \approx \alpha_1 b\|\mathbf{R}\|_2$, with α_2 linked to the randomized rounding parameter and where $\|\mathbf{R}\|_2$ can be bounded heuristically by $\sqrt{2nd} + \sqrt{ndk} + t$ for a slack $t \approx 7$ which thus depends on b as $\sqrt{1/\ln(b)}$. The bitsize of a signature is thus

$$|\mathsf{sig}_{\mathsf{MP}}| \approx nd(1/2 + \log_2(\alpha_1 b\|\mathbf{R}\|_2)) + nd\log_b(q)(1/2 + \log_2(\alpha_1 b\|\mathbf{R}\|_2)). \quad (1)$$

For the sampler from Algorithm 3.1, the Gaussian parameter for $\mathbf{v}_1$ is given by $s \approx \alpha_2 b\|\mathbf{R}\|_2\sqrt{ndk}$ where α_2 defines the repetition rate M. As mentioned in Section 3.2.2, the dimension m for LW signatures is chosen to be $m = dk$ instead

[7] For ease of exposition, we simplify the formulas in this paragraph but we stress that the final estimates in Tables 4.1, 4.2 and 4.3 are computed with the exact parameter settings.

of $m = 2d$ for LW^* signatures. The corresponding bitsizes are thus given by

$$|\mathsf{sig}_{\mathsf{LW}}| \approx nd(\log_b(q) - 1)(1/2 + \log_2(\alpha_2 b\|\mathbf{R}\|_2\sqrt{ndk})) + nd\log_2 q \tag{2}$$

$$|\mathsf{sig}_{\mathsf{LW}^*}| \approx nd(1/2 + \log_2(\alpha_2 b\|\mathbf{R}\|_2\sqrt{ndk})) + nd\log_2 q. \tag{3}$$

We already see that the size of $\mathbf{v}_2$, for both LW and LW^*, is $nd\log_2 q$, independently of the choice of b. This is because we can equivalently send $\mathbf{x} \in R_q^d$ instead of $\mathbf{v}_2 = \mathbf{G}^{-1}(\mathbf{x})$. For those two schemes, the dependency in b thence only comes from the first component $\mathbf{v}_{1,2}$.

In the case of LW^*, the situation is simple according to equation 3: the bitsize increases with b, which pleads for small base b. Conversely, the bitsize of LW signatures essentially benefits from large bases b. The same holds true for MP signatures. In the latter two cases, the optimal base therefore seems to be $b = \lceil\sqrt{q}\rceil$ if we consider this sole metric.

We must now evaluate the impact of the base b on the on the underlying computational assumptions. We indeed recall that, in the security proof, one needs to simulate signatures and program the random oracle responses accordingly. To do so, we use the simulation result from Corollary 3.1 (or its equivalent for the original sampling procedure for MP signatures). After that, we simulate the public key and we thus need to consider parameters that ensure the M-LWE$_{n,d,d,q,U(S_1)}$ problem is hard. The security proof is then concluded by a reduction to M-SIS$_{n,d,d(2+k),q,\beta}$ where $\beta \geq \|\mathbf{v} - \mathbf{v}^*\|_2$ for two preimages $\mathbf{v}, \mathbf{v}^*$. It yields $\beta = 2s\sqrt{nd(2+k)}$ for MP signatures, $\beta = 2\sqrt{ndk(s^2 + (b-1)^2)}$ for LW signatures, and $\beta = 2\sqrt{nd(2s^2 + k(b-1)^2)}$ for LW^* signatures.

For MP signatures, the bound β is dominated by the bottom part $\mathbf{v}_2$ as $k \geq 2$. It thus makes sense to increase b in order to reduce the dimension of dk and thus have balanced contributions of $\mathbf{v}_1$ and $\mathbf{v}_2$ to the M-SIS bound β. On the contrary, for LW^* signatures, $\mathbf{v}_1$ and $\mathbf{v}_2$ have essentially the same dimension but the specificity of this sampler leads to a strong asymmetry between them. This rebalances the contributions of $\mathbf{v}_1$ and $\mathbf{v}_2$ in the bound β which is actually already dominated by the former for $b = 2$. In this case, increasing b will only enlarge the gap between the contributions of $\mathbf{v}_1$ and $\mathbf{v}_2$ to the M-SIS bound and thus decrease the security. In parallel, using too large bases such as $b = \sqrt{q}$ impacts the M-SIS bound too drastically, as noted in e.g. [CGM19], and parameters need to be increased to compensate the security accordingly. In particular, one has to ensure that the infinity norm of the M-SIS solution is smaller than q to avoid trivial solutions.

Estimates. This intricate situation is reflected by the estimated performance of a GPV signature that we describe below, for different samplers and parameter constraints. We aim to achieve $\lambda = 128$ bits of security for the GPV signature using the security assessment methodology described in Appendix A. For all the estimates, we fix when necessary the randomized rounding factor $r = 5$, the spectral norm slack $t = 7$, $Q = 2^{40}$ as the maximal number of emitted signatures per key. We then choose the repetition rate $M \approx 11$ which leads to $\alpha \approx 8$ for

$\varepsilon = 1/Q$. We then find the appropriate rank d and modulus q to achieve the security target while minimizing the signature size.

To highlight the importance of the gadget base, we give the performance of MP, LW and LW* signatures for several choices of bases. The estimates are given in Tables 4.1, 4.2, and 4.3. The values of $\lambda_{\text{M-LWE}}$ and $\lambda_{\text{M-SIS}}$ correspond to the reached security of M-LWE$_{n,m-d,d,q,U(S_1)}$ and M-SIS$_{n,d,m+dk,q,\beta}$ respectively. When the base is said to be $q^{1/k}$, we actually consider $b = \lceil q^{1/k} \rceil$ to have an integer base for which the gadget dimension is dk. The optimal sizes and parameters are highlighted in the tables.

	$\lambda_{\text{M-LWE}}$	$\lambda_{\text{M-SIS}}$	q	d	s	$\lvert\mathbf{v}_{1,2}\rvert$	$\lvert\mathbf{v}_2\rvert$	$\lvert\mathsf{sig}_{\mathsf{MP}}\rvert$
$b = 2$	239	146	$\approx 2^{15.2}$	5	2596	1.85	27.75	29.60
$b = 4$	233	150	$\approx 2^{15.6}$	5	3461	1.92	15.32	17.24
$b = q^{1/5}$	216	147	$\approx 2^{16.8}$	5	7661	2.09	10.47	12.56
$b = q^{1/3}$	181	131	$\approx 2^{19.7}$	5	56804	**2.54**	**7.64**	**10.18**
$b = q^{1/2}$	194	154	$\approx 2^{26.7}$	7	6616938	5.07	10.13	15.20

Table 4.1. Parameter and size estimates of MP signatures using different bases b. The sizes are expressed in KB. The ring degree is $n = 256$.

	$\lambda_{\text{M-LWE}}$	$\lambda_{\text{M-SIS}}$	q	d	s	$\lvert\mathbf{v}_{1,2}\rvert$	$\lvert\mathbf{v}_2\rvert$	$\lvert\mathsf{sig}_{\mathsf{LW}}\rvert$
$b = 2$	> 1000	131	$\approx 2^{23.6}$	6	572109	80.96	4.50	86.46
$b = 4$	> 1000	130	$\approx 2^{23.8}$	6	901768	41.83	4.50	46.33
$b = q^{1/5}$	597	130	$\approx 2^{27.3}$	6	5586865	17.19	5.25	22.44
$b = q^{1/3}$	428	133	$\approx 2^{30.6}$	7	105308864	**11.88**	**6.78**	**18.66**
$b = q^{1/2}$	161	138	$\approx 2^{40.5}$	9	96061795597	10.40	11.53	21.93

Table 4.2. Parameter and size estimates of LW signatures (with the parameter constraints of [LW15]). The sizes are expressed in KB. The ring degree is $n = 256$. We note that we extrapolated the result of [LW15] which is only presented for $b = 2$. In particular, the parameters we give for $b = q^{1/3}$ and $b = q^{1/2}$ do not perfectly meet the regularity condition from their paper, namely $ndk \log_2 s > 3nd \log_2 q + 4\lambda$. For low values of k, one would need to increase s but it would also lead to increasing q to compensate the security loss.

These estimates show that the choice of the base is far from anecdotal, with a 3-4 ratio for the signature size between the best option and the worst one. They also show that there is no generic choice as $b = 2$ is optimal in our case (LW*) whereas it corresponds to the worst case for both MP and LW. When plugged into

	$\lambda_{\text{M-LWE}}$	$\lambda_{\text{M-SIS}}$	q	d	s	$\lvert\mathbf{v}_{1,2}\rvert$	$\lvert\mathbf{v}_2\rvert$	$\lvert\mathsf{sig}_{\mathsf{LW}^*}\rvert$
$b = 2$	195	157	$\approx 2^{22.5}$	6	362140	**3.56**	**4.31**	**7.87**
$b = 4$	188	151	$\approx 2^{23.2}$	6	645772	3.71	4.50	8.21
$b = q^{1/5}$	167	134	$\approx 2^{25.6}$	6	3576993	4.18	4.87	9.05
$b = q^{1/3}$	167	137	$\approx 2^{30.3}$	7	90206170	5.89	6.78	12.67
$b = q^{1/2}$	162	138	$\approx 2^{40.3}$	9	90202905475	10.37	11.53	21.90

Table 4.3. Parameter and size estimates of LW^* signatures (this work) using different bases b. The sizes are expressed in KB. The ring degree is $n = 256$.

other signature designs [DM14,BFRS18,dPLS18,BEP+21,LNPS21,LNP22b], the conclusions may differ as the relative contributions of $\mathbf{v}_1$ and $\mathbf{v}_2$ to the M-SIS bound may evolve compared to the case of GPV signature.

Besides this sole consideration of optimal base, these tables clearly show the benefits of the LW* sampler as it yields signatures that are about 60% smaller than those produced with the LW sampler and 25% smaller when compared with the MP sampler. It thus shows that one can indeed leverage rejection sampling to improve MP sampling, which solves the apparent paradox of the original LW sampler.

5 Approximate Rejection Sampler

In sections 3 and 4, we revisited the original LW sampler, showing that it could outperform the MP sampler thanks to our new analysis. However, when plugged in the GPV framework, one still ends up with signature sizes that are much larger than the state-of-the-art.

Fortunately, a study initiated by Chen, Genise and Mukherjee [CGM19] improves the performance of gadget-based constructions through the notion of approximate trapdoors. The idea is to drop the low-order gadget entries and only consider a partial gadget $\mathbf{G}_H = \mathbf{I}_d \otimes \mathbf{g}_H^T$, with $\mathbf{g}_H = [b^\ell | \dots | b^{k-1}]^T$, which reduces the signature dimension and the number of columns in the trapdoor $\mathbf{R}$. Obviously, this introduces an error in the preimage which depends on ℓ but also on the specificities of the sampler. In [CGM19], which is based on the MP sampler, the authors generate normally (see Section 3.1) $\mathbf{z} \sim \mathcal{D}_{\mathcal{L}_q^{\mathbf{x}}(\mathbf{G}), s_{\mathbf{G}}}$ for the full gadget matrix $\mathbf{G}$ and some appropriate vector $\mathbf{x}$ and then drop the component $\mathbf{z}_L$ of $\mathbf{z}$ corresponding to $\mathbf{G}_L = \mathbf{I}_d \otimes \mathbf{g}_L^T$, with $\mathbf{g}_L = [1 | \dots | b^{\ell-1}]^T$. This leads to a Gaussian error $\mathbf{e} = \mathbf{G}_L \mathbf{z}_L$ whose infinity norm is likely to be larger than $b^\ell - 1$, which does not seem optimal.

Conversely, in the case of the LW* sampler, $\mathbf{z}$ is exactly $\mathbf{G}^{-1}(\mathbf{w})$ for some syndrome $\mathbf{w}$. Put differently, $\mathbf{z}$ is simply the signed base-b decomposition of $\mathbf{w}$. Applying the approximate trapdoor approach in our case then essentially consists in discarding the lower-order entries $\mathbf{z}_L$ of this decomposition, which leads to an error $\mathbf{e} = \mathbf{G}_L \mathbf{z}_L$, with $\|\mathbf{e}\|_\infty < b^\ell$. Actually, we show afterwards

that this error is (almost) uniform over a subset of $S_{b^\ell-1}$, which also improves the bound on $\|\mathbf{e}\|_2$. This smaller error, having a similar behaviour than the one in [YJW23], allows for dropping more entries than in [CGM19], leading to better performance. In our scheme in Section 6, we can in particular drop $\ell = k-1$ entries, yielding a gadget of length 1 as in [YJW23].

The formal description of our approximate sampler is provided in Algorithm 5.1.

Algorithm 5.1: Approx.SamplePreRej($\mathbf{R}; \mathbf{A}', \mathbf{u}, \mathscr{D}_s, \mathscr{D}_t$)

Input: Trapdoor $\mathbf{R} \in R^{2d\times d(k-\ell)}$, Matrix $\mathbf{A}' \in R_q^{d\times d}$, Syndrome $\mathbf{u} \in R_q^d$, Source and target distributions $\mathscr{D}_s$ and $\mathscr{D}_t$ over R^{2d} such that rejection sampling can be performed.

1. $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$.
2. $\mathbf{w} \leftarrow \mathbf{u} - [\mathbf{I}_d|\mathbf{A}']\mathbf{p}_1 \bmod qR$. ▷ Syndrome correction
3. $\mathbf{z} \leftarrow \mathbf{G}^{-1}(\mathbf{w}) \in S_{b-1}^{dk}$. ▷ Deterministic.
4. Parse $\mathbf{z}$ into $\mathbf{z}_L \in S_{b-1}^{d\ell}$ and $\mathbf{z}_H \in S_{b-1}^{d(k-\ell)}$ so that $\mathbf{G}\mathbf{z} = \mathbf{G}_L\mathbf{z}_L + \mathbf{G}_H\mathbf{z}_H$.
5. $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{R}\mathbf{z}_H$.
6. $u \hookleftarrow U([0,1])$ ▷ Continuous
7. **if** $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M\mathscr{D}_s(\mathbf{p}_1)))$, go back to 1.
8. **else** $\mathbf{v}_1 \leftarrow \mathbf{v}_1' + \begin{bmatrix}\mathbf{G}_L\mathbf{z}_L \\ \mathbf{0}\end{bmatrix}$
9. $\mathbf{v}_2 \leftarrow \mathbf{z}_H$

Output: $\mathbf{v} = \begin{bmatrix}\mathbf{v}_1 \\ \mathbf{v}_2\end{bmatrix}$.

Lemma 5.1 (Correctness). *For all $\mathbf{A}' \in R_q^{d\times d}$, $\mathbf{R} \in R^{2d\times(k-\ell)d}$, $\mathbf{u} \in R_q^d$, $\mathscr{D}_s, \mathscr{D}_t$, and $\mathbf{v} \leftarrow$ Approx.SamplePreRej$(\mathbf{R}, \mathbf{A}', \mathbf{u}, \mathscr{D}_s, \mathscr{D}_t)$, it holds that the preimage verifies $\mathbf{v} \in \mathcal{L}_q^{\mathbf{u}}([\mathbf{I}_d|\mathbf{A}'|\mathbf{G}_H - \mathbf{B}])$, where $\mathbf{G}_H = \mathbf{I}_d \otimes [b^\ell|\dots|b^{k-1}]$ and $\mathbf{B} = [\mathbf{I}_d|\mathbf{A}']\mathbf{R} \bmod qR$.*

Proof. We indeed have

$$\begin{aligned}[\mathbf{I}_d|\mathbf{A}'|\mathbf{G}_H - \mathbf{B}]\mathbf{v} &= [\mathbf{I}_d|\mathbf{A}'](\mathbf{p}_1 + \mathbf{R}\mathbf{z}_H) + \mathbf{G}_L\mathbf{z}_L + (\mathbf{G}_H - \mathbf{B})\mathbf{z}_H \\ &= (\mathbf{u} - \mathbf{w}) + \mathbf{B}\mathbf{z}_H + \mathbf{G}_L\mathbf{z}_L + (\mathbf{G}_H - \mathbf{B})\mathbf{z}_H \bmod qR \\ &= (\mathbf{u} - \mathbf{G}\mathbf{z}) + \mathbf{G}_L\mathbf{z}_L + \mathbf{G}_H\mathbf{z}_H \bmod qR \\ &= \mathbf{u} \bmod qR.\end{aligned}$$

□

For completeness, we also need to adapt the Theorem 3.1 on the simulatability of preimages. The proof is very similar to that of the exact version of the sampler but is included for completeness. We slightly abuse notations and denote by $\mathbf{G}_H^{-1}$ (resp. $\mathbf{G}_L^{-1}$) the map that from $\mathbf{w}$ computes $\mathbf{z} = \mathbf{G}^{-1}(\mathbf{w})$ and outputs the vector $\mathbf{z}_H$ (resp. $\mathbf{z}_L$) defined above. We nevertheless recall that $\mathbf{G}_L\mathbf{G}_L^{-1}(\mathbf{w}) = \mathbf{w}$ only holds for some vectors $\mathbf{w}$ and not in general. We also note that $\mathbf{G}_H^{-1}(R_q^d) \subset S_{b-1}^{d(k-\ell)}$ but equality does not hold simply by a counting argument.

Theorem 5.1. *Let K be a number field and R its ring of integers. Let d, q, b be positive integers with $b \geq 2$. We define $k = \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$ and let $\ell \in [0, k-1]$. Let $\mathscr{D}_r, \mathscr{D}_s, \mathscr{D}_t$ be three distributions over $R^{2d \times d(k-\ell)}$, R^{2d} and R^{2d} respectively. Let $\mathbf{A}' \sim U(R_q^{d \times d})$, $\mathbf{R} \sim \mathscr{D}_r$ and $\mathbf{A} = [\mathbf{I}_d | \mathbf{A}'] \in R_q^{d \times 2d}$. Then, let $Y \subseteq R^{2d}$ be the support of the distribution of $\mathbf{R} \cdot \mathbf{G}_H^{-1}(U(R_q^d))$. Let $M > 1, \varepsilon \in [0, 1/2]$ such that $\max_{\mathbf{Rz}_H \in Y} RD_\infty^\varepsilon(\mathscr{D}_t \| \mathscr{D}_s^{+\mathbf{Rz}_H}) \leq M$. We also define the error distribution $\mathscr{D}_e = \mathbf{G}_L \mathbf{G}_L^{-1}(U(R_q^d))$ over $S_{b^\ell - 1}^d$. We then define two distributions*

$\mathcal{P}_1$

$\mathbf{u} \hookleftarrow U(R_q^d)$, *and* $\mathbf{v} \leftarrow \mathsf{Approx.SamplePreRej}(\mathbf{R}; \mathbf{A}', \mathbf{u}, \mathscr{D}_s, \mathscr{D}_t)$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{P}_2$

1. $\mathbf{v}_1' \hookleftarrow \mathscr{D}_t$, $\mathbf{v}_2 \hookleftarrow \mathbf{G}_H^{-1}(U(R_q^d))$, $\mathbf{e} \hookleftarrow \mathscr{D}_e$.
2. $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [\mathbf{e}^T | \mathbf{0}] | \mathbf{v}_2^T]^T$.
3. $\mathbf{u} \leftarrow [\mathbf{A} | \mathbf{G}_H - \mathbf{AR}]\mathbf{v} \bmod qR$.
4. *With probability* $1 - 1/M$ *go back to* 1.

Output: $(\mathbf{v}, \mathbf{u})$.

Then, $\Delta(\mathcal{P}_0, \mathcal{P}_1) \leq \varepsilon$ and for all $a \in (1, +\infty]$, $RD_a(\mathcal{P}_0 \| \mathcal{P}_1) \leq 1/(1-\varepsilon)^{a/(a-1)}$.

Proof. We define the following hybrid distributions from $\mathcal{H}_1$ to $\mathcal{H}_6$, where $\mathcal{H}_1 = \mathcal{P}_1$ and $\mathcal{H}_6 = \mathcal{P}_2$.

$\mathcal{H}_1$

$\mathbf{u} \hookleftarrow U(R_q^d)$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{w} \leftarrow \mathbf{u} - \mathbf{Ap}_1 \bmod qR$, $\mathbf{z} \leftarrow \mathbf{G}^{-1}(\mathbf{w})$, $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{Rz}_H$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [(\mathbf{G}_L \mathbf{z}_L)^T | \mathbf{0}] | \mathbf{z}_H^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_2$

$\mathbf{w} \hookleftarrow U(R_q^d)$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{u} \leftarrow \mathbf{w} + \mathbf{Ap}_1 \bmod qR$, $\mathbf{z} \leftarrow \mathbf{G}^{-1}(\mathbf{w})$, $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{Rz}_H$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [(\mathbf{G}_L \mathbf{z}_L)^T | \mathbf{0}] | \mathbf{z}_H^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_3$

$\mathbf{z} \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$, $\mathbf{w} \leftarrow \mathbf{Gz} \bmod qR$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{u} \leftarrow \mathbf{w} + \mathbf{Ap}_1 \bmod qR$, $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{Rz}_H$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [(\mathbf{G}_L \mathbf{z}_L)^T | \mathbf{0}] | \mathbf{z}_H^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_4$

$\mathbf{e} \hookleftarrow \mathscr{D}_e$, $\mathbf{z}_H \leftarrow \mathbf{G}_H^{-1}(U(R_q^d))$, $\mathbf{w} \leftarrow \mathbf{e} + \mathbf{G}_H \mathbf{z}_H \bmod qR$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{u} \leftarrow \mathbf{w} + \mathbf{Ap}_1 \bmod qR$, $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{Rz}_H$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [\mathbf{e}^T | \mathbf{0}] | \mathbf{z}_H^T]^T$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_5$

$\mathbf{e} \hookleftarrow \mathscr{D}_e$, $\mathbf{z}_H \leftarrow \mathbf{G}_H^{-1}(U(R_q^d))$, $\mathbf{p}_1 \hookleftarrow \mathscr{D}_s$, $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{Rz}_H$. Then $u \hookleftarrow U([0,1])$ and restart if $u > \min(1, \mathscr{D}_t(\mathbf{v}_1')/(M \cdot \mathscr{D}_s(\mathbf{p}_1)))$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [\mathbf{e}^T | \mathbf{0}] | \mathbf{z}_H^T]^T$, and $\mathbf{u} \leftarrow [\mathbf{A} | \mathbf{G}_H - \mathbf{AR}]\mathbf{v} \bmod qR$.
Output: $(\mathbf{v}, \mathbf{u})$.

$\mathcal{H}_6$

$\mathbf{e} \hookleftarrow \mathscr{D}_e$, $\mathbf{z}_H \leftarrow \mathbf{G}_H^{-1}(U(R_q^d))$, $\mathbf{v}_1' \hookleftarrow \mathscr{D}_t$. Then $u \hookleftarrow U([0,1])$ and restart if $u > 1 - 1/M$. Otherwise define $\mathbf{v} \leftarrow [\mathbf{v}_1'^T + [\mathbf{e}^T | \mathbf{0}] | \mathbf{z}_H^T]^T$, and $\mathbf{u} \leftarrow [\mathbf{A} | \mathbf{G}_H - \mathbf{AR}]\mathbf{v} \bmod qR$.

Output: $(\mathbf{v}, \mathbf{u})$.

Let us now show that these distributions are statistically close to each other.

$\mathcal{H}_1$ - $\mathcal{H}_2$: Here we just change the sampling order of $\mathbf{u}$ and $\mathbf{w}$. In $\mathcal{H}_2$ the vector $\mathbf{w}$ is uniform and independent of $\mathbf{A}\mathbf{p}_1$ implying that $\mathbf{u}$ is also uniform, as in $\mathcal{H}_1$. Hence $\mathcal{H}_1$ and $\mathcal{H}_2$ are identically distributed.

$\mathcal{H}_2$ - $\mathcal{H}_3$: We now change the way $\mathbf{w}$ is generated. Notice that, for correctness, once $\mathbf{w}$ is fixed then so is $\mathbf{z}$ and vice-versa. In $\mathcal{H}_2$, $\mathbf{w}$ is uniform over R_q^d which means that $\mathbf{z}$ follows exactly $\mathbf{G}^{-1}(U(R_q^d))$ as in $\mathcal{H}_3$. Also, $\mathbf{w}$ is coherently set in $\mathcal{H}_3$. Thence, $\mathcal{H}_2$ and $\mathcal{H}_3$ are identically distributed as well.

$\mathcal{H}_3$ - $\mathcal{H}_4$: $\mathcal{H}_3$ simply separates the sampling of low-order and high-order parts compared to $\mathcal{H}_3$. When $\mathbf{z}$ is drawn from $\mathbf{G}^{-1}(U(R_q^d))$, the corresponding $\mathbf{z}_L$ and $\mathbf{z}_H$ are independent. So $\mathbf{z}_H \sim \mathbf{G}_H^{-1}(U(R_q^d))$ and $\mathbf{z}_L \sim \mathbf{G}_L^{-1}(U(R_q^d))$. As such, $\mathbf{z}_H$ is identically distributed in $\mathcal{H}_3$ as in $\mathcal{H}_2$ by definition of $\mathbf{G}_H^{-1}$ which samples a whole vector and drops the low-order entries. Since $\mathbf{z}_L$ is not directly used but only as $\mathbf{e} = \mathbf{G}_L\mathbf{z}_L$, and because $\mathbf{z}_L \sim \mathbf{G}_L^{-1}(U(R_q^d))$ in both $\mathcal{H}_3$ and $\mathcal{H}_4$, it holds that $\mathbf{e} \sim \mathbf{G}_L\mathbf{G}_L^{-1}(U(R_q^d)) = \mathscr{D}_e$ in both hybrids. The way $\mathbf{z}_L$ is sampled, recomposing the low-order entries gives $\mathbf{e} \in S_\gamma^d$ where $\gamma = \sum_{i=0}^{\ell-1}(b-1)b^i = b^\ell - 1$, as desired. This shows that $\mathcal{H}_3$ and $\mathcal{H}_4$ are identically distributed.

$\mathcal{H}_4$ - $\mathcal{H}_5$: $\mathcal{H}_5$ is merely a re-writing of $\mathcal{H}_4$. Indeed, in $\mathcal{H}_4$, $\mathbf{w}$ only acts as an intermediate vector to define $\mathbf{u}$. Defining $\mathbf{R}' = [\mathbf{R}^T|\mathbf{I}_{m_2}]^T$, we have $[\mathbf{A}|\mathbf{G}_H - \mathbf{A}\mathbf{R}]\mathbf{R}' = \mathbf{G}_H \bmod qR$. In $\mathcal{H}_3$, this yields

$$\begin{aligned}\mathbf{u} = \mathbf{G}_H\mathbf{z}_H + \mathbf{e} + \mathbf{A}\mathbf{p}_1 \bmod qR &= [\mathbf{A}|\mathbf{G}_H - \mathbf{A}\mathbf{R}]\mathbf{R}'\mathbf{z}_H + \mathbf{A}\mathbf{p}_1 + \mathbf{e} \bmod qR \\ &= [\mathbf{A}|\mathbf{G}_H - \mathbf{A}\mathbf{R}]\mathbf{v} \bmod qR,\end{aligned}$$

as $\mathbf{v} = [\mathbf{p}_1^T + [\mathbf{e}^T|\mathbf{0}]|\mathbf{0}]^T + \mathbf{R}'\mathbf{z}_H$. Again, $\mathcal{H}_4$ and $\mathcal{H}_5$ are identical.

$\mathcal{H}_5$ - $\mathcal{H}_6$: We now change the way $\mathbf{v}_1'$ is generated by using the rejection sampling result. In $\mathcal{H}_5$, $\mathbf{R}\mathbf{z}_H$ is distributed according to $\mathbf{R} \cdot \mathbf{G}_H^{-1}(U(R_q^d))$ with support Y as defined in the theorem statement. By our assumptions on Y, $\mathscr{D}_s$, $\mathscr{D}_t$, the rejection sampling result from Lemma 2.2 yields that

$$\Delta((\mathbf{v}_1', \mathbf{z}_H)_{\mathcal{H}_5}, (\mathbf{v}_1', \mathbf{z}_H)_{\mathcal{H}_6}) \leq \varepsilon \quad \text{and} \quad RD_a((\mathbf{v}_1', \mathbf{z}_H)_{\mathcal{H}_5} \| (\mathbf{v}_1', \mathbf{z}_H)_{\mathcal{H}_6}) \leq \frac{1}{(1-\varepsilon)^{\frac{a}{a-1}}},$$

for all $a > 1$. By the data processing inequality of the statistical distance and Rényi divergence, it holds

$$\Delta(\mathcal{H}_5, \mathcal{H}_6) \leq \varepsilon \quad \text{and} \quad RD_a(\mathcal{H}_5 \| \mathcal{H}_6) \leq \frac{1}{(1-\varepsilon)^{\frac{a}{a-1}}}.$$

Since $\mathcal{H}_1 = \mathcal{P}_1$ and $\mathcal{H}_6 = \mathcal{P}_2$, combining the above gives the result. □

The study carried in Section 4 leads to the same conclusions for the approximate samplers, although the analysis is slightly more complex as one can optimize over the number of dropped entries ℓ as well. Because the sampling

error $\mathbf{e}$ is smaller in our case, we can drop more entries and thus increase the performance gap between the approximate MP sampler and ours. In particular, we observe an improvement in the signature size of around $30 - 35\%$ over the former, for the same estimation methodology as Section 4. The signature sizes are now more attractive, but we push the performance in Section 6 by providing a new hash-and-sign scheme based on our approach.

6 A New Hash-and-Sign Scheme: Phoenix

In our quest to test the limits of the LW sampler, we plug our approximate LW* sampler described above in the GPV framework to build a new signature scheme which we call Phoenix. The optimal base for our approximate sampler is also $b = 2$ and we thus we express everything in base 2 directly. Also, we choose the modulus to be $q = 2^{k+1} - 1$ so that the representatives of $\mathbb{Z}_q$ are taken in the centered interval $[-(q-1)/2, (q-1)/2] = [-(2^k - 1), 2^k - 1]$. The resulting gadget dimension is $\lceil \log_2(\lceil (q-1)/2 \rceil + 1) \rceil = k$. We start by presenting in Section 6.1 the public key compression technique we apply for Phoenix, before giving the full description of the scheme in Section 6.2. We then detail the security analysis in Section 6.3 and give concrete instantiations in Section 6.4.

6.1 Adding Public Key Compression

The approximate sampler already enables a significant compression of both the public key and the signature. However, in the context of hash-and-sign signatures following the GPV framework, the public key[8] is $\mathbf{B} = [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR \in R_q^{d \times d(k-\ell)}$ which remains quite large for typical parameters. Fortunately, the features of our sampler allows us to use standard techniques for public key compression, like the one used in [DKL+18,DSH21] for example. As this is a standard trick, we only sketch the idea and explain the impact it has on the construction.

Let ℓ' be a positive integer in $[0, k-1]$. Once the public key $\mathbf{B}$ has been generated, we interpret it in R as a matrix over $S_{(q-1)/2}$. We then write it as $\mathbf{B} = \mathbf{B}_L + \mathbf{B}_H$ to separate the low-order and high-order bits, with $\mathbf{B}_L \in S_{2^{\ell'}-1}^{d \times d(k-\ell)}$ and $\mathbf{B}_H \in 2^{\ell'} S_\gamma^{d \times d(k-\ell)}$ for $\gamma = \left\lfloor 2^{-\ell'} \frac{q-1}{2} \right\rfloor = 2^{k-\ell'} - 1$. The public key now only consists of the matrix $\mathbf{B}_H$ (or $2^{-\ell'}\mathbf{B}_H$) which can be stored using $nd^2(k-\ell)(1+k-\ell')$ bits, thus saving ℓ' bits per coefficients.

The discarded low-order bits then introduce a new error $\mathbf{e}_{\mathsf{pk}} = \mathbf{B}_L \mathbf{v}_2 = \mathbf{B}_L \mathbf{z}_H$ in the signature. This error can be combined with the sampling error $\mathbf{e}$ during preimage sampling.

The reason why this compression technique is particularly interesting in our situation, as opposed to Eagle [YJW23] for example, is because $\mathbf{z}_H$ is ternary and not Gaussian. As such, the error $\mathbf{e}_{\mathsf{pk}}$ remains moderate compared to $\mathbf{e}$ if $\ell' \leq \ell$, as detailed below.

[8] The other public key matrix $\mathbf{A}'$ is generated using a public seed of 256 bits.

6.2 Description

We give an instantiation of Phoenix based on discrete Gaussian distributions. We also defer in Appendix B another version of the scheme using only uniform distributions.

Algorithm 6.1: Phoenix.Setup

Input: Security parameter λ.

1. Choose positive integers d, k.
2. $q \leftarrow 2^{k+1} - 1$.
3. Choose $\ell, \ell' \in [0, k-1]$.
4. $\mathbf{G} = \mathbf{I}_d \otimes [1 \cdots 2^{k-1}] \in R_q^{d \times dk}$.
5. $\mathbf{G}_H = \mathbf{I}_d \otimes [2^{\ell} \cdots 2^{k-1}] \in R_q^{d \times d(k-\ell)}$.
6. $\mathbf{G}_L = \mathbf{I}_d \otimes [1 \cdots 2^{\ell-1}] \in R_q^{d \times d\ell}$.
7. $\varepsilon \leftarrow 1/4Q$ ▷ Rejection sampling loss
8. Choose $M > 1$. ▷ Repetition rate
9. $\alpha \leftarrow \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}})$. ▷ Rejection sampling slack
10. $s \leftarrow \alpha\sqrt{nd(k-\ell)}(\sqrt{2nd} + \sqrt{nd(k-\ell)})$. ▷ Gaussian width
11. $\mathbf{A}' \hookleftarrow U(R_q^{d \times d})$.

Output: $\mathsf{pp} = (\mathbf{A}'; \mathbf{G}, \mathbf{G}_L, \mathbf{G}_H; \lambda, n, q, d, k, \ell, s, M)$.

Algorithm 6.2: Phoenix.KeyGen

Input: Public parameters pp as in Algorithm 6.1.

1. $\mathbf{R} \hookleftarrow U(S_1^{2d \times d(k-\ell)})$ such that $\|\mathbf{R}\|_2 \leq \sqrt{2nd} + \sqrt{nd(k-\ell)}$.
2. $\mathbf{B} \leftarrow [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR \in R_q^{d \times d(k-\ell)}$
3. Parse $\mathbf{B}$ as $\mathbf{B}_L + \mathbf{B}_H$ with $\mathbf{B}_L \in S_{2^{\ell'}-1}^{d \times d(k-\ell)}$ and $\mathbf{B}_H \in 2^{\ell'} S_{2^{k-\ell'}-1}^{d \times d(k-\ell)}$.

Output: $\mathsf{pk} = \mathbf{B}_H$, and $\mathsf{sk} = \mathbf{R}$. ▷ pp stored with pk for simplicity

Algorithm 6.3: Phoenix.Sign

Input: Secret key sk, Message $\mathbf{m} \in \{0,1\}^*$, Public key pk.

1. $\mathsf{salt} \hookleftarrow U(\{0,1\}^{320})$.
2. $(\widetilde{\mathbf{v}_1}, \mathbf{v}_2) \leftarrow \mathsf{Approx.SamplePreRej}(\mathbf{R}; \mathbf{A}', \mathcal{H}(\mathbf{m}, \mathsf{salt}), s)$. ▷ Algorithm 5.1
3. $\mathbf{e}_{\mathsf{pk}} \leftarrow (([\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR) - \mathbf{B}_H)\mathbf{v}_2$. ▷ Recomputing $\mathbf{B}_L$, and $\mathbf{B}_L\mathbf{z}_H$
4. Parse $\widetilde{\mathbf{v}_1}$ as $[\widetilde{\mathbf{v}_{1,1}}^T | \mathbf{v}_{1,2}^T]^T$ with $\widetilde{\mathbf{v}_{1,1}}, \mathbf{v}_{1,2} \in R^d$.
5. $\mathbf{v}_{1,1} \leftarrow \widetilde{\mathbf{v}_{1,1}} - \mathbf{e}_{\mathsf{pk}}$.
6. $\gamma_1 \leftarrow (\|\mathbf{v}_{1,1}\|_2 \leq B_{1,1}) \wedge (\|\mathbf{v}_{1,2}\|_2 \leq B_{1,2})$.
7. $\gamma_2 \leftarrow (\|\mathbf{v}_{1,1}\|_\infty \leq B_{1,1}^\infty) \wedge (\|\mathbf{v}_{1,2}\|_\infty \leq B_{1,2}^\infty) \wedge (\|\mathbf{v}_2\|_\infty \leq 1)$.
8. **if** $\gamma_1 \wedge \gamma_2 = 0$, restart.

Output: $\mathsf{sig} = (\mathsf{salt}, \mathbf{v}_{1,2}, \mathbf{v}_2)$.

Algorithm 6.4: Phoenix.Verify

Input: Public key pk, Message $\mathbf{m} \in \{0,1\}^*$, Signature sig.

1. $\mathbf{v}_{1,1} \leftarrow \mathcal{H}(\mathbf{m}, \mathsf{salt}) - \mathbf{A}'\mathbf{v}_{1,2} - (\mathbf{G}_H - \mathbf{B}_H)\mathbf{v}_2 \bmod qR \in R^d$.
2. $\gamma_1 \leftarrow (\|\mathbf{v}_{1,1}\|_2 \leq B_{1,1}) \wedge (\|\mathbf{v}_{1,2}\|_2 \leq B_{1,2})$.
3. $\gamma_2 \leftarrow (\|\mathbf{v}_{1,1}\|_\infty \leq B_{1,1}^\infty) \wedge (\|\mathbf{v}_{1,2}\|_\infty \leq B_{1,2}^\infty) \wedge (\|\mathbf{v}_2\|_\infty \leq 1)$.

Output: $\gamma_1 \wedge \gamma_2$. ▷ 1 if valid, 0 otherwise

Preimage error distribution. Let us define a modified error distribution $\mathscr{D}_e^+$ where we sample $\mathbf{e} \hookleftarrow \mathscr{D}_e$ and output $\mathbf{e}^+$ corresponding to $\mathbf{e}$ but where the coefficient embeddings of $\mathbf{e}^+$ are the magnitude of that of $\mathbf{e}$. We observe that $\mathscr{D}_e^+$ is almost the uniform distribution over $\tau^{-1}(\{0, \ldots, 2^\ell - 1\}^{nd})$ because of the

form of q. This means that the Euclidean norm of $\mathbf{e}$ will be distributed the same way as that of $\mathbf{e}^+$. The variance of $U(\{0,\dots,2^\ell-1\})$ is exactly $(2^{2\ell}-1)/12$ and the norm can be bounded on average by $\sqrt{(2^\ell-1))(2^{\ell+1}-1)/6}\sqrt{nd}$ by the central limit theorem.

Verification bounds. We now explain how the verification bounds are set. First, the ones on $\mathbf{v}_{1,2}$ are simply taken from Lemma 2.1 by adjusting the slack to avoid too many repetitions. As such we set $B_{1,2} = 1.048 \cdot s\sqrt{nd/2\pi}$ and $B^\infty_{1,2} = \lfloor 4.6s/\sqrt{2\pi} \rceil$.

Choosing appropriate bounds is more complex for $\mathbf{v}_{1,1}$ because the value recovered by the verifier is $\mathbf{v}'_{1,1}+\mathbf{e}-\mathbf{e}_{\mathsf{pk}}$ which contains the error terms. Bounding each term separately overshoots the actual norm of $\mathbf{v}_{1,1}$. We thus give a more fine-grained analysis based on the following observations. We first notice that the coefficients of $\mathbf{e}_{\mathsf{pk}} = \mathbf{B}_L\mathbf{v}_2$ behave in a similar fashion to the drift of lazy random walks with adaptive steps whose magnitude are at most $2^{\ell'}-1$, up to a slack factor μ depending on the conductor of the cyclotomic field[9]. As such, we can approach the bounds on $\mathbf{v}_{1,1} - \mathbf{e}_{\mathsf{pk}}$ by the Gaussian tail bound with the appropriate variance. Then, $\|\mathbf{e}\|_2$ can be evaluated as described above which also behaves like the Gaussian tail bound, and $\|\mathbf{e}\|_\infty$ is very likely to be close to the worst-case bound $2^\ell - 1$. Using these Gaussian approximations, we set

$$B_{1,1} = 1.04\sqrt{\frac{s^2}{2\pi} + \frac{(2^\ell-1)(2^{\ell+1}-1)}{6} + \mu^2\frac{2^{\ell'}(2^{\ell'}-1)}{6}\frac{nd(k-\ell)}{2}}\sqrt{nd}$$
$$B^\infty_{1,1} = \left\lfloor 3.8 \cdot \sqrt{\frac{s^2}{2\pi} + \mu^2\frac{2^{\ell'}(2^{\ell'}-1)}{6}\frac{nd(k-\ell)}{2}} \right\rfloor + (2^\ell - 1).$$

which are verified empirically and only entail a small degradation of the average number of repetition M. The term in $nd(k-\ell)/2$ stems from the contribution of $\mathbf{e}_{\mathsf{pk}}$, and naturally comes from average number of steps in the lazy random walk due to the Hamming weight of $\tau(\mathbf{v}_2)$. As a result, choosing $\ell' \approx \ell$ would not be optimal because it would essentially make $\mathbf{e}_{\mathsf{pk}}$ larger than $\mathbf{e}$ as $\mathbf{e}_{\mathsf{pk}}$ grows faster with ℓ' than $\mathbf{e}$ does with ℓ. For common parameters (see Section 6.4), where ℓ is close to k[10], choosing $\ell' \approx (k+1)/2$ seems to be the best option as it halves the public key size while incurring (almost) no security loss. This is because for such parameters $\mathbf{e}_{\mathsf{pk}}$ is overpowered by the preimage error $\mathbf{e}$.

Remark 6.1. Phoenix shares with [ETWY22] the goal of moving the bulk of the preimage in $\mathbf{v}_{1,1}$ which is not transmitted. Our treatment is howbeit very different from the twisted norm approach of the latter work. This gives further evidence of the benefits of the asymmetry in concrete lattice-based cryptography.

[9] This slack comes from the multiplication $M_\tau(\mathbf{B}_L)\tau(\mathbf{v}_2)$ in the coefficient embedding. Later we choose 3-smooth conductors yielding $\mu = \sqrt{2}$, and $\mu = 1$ for power-of-two conductors.

[10] Choosing $\ell = k-2$ or $\ell = k-1$ is possible as opposed to the approach in [CGM19] because $\mathbf{e}$ is smaller by a factor of $\sqrt{3}\omega(\sqrt{\log_2 nd})$.

6.3 Security Analysis

Our scheme follows the GPV framework. One can thus use the simulation result of Theorem 5.1 adapted to Phoenix, which we provide here.

Corollary 6.1. *Let d, k be positive integers, define $q = 2^{k+1} - 1$ and let $\ell \in [0, k-1]$. Let $T = \sqrt{nd(k-\ell)}(\sqrt{2nd} + \sqrt{nd(k-\ell)})$. Let $M > 1$, $\varepsilon \in (0, 1/2]$ and define $\alpha = \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}})$. Finally $s = \alpha T$. Let $\mathbf{A}' \sim U(R_q^{d \times d})$, $\mathbf{R} \sim U(S_1^{2d \times d(k-\ell)})$ conditioned on $\|\mathbf{R}\|_2 \leq \sqrt{2nd} + \sqrt{nd(k-\ell)}$. We define $\mathcal{P}_1$ and $\mathcal{P}_2$ the same way as in Theorem 5.1 but where $\mathscr{D}_s, \mathscr{D}_t$ are replaced with $\mathcal{D}_{R^{2d},s}$. Then, it holds that $\Delta(\mathcal{P}_1, \mathcal{P}_2) \leq \varepsilon$ and $RD_a(\mathcal{P}_1 \| \mathcal{P}_2) \leq 1/(1-\varepsilon)^{a/(a-1)}$ for all $a \in (1, +\infty]$.*

Proof. We simply have to verify that the smooth Rényi divergence condition of Theorem 5.1 holds. In our context, we restrict the matrices $\mathbf{R}$ to have a bounded spectral norm. Following the notations of Theorem 5.1, the distribution $\mathscr{D}_r$ consists in sampling $\mathbf{R}$ from $U(S_1^{2d \times d(k-\ell)})$ such that $\|\mathbf{R}\|_2 \leq B$, where $B = \sqrt{2nd} + \sqrt{nd(k-\ell)}$. The set Y is the support of $\mathbf{R} \cdot \mathbf{G}_H^{-1}(U(R_q^d))$. Hence, for all $\mathbf{R}\mathbf{z}_H$ in Y, we have $\|\mathbf{R}\mathbf{z}_H\|_2 \leq \|\mathbf{R}\|_2 \|\mathbf{z}_H\|_2 \leq B \cdot \sqrt{nd(k-\ell)} = T$. We note that since $Y \subset R^{2d}$, we have $\mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{z}_H} = \mathcal{D}_{R^{2d},s,\mathbf{R}\mathbf{z}_H}$ for all $\mathbf{R}\mathbf{z}_H \in Y$. Using Lemma 2.3, it thus holds that

$$\begin{aligned} RD_\infty^\varepsilon(\mathcal{D}_{R^{2d},s} \| \mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{z}_H}) &\leq \exp\left(\pi \frac{\|\mathbf{R}\mathbf{z}_H\|_2^2}{s^2} + 2\frac{\|\mathbf{R}\mathbf{z}_H\|_2}{s}\sqrt{\pi \ln \varepsilon^{-1}}\right) \\ &\leq \exp\left(\pi \frac{T^2}{s^2} + 2\frac{T}{s}\sqrt{\pi \ln \varepsilon^{-1}}\right) \\ &\leq M, \end{aligned}$$

where the last inequality follows from the fact that $s = \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}}) \cdot T$. Thence, $\max_{\mathbf{R}\mathbf{z}_H \in Y} RD_\infty^\varepsilon(\mathcal{D}_{R^{2d},s} \| \mathcal{D}_{R^{2d},s}^{+\mathbf{R}\mathbf{z}_H}) \leq M$. Theorem 5.1 then yields the result. □

We can now formally state the strong EUF-CMA security of Phoenix for uncompressed public key and then discuss the slight differences stemming from key compression. Compared to the original GPV security result [GPV08], we note that we rely on a version of M-SIS which adds a norm check in the infinity norm of the candidate solutions.

Theorem 6.1 ([GPV08] adapted). *Phoenix is strongly EUF-CMA-secure in the random oracle model under* $\text{M-LWE}_{n,d,d,q,U(S_1)}$ *and* $\text{M-SIS}_{n,d,d(2+k-\ell),q,\beta,\beta_\infty}$, *where* $\beta = 2\sqrt{B_{1,1}^2 + B_{1,2}^2 + nd(k-\ell)}$ *and* $\beta_\infty = 2\max(B_{1,1}^\infty, B_{1,2}^\infty, 1) = 2B_{1,1}^\infty$. *More precisely, the advantage of $\mathcal{A}$ attacking the unforgeability of Phoenix is bounded by*

$$Adv[\mathcal{A}] \lesssim \left(\frac{1}{1-\varepsilon}\right)^Q (\varepsilon_{\text{M-SIS}} + d(k-\ell)\varepsilon_{\text{M-LWE}}).$$

As our scheme features key compression, we can use the M-LWE assumption in the security reduction but the public key will not be uniform over R_q but only over the high-order bits. This would give a skewed M-SIS assumption over the instance $[\mathbf{I}_d|\mathbf{A}'|\mathbf{G}_H - \mathbf{B}_H]$ where the third block only has high-order bits. Since solving M-SIS involves discarding columns as described in Appendix A to find an optimal subdimension between nd and $2nd$, this skewed assumption could be estimated by $\text{M-SIS}_{n,d,2d,q,\beta',\beta'_\infty}$ where the bounds are set by taking $\mathbf{v}_2 = \mathbf{0}$. In all cases, this will not affect our concrete parameters that we directly derive from the M-ISIS instance corresponding to our signature, as explained below.

6.4 Concrete Parameters

We now suggest parameter sets to instantiate Phoenix in Table 6.1. Although our scheme is presented over modules of rank d, working over rings offers better key compression. We thus give parameters in the ring setting. As all our tools hold for general number fields, we can use cyclotomic fields of composite conductors. This has been done in Mitaka [EFG+22] to achieve fine-grained security levels where they consider 3-smooth conductors. In this case, it incurs a loss of $\sqrt{2}$ in the quality of our sampler similarly to [EFG+22] due to the spectral bound on $\mathbf{R}$, which we take into account in our parameter selection. An alternative would be to choose a power-of-two cyclotomic ring of smaller degree and a larger rank d so that nd matches the dimension we suggest, the parameters scaling with nd. Although it would deteriorate the key sizes, it can be acceptable in applications where the public key is not sent often.

The concrete security is assessed as described in Appendix A. At a high-level, the key recovery is evaluated via the M-LWE assumption. The complexity of the forgery is lower-bounded by Theorem 6.1. However, it is best approximated via the inhomogeneous variant M-ISIS as is done in most hash-and-sign schemes [PFH+20,EFG+22,YJW23]. We follow the same approach and estimate it using the Approx-CVP attack carried by the nearest-colattice algorithm [EK20]. We see that the forgery security for Phoenix-III, estimated through M-ISIS in Euclidean norm, falls a few bits short of NIST-III level. Our estimate is however rather pessimistic because we discard the infinity norm bound and the asymmetry between $\mathbf{v}_{1,1}$ and $\mathbf{v}_{1,2}$. Our cryptanalysis thus underestimates the actual complexity of the forgery. As pointed out in Appendix A, we believe that a thorough cryptanalysis would place the cost of the forgery above the NIST-III requirement, and also yield a better security for Phoenix-II and Phoenix-V. We however leave this cryptanalysis for future work.

We now compare in Table 6.2 the performance and security of Phoenix with the other M-LWE-based signatures Dilithium [DKL+18], Haetae [CCD+23], Raccoon [dPEK+], and Eagle [YJW23], although only Eagle is a hash-and-sign scheme. We note for completeness that other hash-and-sign schemes based on NTRU such as Falcon [PFH+20], Mitaka [EFG+22], or Robin [YJW23] usually achieve a smaller bandwidth (signature + public key) than the schemes from Table 6.2, but at the expense of an extra assumption. Designs based on M-LWE may be preferred to those based on NTRU in specific use cases, e.g., with strong

	Phoenix-II	Phoenix-III	Phoenix-V
Security	NIST-II	NIST-III	NIST-V
Conductor	2^{11}	$2^4 3^5$	$2^3 3^6$
n	1024	1296	1944
d	1	1	1
(k, ℓ, ℓ')	(16,15,8)	(17,16,9)	(18,17,10)
q	$2^{17} - 1$	$2^{18} - 1$	$2^{19} - 1$
(M, ε, α)	$(20, 2^{-66}, 8.13)$	$(20, 2^{-66}, 8.13)$	$(20, 2^{-66}, 8.13)$
s	20105	35986	53978
$B_{1,1}$	688341.2	1541069.0	3705333.9
$B_{1,2}$	268983.0	541623.4	995025.8
$B^{\infty}_{1,1}$	64537	127114	238760
$B^{\infty}_{1,2}$	36895	66037	99056
|sk| (B)	512	648	972
|pk| (B)	1184	1490	2219
|sig| (B)	2190	2897	4468
Key Recovery (C/Q)	162/147	203/184	312/283
Forgery (C/Q)	125/113	161/146	257/233

Table 6.1. Suggested parameter sets for Phoenix. Sizes are in bytes. The public key size includes 32 bytes for the seed that expands to $\mathbf{A}'$. The size of Gaussian vectors is estimated by the entropy bound which can be achieved via the rANS encoding (see [ETWY22]). The bit security is the estimated core-SVP hardness (classical C, quantum Q).

and lasting privacy features, or with stretched parameters. Such schemes also carry a certain complexity of implementation due to complex Gaussian samplers (FFO sampler for [PFH+20], hybrid sampler for [EFG+22], perturbation samplers for [YJW23]).

We can see that, at the exception of Haetae, we achieve more compact keys and signatures than these schemes. Our scheme also benefits from interesting features due to the nature of the LW sampler. More precisely, Phoenix inherits the implementation-friendly nature of Fiat-Shamir designs as it only involves spherical Gaussians. In particular, it does not need complex Gaussian samplers as in previous hash-and-sign schemes which is desirable for side-channel protection. Phoenix can also be adapted to use other distributions like uniform bounded to avoid floating points altogether, as described in Appendix B. Despite our signature overhead compared to Haetae, we achieve similar public key sizes, and our secret keys are about 65% smaller. Also, although it reduces to M-SIS, the forgery of Fiat-Shamir constructions [DKL+18,CCD+23,dPEK+] relies on (variants of) the self-target-M-SIS assumption which is less standard than M-ISIS. Certain use cases may thus benefit from the more conservative assumption of Phoenix.

Overall, our signature combines the benefits of Fiat-Shamir with Aborts schemes and of hash-and-sign schemes, as was originally expected from the LW sampler, and can therefore be seen as an addition to the toolbox of signature

	$\|\mathsf{sk}\|$ (B)	$\|\mathsf{pk}\|$ (B)	$\|\mathsf{sig}\|$ (B)	λ (C/Q)
Dilithium-2	2544	1312	2420	121/110
Haetae-120	1376	992	1463	97/85
Raccoon-128	14800	2256	11524	133/114
Phoenix-II	512	1184	2190	125/113
Dilithium-3	4016	1952	3293	176/159
Haetae-180	2080	1472	2337	149/131
Raccoon-192	18840	3160	14554	193/166
Eagle-1024	512	1952	3052	176/160
Phoenix-III	648	1490	2897	161/146
Dilithium-5	4880	2592	4595	252/229
Haetae-260	2720	2080	2908	214/188
Raccoon-256	26016	4064	20330	284/243
Phoenix-V	972	2219	4468	257/233

Table 6.2. Security (strong EUF-CMA versions) and performance comparisons between Dilithium [DKL+18], Haetae [CCD+23], Raccoon [dPEK+], Eagle [YJW23], and Phoenix.

designs. It also shows that said sampler is not only of theoretical interest but can be used to design schemes with desirable features. More generally, we believe that there may be other applications that could benefit from the unique features of the LW sampler and thus hope that our work will incite to investigate them.

Acknowledgments. This work has received a French government support managed by the National Research Agency in the ASTRID program, under the national project AMIRAL with reference ANR-21-ASTR-0016, and in the MobiS5 project with reference ANR-18-CE-39-0019-02 MobiS5. We warmly thank Vadim Lyubashevsky for helpful discussions on the Lyubashevsky-Wichs sampler, as well Nicholas Genise for interesting discussions on approximate trapdoors. We also thank David Pointcheval for his insight on the use of the forking lemma, and Katharina Boudgoust for her constructive feedback on the use of Gaussians in aggregate signatures.

References

ABB+20. E. Alkim, P. S. L. M. Barreto, N. Bindel, J. Krämer, P. Longa, and J. E. Ricardini. The lattice-based digital signature scheme qtesla. In *ACNS*, 2020.

AKSY22. S. Agrawal, E. Kirshanova, D. Stehlé, and A. Yadav. Practical, round-optimal lattice-based blind signatures. In *CCS*, 2022.

APS15. M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Math. Cryptol.*, 2015.

Ban93. W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Math. Ann.*, 1993.

BDK+18. J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *EuroS&P*, 2018.

BEP+21. P. Bert, G. Eberhart, L. Prabel, A. Roux-Langlois, and M. Sabt. Implementation of lattice trapdoors on modules and applications. In *PQCrypto*, 2021.

BFRS18. P. Bert, P.-A. Fouque, A. Roux-Langlois, and M. Sabt. Practical implementation of ring-sis/lwe based signature and IBE. In *PQCrypto*, 2018.

BGLS03. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT*, 2003.

BGP22. K. Boudgoust, E. Gachon, and A. Pellet-Mary. Some easy instances of ideal-svp and implications on the partial vandermonde knapsack problem. In *CRYPTO*, 2022.

BJRW23. K. Boudgoust, C. Jeudy, A. Roux-Langlois, and W. Wen. On the hardness of module learning with errors with short distributions. *J. Cryptol.*, 2023.

BLNS23. W. Beullens, V. Lyubashevsky, N. K. Nguyen, and G. Seiler. Lattice-based blind signatures: Short, efficient, and round-optimal. *IACR Cryptol. ePrint Arch.*, page 77, 2023.

BN06. M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *CCS*, 2006.

BNN07. M. Bellare, C. Namprempre, and G. Neven. Unrestricted aggregate signatures. In *ICALP*, 2007.

BR21. K. Boudgoust and A. Roux-Langlois. Non-interactive half aggregate signatures based on module lattices - a first attempt. *IACR Cryptol. ePrint Arch.*, page 263, 2021.

BTT22. C. Boschini, A. Takahashi, and M. Tibouchi. Musig-l: Lattice-based multi-signature with single-round online phase. In *CRYPTO*, 2022.

CCD+23. J. H. Cheon, H. Choe, J. Devevey, T. Güneysu, D. Hong, M. Krausz, G. Land, M. Möller, D. Stehlé, and M. Yi. HAETAE: shorter lattice-based fiat-shamir signatures. *IACR Cryptol. ePrint Arch.*, page 624, 2023.

CGM19. Y. Chen, N. Genise, and P. Mukherjee. Approximate trapdoors for lattices and smaller hash-and-sign signatures. In *ASIACRYPT*, 2019.

Che13. Y. Chen. *Réduction de Réseau et Sécurité Concrète du Chiffrement Complètement Homomorphe*. PhD thesis, Paris 7, 2013.

DEP23. L. Ducas, T. Espitau, and E. W. Postlethwaite. Finding short integer solutions when the modulus is small. In *CRYPTO*, 2023.

DFPS22. J. Devevey, O. Fawzi, A. Passelègue, and D. Stehlé. On rejection sampling in lyubashevsky's signature scheme. In *ASIACRYPT*, 2022.

DHSS20. Y. Doröz, J. Hoffstein, J. H. Silverman, and B. Sunar. MMSAT: A scheme for multimessage multiuser signature aggregation. *IACR Cryptol. ePrint Arch.*, page 520, 2020.

DKL+18. L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *TCHES*, 2018.

DLP14. L. Ducas, V. Lyubashevsky, and T. Prest. Efficient identity-based encryption over NTRU lattices. In *ASIACRYPT*, 2014.

DM14. L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO*, 2014.

DORS08. Y. Dodis, R. Ostrovsky, L. Reyzin, and A. D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 2008.

dPEK+. R. del Pino, T. Espitau, S. Katsumata, M. Maller, F. Mouhartem, T. Prest, M. Rossi, and M.-J. Saarinen. *Raccoon: A Side-Channel Secure Signature Scheme.*

dPK22. R. del Pino and S. Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In *CRYPTO*, 2022.

dPLS18. R. del Pino, V. Lyubashevsky, and G. Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. In *CCS*, 2018.

DSH21. A. Le Dévéhat, H. Shizuya, and S. Hasegawa. On the higher-bit version of approximate inhomogeneous short integer solution problem. In *CANS*, 2021.

EFG+22. T. Espitau, P.-A. Fouque, F. Gérard, M. Rossi, A. Takahashi, M. Tibouchi, A. Wallet, and Y. Yu. Mitaka: A simpler, parallelizable, maskable variant of falcon. In *EUROCRYPT*, 2022.

EK20. T. Espitau and P. Kirchner. The nearest-colattice algorithm: Time-approxmation tradeoff for approx-cvp. In *ANTS XIV*, 2020.

ETWY22. T. Espitau, M. Tibouchi, A. Wallet, and Y. Yu. Shorter hash-and-sign lattice-based signatures. In *CRYPTO*, 2022.

GM18. Nicholas Genise and Daniele Micciancio. Faster gaussian sampling for trapdoor lattices with arbitrary modulus. In *EUROCRYPT*, 2018.

GMPW20. N. Genise, D. Micciancio, C. Peikert, and M. Walter. Improved discrete gaussian and subgaussian analysis for lattice cryptography. In *PKC*, 2020.

GPV08. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.

HKW15. S. Hohenberger, V. Koppula, and B. Waters. Universal signature aggregators. In *EUROCRYPT*, 2015.

HW18. S. Hohenberger and B. Waters. Synchronized aggregate signatures from the RSA assumption. In *EUROCRYPT*, 2018.

JRS23. C. Jeudy, A. Roux-Langlois, and O. Sanders. Lattice signature with efficient protocols, application to anonymous credentials. In *CRYPTO*, 2023.

LLM+16. B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *ASIACRYPT*, 2016.

LNP22a. V. Lyubashevsky, N. K. Nguyen, and M. Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. *CRYPTO*, 2022.

LNP22b. V. Lyubashevsky, N. K. Nguyen, and M. Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. *IACR Cryptol. ePrint Arch.*, page 284, 2022.

LNPS21. V. Lyubashevsky, N. K. Nguyen, M. Plançon, and G. Seiler. Shorter lattice-based group signatures via "almost free" encryption and other optimizations. In *ASIACRYPT*, 2021.

LS15. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *DCC*, 2015.

LW15. V. Lyubashevsky and D. Wichs. Simple lattice trapdoor sampling from a broad class of distributions. In *PKC*, 2015.

Lyu12. V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, 2012.

MP12. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, 2012.

MP13. D. Micciancio and C. Peikert. Hardness of SIS and LWE with small parameters. In *CRYPTO*, 2013.

MR07. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 2007.

MW16. D. Micciancio and M. Walter. Practical, predictable lattice basis reduction. In *EUROCRYPT*, 2016.

NISa. NIST. Post-quantum cryptography standardization. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization.

NISb. NIST. Post-quantum cryptography: Standardization of additional digital signature schemes. https://csrc.nist.gov/Projects/pqc-dig-sig/standardization.

Pei08. C. Peikert. Limits on the hardness of lattice problems in l_p norms. *Comput. Complex.*, 2008.

Pei10. C. Peikert. An efficient and parallel gaussian sampler for lattices. In *CRYPTO*, 2010.

PFH+20. T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. *FALCON. Tech. rep.*, 2020. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

PR06. C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, 2006.

Pre17. T. Prest. Sharper bounds in lattice-based cryptography using the rényi divergence. In *ASIACRYPT*, 2017.

RS13. M. Rückert and D. Schröder. Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. *IACR Cryptol. ePrint Arch.*, page 20, 2013.

TS23. T. Tomita and J. Shikata. Compact signature aggregation from module-lattices. *IACR Cryptol. ePrint Arch.*, page 471, 2023.

Ver12. R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing*. 2012.

YJW23. Y. Yu, H. Jia, and X. Wang. Compact lattice gadget and its applications to hash-and-sign signatures. In *CRYPTO*, 2023.

ZY22. S. Zhang and Y. Yu. Towards a simpler lattice gadget toolkit. In *PKC*, 2022.

A Concrete Security Analysis

In this section we recall the methodology we use to estimate the bit security of the forgery and key recovery attacks in Section 4 and for our Phoenix signature scheme in Section 6.4. All our estimates use the Core-SVP model where the cost of the attack is given by the cost of running once the self-dual BKZ lattice reduction [MW16] with block size B. The cost is then modeled by the best known cost for lattice sieving, i.e., $2^{0.292B}$ for the classical security and $2^{0.265B}$ for the quantum security.

Under the Gaussian Heuristic and the Geometric Series Assumption, the BKZ algorithm with blocksize B would find a vector $\mathbf{v}$ in a N-dimensional lattice $\mathcal{L}$ with $\|\mathbf{v}\|_2 \leq \delta_B^N \mathrm{Vol}(\mathcal{L})^{1/N}$, where

$$\delta_B \approx \left(\frac{(\pi B)^{\frac{1}{B}} B}{2\pi e} \right)^{\frac{1}{2(B-1)}}, \tag{4}$$

by [Che13].

A.1 Key Recovery: M-LWE

In all the schemes derived from the samplers in Section 4, the public key is given by $\mathbf{A}' \in R_q^{d \times m_1 - d}$ and $\mathbf{B} = [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR$ and the secret key by $\mathbf{R} \sim U(S_1^{m_1 \times m_2})$. Except for the LW signature, all our schemes use $m_1 = 2d$. Key recovery thus corresponds to an instance of search M-LWE$_{n,d,m_1-d,q,U(S_1)}$ with m_2 uniform ternary secrets. We use the lattice estimator [APS15] on the instance LWE$_{nd,n(m_1-d),q,U(\{-1,0,1\})}$ to determine the minimal BKZ block size B among all the evaluated attacks. We discard the structure of the underlying ring and simply extend the dimensions by the ring degree n by considering the matrix $M_\tau(\mathbf{A}')$. To account for the m_2 secrets, we consider the final cost to be that of running m_2 times BKZ which gives a cost of $m_2 2^{\nu B}$ for $\nu \in \{0.292, 0.265\}$.

In the case of Phoenix, we apply public key compression which means that the adversary only has access to the high-order bits of $\mathbf{B}$. At a high-level, the key recovery consists in solving $d(k-\ell)$ instances of M-LWE to recover $\mathbf{R}$ from $\mathbf{B}_H \bmod q$. Since $\mathbf{B}_H$ contains less information on $\mathbf{R}$ than the full matrix $\mathbf{B} = [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR$, we lower bound the complexity of key recovery by assessing the cost of recovering $\mathbf{R}$ given $\mathbf{B}$ as described above.

A.2 Forgery: M-SIS or M-ISIS

The complexity of the forgery can be estimated either by the security proof which relies on the M-SIS assumption, or by the M-ISIS assumption. In Section 4, we use the former approach on the M-SIS$_{n,d,m,q,\beta,\beta_\infty}$ assumption where the infinity norm $\beta_\infty < q$ is discarded except for ensuring that q-vectors are not solutions. For Phoenix, we aim for a tighter security assessment using the M-ISIS$_{n,d,m,q,\beta',\beta'_\infty}$ assumption. Both approaches are detailed below.

A.2.1 Solving M-SIS. To estimate the security of M-SIS$_{n,d,m,q,\beta,\beta_\infty}$, we find the cost of finding $\mathbf{v} \in \mathcal{L}_q^\perp([\mathbf{I}_d|\mathbf{A}'|\mathbf{B}'])$ such that $\|\mathbf{v}\|_2 \leq \beta$ and $\|\mathbf{v}\|_\infty \leq \beta_\infty$, given $\mathbf{A}' \sim U(R_q^{d\times m_1-d})$ and $\mathbf{B}' \sim U(R_q^{d\times m_2})$ (with $m = m_1 + m_2$). We again look at the unstructured problem SIS$_{nd,nm,q,\beta,\beta_\infty}$. For that, we first check that $\min(\beta, \beta_\infty) < q$ to avoid trivial solutions. Then, a standard optimization consists in finding a solution in a lattice of smaller dimension $nd \leq m^* \leq nm$ and completing the solution with zeros. We then use BKZ in block size B such that

$$\beta \geq \min_{nd\leq m^*\leq nm} \delta_B^{m^*} q^{nd/m^*}.$$

More precisely, for a fixed β, we find m^* that maximizes $\delta_B = \beta^{1/m^*} q^{-nd/m^{*2}}$ and then use Equation (4) to determine the corresponding block size B.

A.2.2 Direct Forgery: M-ISIS. In Phoenix, we estimate the forgery security via the M-ISIS assumption. A forgery consists of a vector $\mathbf{v} = [\mathbf{v}_{1,1}^T|\mathbf{v}_{1,2}^T|\mathbf{v}_2^T]^T$ such that $[\mathbf{I}_d|\mathbf{A}'|\mathbf{G}_H - \mathbf{B}_H]\mathbf{v} = \mathbf{u} \bmod qR$ for a seemingly random and non-adversarial syndrome $\mathbf{u} = \mathcal{H}(\mathsf{salt}, \mathbf{m})$. Since the adversary must provide the salt as part of the signature, the best strategy is to select an arbitrary message and salt, compute $\mathbf{u} = \mathcal{H}(\mathsf{salt}, \mathbf{m})$ and find $\mathbf{v}$. Additionally, as $\mathbf{v}_2$ has very strict bounds (ternary), it is unlikely to have such small coefficients for $\mathbf{v}_2$ by solving M-ISIS on $([\mathbf{I}_d|\mathbf{A}'|\mathbf{G}_H - \mathbf{B}_H], \mathbf{u})$, unless they are set to zero. To hope for a valid forgery, one would thus fix a value for $\mathbf{v}_2 \in S_1^{d(k-\ell)}$ and solve the M-ISIS instance $([\mathbf{I}_d|\mathbf{A}'], \mathbf{u}' = \mathbf{u} - (\mathbf{G}_H - \mathbf{B}_H)\mathbf{v}_2)$ with norm bounds set from the signature verification from Algorithm 6.4. Setting $\mathbf{v}_2 = \mathbf{0}$ would discard these columns which is done in the concrete attack below anyway. Due to the asymmetry of our preimages, the solution returned by the adversary should also have a specific form. In particular $\mathbf{v}_{1,1}, \mathbf{v}_{1,2}$ are bounded both in Euclidean and infinity norms. This makes the fine-grained cryptanalysis difficult as current lattice reduction algorithms focus mostly on the Euclidean norm. Our approach is therefore once again to underestimate the actual cost of the attack by discarding the infinity norm and also the asymmetry of the solution. We believe that a thorough cryptanalysis would show that the forgery is more complex than the approach we describe here. More precisely, we simply evaluate the complexity of finding $\mathbf{v}_1$ such that $[\mathbf{I}_d|\mathbf{A}']\mathbf{v}_1 = \mathbf{u}' \bmod q$ and $\|\mathbf{v}_1\|_2 \leq \beta = \sqrt{B_{1,1}^2 + B_{1,2}^2}$. We note that if β is close to or larger than $q\sqrt{nd/12}$, this M-ISIS instance becomes trivial but not the forgery because of our infinity norm checks.

If $\beta < q\sqrt{nd/12}$, a solution can be found using the Approximate CVP attack using the nearest-colattice algorithm of Espitau and Kirchner [EK20]. Given $(M_\tau([\mathbf{I}_d|\mathbf{A}']), \tau(\mathbf{u}')) \in \mathbb{Z}_q^{N\times D} \times \mathbb{Z}_q^N$, where $N = nd$ and $D = 2nd$, the algorithm can compute a solution within Euclidean norm β with BKZ of block size B such that

$$\beta \geq \min_{k^*\leq D-N} \delta_B^{D-k^*} q^{N/(D-k^*)}.$$

Again, for a fixed β, we find k^* which maximizes $\delta_B = \beta^{1/(D-k^*)} q^{-N/(D-k^*)^2}$ and use Equation (4) to determine the block size B.

Although our modulus is not particularly small with respect to the dimension and the M-ISIS bound, we also ran the estimator recently proposed by Ducas, Espitau and Postlethwaite [DEP23] as a sanity check to make sure it does not lead to a more efficient attack than the previously described approach. Their tool unfortunately suffers from large memory requirements when computing the intersection of the hypercube and ball if the parameters are too large. We also leave this cryptanalysis to future work. The preimage and key compression can easily be reduced, and as a result the M-ISIS bound, to avoid these vulnerable parameter regimes at the expense of slightly larger signatures and/or keys. Nevertheless, we again insist on the fact that our scheme also places infinity norm bounds which may invalidate the attack or make it much more complex.

B Uniform Version of Phoenix

We describe a version of the Phoenix signature scheme where we instantiate the distribution of signatures with uniform distributions instead of Gaussians. Although it suffers from larger signature sizes, it has the advantage of requiring no floating point arithmetic whatsoever. Additionally, the rejection step is deterministic which makes the scheme even easier to implement. Although it follows the hash-and-sign paradigm in the GPV framework, the resulting scheme has many similarities with the Dilithium signature scheme [DKL+18]. As such, further optimizations to Dilithium could also be applied to our scheme to heighten its efficiency.

B.1 Bounds and Uniform Approximate Rejection Sampler

For completeness, we give the modified sampler tailored for uniform distributions. As for Phoenix, we choose $q = 2^{k+1} - 1$, and the gadget decomposition is centered. The error distribution coming from dropping low-order bits is exactly the same as that of Phoenix, and so is the key compression error

Algorithm B.1: Approx.SamplePreRej($\mathbf{R}; \mathbf{A}', \mathbf{u}, \gamma, B$)

Input: Trapdoor $\mathbf{R} \in R^{2d \times d(k-\ell)}$, Matrix $\mathbf{A}' \in R_q^{d \times d}$, Syndrome $\mathbf{u} \in R_q^d$, Mask bound $\gamma > 0$, Secret bound $B > 0$.

1. $\mathbf{p}_1 \hookleftarrow U(S_\gamma^{2d})$.
2. $\mathbf{w} \leftarrow \mathbf{u} - [\mathbf{I}_d | \mathbf{A}']\mathbf{p}_1 \bmod qR$. ▷ Syndrome correction
3. $\mathbf{z} \leftarrow \mathbf{G}^{-1}(\mathbf{w}) \in S_1^{dk}$.
4. Parse $\mathbf{z}$ into $\mathbf{z}_L \in S_1^{d\ell}$ and $\mathbf{z}_H \in S_1^{d(k-\ell)}$ so that $\mathbf{Gz} = \mathbf{G}_L\mathbf{z}_L + \mathbf{G}_H\mathbf{z}_H$.
5. $\mathbf{v}_1' \leftarrow \mathbf{p}_1 + \mathbf{R}\mathbf{z}_H$.
6. **if** $\|\mathbf{v}_1'\|_\infty > \gamma - B$, go back to 1.
7. **else** $\mathbf{v}_1 \leftarrow \mathbf{v}_1' + \begin{bmatrix} \mathbf{G}_L\mathbf{z}_L \\ \mathbf{0} \end{bmatrix}$
8. $\mathbf{v}_2 \leftarrow \mathbf{z}_H$

Output: $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$.

We can once again adapt Theorem 5.1 to ensure the simulatability of preimages. We state it in the following corollary for completeness.

Corollary B.1. *Let d, k, ℓ, q be positive integers such that $q = 2^k - 1$ and $\ell \in [0, k-1]$. Let $\mathbf{A}' \sim U(R_q^{d\times d})$ and $\mathbf{R} \sim U(S_1^{2d\times d(k-\ell)})$. Then, we let B be a bound on $\|\mathbf{R}\mathbf{z}_H\|_\infty$ and $M > 1$ be the average repetition rate. We define $\gamma = \lceil B \cdot M^{1/2nd}/(M^{1/2nd} - 1) - 1/2 \rceil$. We define $\mathcal{P}_1$ and $\mathcal{P}_2$ the same way as in Theorem 5.1 but where $\mathscr{D}_s = U(S_\gamma^{2d})$ and $\mathscr{D}_t = U(S_{\gamma-B}^{2d})$. Then, it holds that $\mathcal{P}_1$ and $\mathcal{P}_2$ are identical.*

B.2 The Signature Scheme

Algorithm B.2: Setup

Input: Security parameter λ.

1. Choose positive integers d, k.
2. $q \leftarrow 2^{k+1} - 1$.
3. Choose $\ell, \ell' \in [0, k-1]$.
4. $\mathbf{G} = \mathbf{I}_d \otimes [1 \cdots 2^{k-1}] \in R_q^{d\times dk}$.
5. $\mathbf{G}_H = \mathbf{I}_d \otimes [2^\ell \cdots 2^{k-1}] \in R_q^{d\times d(k-\ell)}$.
6. $\mathbf{G}_L = \mathbf{I}_d \otimes [1 \cdots 2^{\ell-1}] \in R_q^{d\times d\ell}$.
7. Fix B a bound on $\|\mathbf{R}\mathbf{z}_H\|_\infty$.
8. Choose $M > 1$. ▷ Repetition rate
9. $\gamma \leftarrow \left\lceil B \cdot M^{1/2nd}/(M^{1/2nd} - 1) - 1/2 \right\rceil$
10. $\mathbf{A}' \hookleftarrow U(R_q^{d\times d})$.

Output: $\mathsf{pp} = (\mathbf{A}'; \mathbf{G}, \mathbf{G}_L, \mathbf{G}_H; \lambda, n, q, d, k, \ell, B, \gamma, M)$.

Algorithm B.3: KeyGen

Input: Public parameters pp as in Algorithm B.2.

1. $\mathbf{R} \hookleftarrow U(S_1^{2d\times d(k-\ell)})$.
2. $\mathbf{B} \leftarrow [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR \in R_q^{d\times d(k-\ell)}$
3. Parse $\mathbf{B}$ as $\mathbf{B}_L + \mathbf{B}_H$ with $\mathbf{B}_L \in S_{2^{\ell'}-1}^{d\times d(k-\ell)}$ and $\mathbf{B}_H \in 2^{\ell'} S_{2^{k-\ell'}-1}^{d\times d(d-\ell)}$.

Output: $\mathsf{pk} = \mathbf{B}_H$, and $\mathsf{sk} = \mathbf{R}$. ▷ pp stored with pk for simplicity

Algorithm B.4: Sign

Input: Secret key sk, Message $\mathbf{m} \in \{0,1\}^*$, Public key pk.

1. $\mathsf{salt} \hookleftarrow U(\{0,1\}^{320})$.
2. $(\mathbf{v}_1, \mathbf{v}_2) \leftarrow \mathsf{Approx.SamplePreRej}(\mathbf{R}; \mathbf{A}', \mathcal{H}(\mathbf{m}, \mathsf{salt}), s)$. ▷ Algorithm B.1
3. $\mathbf{e}_{\mathsf{pk}} \leftarrow ([\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR) - \mathbf{B}_H)\mathbf{v}_2$.
4. Parse $\widetilde{\mathbf{v}_1}$ as $[\widetilde{\mathbf{v}_{1,1}}^T | \mathbf{v}_{1,2}^T]^T$ with $\widetilde{\mathbf{v}_{1,1}}, \mathbf{v}_{1,2} \in R^d$.
5. $\mathbf{v}_{1,1} \leftarrow \widetilde{\mathbf{v}_{1,1}} - \mathbf{e}_{\mathsf{pk}}$.
6. $\gamma_1 \leftarrow (\|\mathbf{v}_{1,1}\|_\infty \le B_{1,1}^\infty) \wedge (\|\mathbf{v}_{1,2}\|_\infty \le B_{1,2}^\infty) \wedge (\|\mathbf{v}_2\|_\infty \le 1)$.
7. **if** $\gamma_1 = 0$, restart.

Output: $\mathsf{sig} = (\mathsf{salt}, \mathbf{v}_{1,2}, \mathbf{v}_2)$.

Algorithm B.5: Verify

Input: Public key pk, Message $\mathbf{m} \in \{0,1\}^*$, Signature sig.

1. $\mathbf{v}_{1,1} \leftarrow \mathcal{H}(\mathbf{m}, \mathsf{salt}) - \mathbf{A}'\mathbf{v}_{1,2} - (\mathbf{G}_H - \mathbf{B}_H)\mathbf{v}_2 \bmod qR \in R^d$.
2. $\gamma_1 \leftarrow (\|\mathbf{v}_{1,1}\|_\infty \le B_{1,1}^\infty) \wedge (\|\mathbf{v}_{1,2}\|_\infty \le B_{1,2}^\infty) \wedge (\|\mathbf{v}_2\|_\infty \le 1)$.

Output: γ_1. ▷ 1 if valid, 0 otherwise

The verification bounds are simply set as

$$B_{1,1}^{\infty} = \gamma - B + 2^{\ell} - 1 + \left\lfloor 3.8\mu\sqrt{\frac{2^{\ell'}(2^{\ell'}-1)}{6}\frac{nd(k-\ell)}{2}} \right\rfloor$$
$$B_{1,2}^{\infty} = \gamma - B.$$

The bound B can be derived by studying the distribution of $\mathbf{Rz}_H$. In particular, one can obtain a much tighter bound than the trivial $nd(k-\ell)$ that is still verified with overwhelming probability. Consider the power-of-two cyclotomics case and that $d = k - \ell = 1$. Then, $\mathbf{Rz}_H = [r_1 z | r_2 z]^T$ is a vector of R^2, where $r_i \sim U(S_1)$ and z almost follows the centered binomial distribution. This is because each coefficient of z corresponds to the high order bits of some $|u|$ for u uniform in $[-(q-1)/2, (q-1)/2]$, mutiplied by the sign of u. As we have $\tau(r_i z) = M_\tau(r_i)\tau(z)$ the i-th coefficient is given by $\sum_j \pm r_{i,j_i} z_j$. Because $U([-1,1])$ is centered, $\pm r_{i,j_i}$ follows the same distribution, and thus $\pm r_{i,j_i} z_j$ follows a centered binomial distribution of parameter $2/3$ which we call $\mathcal{B}_{1,2/3}$. That is 0 with probability $2/3$ and ± 1 each with probability $1/6$. We can then use Chernoff bound using the cumulant generating function $K(\cdot)$ of $\mathcal{B}_{1,2/3}$ defined by $K(t) = \ln(\mathbb{E}[\exp(t\mathcal{B}_{1,2/3})]) = \ln(2/3 + 1/3\cosh(t))$ for $t \in \mathbb{R}$. The Chernoff bound gives $\mathbb{P}[|\sum_j \pm r_{i,j_i} z_j| \geq B] \leq 2^{-\lambda\alpha(B,n,\lambda)}$ where $\alpha(B,n,\lambda) = \frac{\log_2 e}{\lambda+1}\sup_{t\geq 0}(tB - nK(t))$. Then, the union bound gives $\mathbb{P}[\|\mathbf{Rz}_H\|_\infty \geq B] \leq 2n \cdot 2^{-\lambda\alpha(B,n,\lambda)}$. For a fixed λ, n, we can then solve for B so that the probability is at most $2^{-\lambda}$. For composite conductors, one also has to account for the slack μ. In practice we observe that B can even be slightly smaller than what the Chernoff bound gives. It could theoretically be enforced by rejecting the $\mathbf{v}_2$ (and thus the $\mathbf{p}_1$) that make $\mathbf{Rv}_2$ larger than B.

B.2.1 Security and Parameters. This scheme once again follows the GPV framework and thus inherit the same security analysis. As the simulation result of Corollary B.1 is a bit different and since the verification only involves the infinity norm, we give the security reduction result for completeness. We note that the version of M-SIS here only performs infinity norm checks. That is, we look for a non-zero vector $\mathbf{x}$ in $\mathcal{L}_q^{\perp}(\overline{\mathbf{A}})$ such that $\|\mathbf{x}\|_\infty \leq \beta_\infty$. When key compression is applied, the M-SIS assumption is also skewed as for Phoenix due to the block $\mathbf{G}_H - \mathbf{B}_H$

Theorem B.1 ([GPV08] adapted). *The signature scheme of Section B.2 is strongly EUF-CMA-secure in the random oracle model under* $\text{M-LWE}_{n,d,d,q,U(S_1)}$ *and* $\text{M-SIS}_{n,d,d(2+k-\ell),q,\beta_\infty}$*, where* $\beta_\infty = 2\max(B_{1,1}^{\infty}, B_{1,2}^{\infty}, 1) = 2B_{1,1}^{\infty}$*. More precisely, the advantage of* $\mathcal{A}$ *attacking the unforgeability is bounded by* $Adv[\mathcal{A}] \leq \varepsilon_{\text{M-SIS}} + d(k-\ell)\varepsilon_{\text{M-LWE}}$.

Just like Phoenix, we perform the forgery security assessment and parameter selection by looking at the M-ISIS instance that the scheme describes. We use the same methodology from Appendix A on the same M-ISIS instance but with a Euclidean bound specific to this scheme. As we deal with uniform elements,

we can evaluate the expected bounds and thus derive the M-ISIS norm bound from them. We essentially use the same observation based on the Gaussian approximation to derive the M-ISIS bound in Euclidean norm. In the case of $\mathbf{v}'_{1,1}$ (before adding the errors) and $\mathbf{v}_{1,2}$, they follow centered uniform distribution with bounds $\gamma - B$. As a result, they can be bounded with high probability by $\sqrt{(\gamma - B)(\gamma - B + 1)/3}\sqrt{nd} \approx (\gamma - B)\sqrt{nd/3}$. Then, just like in Phoenix, the sampling error $\mathbf{e} = \mathbf{G}_L \mathbf{z}_L$ can be bounded by $\sqrt{(2^\ell - 1))(2^{\ell+1} - 1)/6}\sqrt{nd} \approx 2^\ell \sqrt{nd/3}$. Finally, the key compression error $\mathbf{e}_{\mathsf{pk}}$ can be bounded in Euclidean norm by $\sqrt{2^{\ell'}(2^{\ell'} - 1)/3 \cdot nd(k - \ell)/2}\sqrt{nd}$. To be thorough so as to rely on this M-ISIS assumption, we would need to set Euclidean norm checks in the signing and verification process. Concretely, we would set

$$B_{1,1} = 1.04\sqrt{\frac{(\gamma - B)(\gamma - B + 1)}{3} + \frac{(2^\ell - 1)(2^{\ell+1} - 1)}{6} + \mu^2 \frac{2^{\ell'}(2^{\ell'} - 1)}{6} \frac{nd(k - \ell)}{2}}\sqrt{nd}$$

$$B_{1,2} = 1.04\sqrt{\frac{(\gamma - B)(\gamma - B + 1)}{3}}\sqrt{nd},$$

and then define the M-ISIS bound $\beta = \sqrt{B_{1,1}^2 + B_{1,2}^2}$. As for Phoenix, the bound provided by these formulas are verified empirically.

We now suggest parameter sets to instantiate this version in Table B.1. The public key is reasonable compared to prior constructions but it suffers from slightly larger signatures than Phoenix. This is to be balanced with the computational benefits of this variant. Additionally, we have not tried to optimize further this version as it is conceptually close to Dilithium [DKL+18]. Future optimizations could consist in re-using tricks from the latter and subsequent improvements to optimize this scheme. It may help further compress the signature size and public key or heighten security. Additionally, a thorough cryptanalysis is required to have a precise estimate of the security as our analysis does not consider the infinity norm at the moment.

Security	I$^-$	III
Conductor	2^{11}	$2^9 3^2$
n	1024	1536
d	1	1
(k, ℓ, ℓ')	(19,18,11)	(20,19,11)
q	$2^{20} - 1$	$2^{21} - 1$
(M, B)	$(20, 131)$	$(20, 186)$
γ	89622	190071
$B^{\infty}_{1,1}$	423507	838659
$B^{\infty}_{1,2}$	89491	189885
$B_{1,1}$	5359390.0	13189833.4
$B_{1,2}$	1719509.1	4468500.5
$\lvert\mathsf{sk}\rvert$ (B)	512	768
$\lvert\mathsf{pk}\rvert$ (B)	1184	1952
$\lvert\mathsf{sig}\rvert$ (B)	2600	4072
Key Recovery (C/Q)	134/122	211/191
Forgery (C/Q)	105/96	171/155

Table B.1. Suggested parameter sets for the scheme with uniform distribution. Sizes are in bytes. The bit security is the estimated core-SVP hardness (classical C, quantum Q).

C Application: Lattice-Based Aggregate Signature

As another application of how we can leverage the asymmetry of the preimage resulting from our new analysis, we construct the first lattice-based aggregate signature that supports public aggregation and that is more efficient than the naive concatenation of individual signatures. It in particular shows that the LW sampler from Algorithm 3.1 improved as described in Section 3.2 can lead to new signature designs. Note that in this section, we again focus on the *exact* version of the LW sampler and thus compare the aggregate signature to the results of Section 4. The aggregate signature scheme is directly based on the GPV framework, but with some necessary tweaks in order to be secure. It fully leverages the asymmetry between $\mathbf{v}_1$ and $\mathbf{v}_2$.

An aggregate signature is a regular signature scheme completed by a mechanism $\mathsf{AggSign}$ taking the public keys pk_i of N users as well as pairs of message-signature $(\mathbf{m}_i, \mathsf{sig}_i)$ from each user, and compresses all the sig_i into a single signature $\mathsf{sig}_{\mathsf{agg}}$. A second mechanism $\mathsf{AggVerify}$ is appended to verify that $\mathsf{sig}_{\mathsf{agg}}$ is a valid *aggregate* signature on the messages $\mathbf{m}_i$ under the keys pk_i, but without requiring the individual sig_i. One of the key features is that the aggregation is public and non-interactive, meaning it does not require the signers' secret keys nor does it need them to interact to produce $\mathsf{sig}_{\mathsf{agg}}$. A basic efficiency requirement is that the size of $\mathsf{sig}_{\mathsf{agg}}$ should be lower than the concatenation of the sig_i, the latter being the simplest form of aggregate signature.

Such primitives were first introduced by Boneh et al. [BGLS03], which has led to several efficient constructions on classical groups, such as for example

the works in [BGLS03,BNN07,RS13,HKW15,HW18]. Post-quantum constructions were however unknown until the first attempt of Döroz et al. [DHSS20]. This lattice-based proposal turned out to be either less efficient than the trivial concatenation of signatures, or prone to attacks due to their compression technique as pointed out by Boudgoust and Roux-Langlois [BR21]. Additionally, their construction was based on a non-standard assumption called the Partial Fourier Recovery problem for which the hardness confidence is limited due to recent results by Boudgoust, Gachon and Pellet-Mary [BGP22]. Boudgoust and Roux-Langlois also proposed in [BR21] an aggregate signature based on module lattices following the FSwA signature paradigm. Again, it turned out that the peculiarities of aggregate signature security led to $\mathsf{sig}_{\mathsf{agg}}$ being larger than the concatenation.

In this section, we construct the first lattice-based aggregate signature with public aggregation that achieves relevant compression compared to the concatenation of individual signatures. Our scheme stems from the GPV signature [GPV08] instantiated with MP trapdoors [MP12], and the LW sampler from [LW15] in our improved parameter setting as a key element. At a high level, each user has a key pair $(\mathsf{sk}_i, \mathsf{pk}_i) = (\mathbf{R}_i, \mathbf{B}_i = \mathbf{A}\mathbf{R}_i)$, where the matrix $\mathbf{A}$ is common to every signer. To sign a message $\mathbf{m}_i$, user i samples a short preimage $\mathbf{v}_i = [\mathbf{v}_{1,i}^T | \mathbf{v}_{2,i}^T]^T$ of $\mathcal{H}(\mathbf{m}_i)$ using our new method, where $\mathcal{H}$ is modeled as a random oracle. At this stage, it is tempting to simply add the first components $\mathbf{v}_{1,i}$ of each signature and concatenate the (very short) second ones $\mathbf{v}_{2,i}$. This would be correct, but the resulting scheme is completely insecure as we will explain. We then resort to a technique generally used to circumvent rogue-key attacks to ensure security, but with some necessary tweaks.

Concretely, to aggregate the $\mathbf{v}_i$, one first obtains small random weights e_i and computes $\mathsf{sig}_{\mathsf{agg}} = (\mathbf{v}_1 = \sum_i e_i \mathbf{v}_{1,i}, (\mathbf{v}_{2,i})_i)$. While this technique seems classical, we note that it is not as straightforward to generate suitable e_i as one might think at first glance. Indeed, generating e_i as the output of a single hash function does not seem sufficient to prove security, even in the random oracle model. This problem, which does not arise in classical cyclic groups, was already faced by the authors of [BR21] who circumvented it by weakening the security model. We show that we can avoid this by resorting to two random oracles $\mathcal{H}_f, \mathcal{H}_e$ to generate the weights e_i so as to deal with the peculiarities of the forking lemma. Concretely, we first compute $f = \mathcal{H}_f(\{\mathbf{B}_j, \mathbf{v}_{2,j}, \mathbf{m}_j\}_{1 \leq j \leq N})$, and then $e_i = \mathcal{H}_e(f, i) \in \mathcal{C}$ for all i, where $\mathcal{C}$ is the set of ternary polynomials with fixed Hamming weight. To verify, one can then recompute the weights e_i and check that $\mathbf{A}\mathbf{v}_1 + \sum_i e_i(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} = \sum_i e_i \mathcal{H}(\mathbf{m}_i)$. We thus manage to prove security according to the conventional model for aggregate signatures at the cost of only one additional hashing.

We only achieve partial aggregation because of the fact that $\mathbf{v}_{2,i}$ faces the matrix $\mathbf{B}_i$ which differs for every user. As a result, we need to transmit all the individual $\mathbf{v}_{2,i}$, thus yielding a size linear in N. However, because our new preimage sampling algorithm minimizes the size of the $\mathbf{v}_{2,i}$'s, it amortizes this linear dependency, enough to have relevant compression compared to the naive

concatenation. In particular, we obtain aggregate signatures that are up to 15% smaller than the concatenation for N ranging from 5 to 2000 which is a range coherent with real-life applications, such as certificate chains, blockchains or batch software updates for example.

We note that a work related to ours was very recently proposed online by Tomita and Shikata [TS23] which follows a different and generic approach based on batch arguments for NP relations and performing a proof of the individual signatures' verification. Additionally, no efficiency assessment is given beyond an asymptotic behavior, while our approach gives concrete estimates and does not rely on any proof system.

We start by giving the necessary preliminaries in Section C.1. Then, we recall the definition of aggregate signature schemes in Section C.2, before presenting our construction in Section C.3. Then, we prove the security of our scheme in the aggregate chosen-key model coined by Boneh et al. [BGLS03] in Section C.4. Finally, we dedicate Section C.5 to discussing the performance of our scheme.

C.1 Additional Preliminaries

The Hermitian of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^H$. We call $\mathsf{Vol}\ \mathcal{L}$ the *volume* of a lattice $\mathcal{L}$. For $\mathbf{x}$ a discrete random variable over a set S, we define its min-entropy as $H_\infty(\mathbf{x}) = -\log_2(\max_{\mathbf{x}' \in S} \mathbb{P}_{\mathbf{x}}[\mathbf{x} = \mathbf{x}'])$. We recall the following result stating that $\mathcal{D}_{\mathcal{L},s,\mathbf{c}}$ carries a good amount of entropy when s is sufficiently large. A similar result is given in [PR06, Lem. 2.10], but we give a tighter bound directly resulting from Poisson's summation formula. We give the proof for completeness.

Lemma C.1. *Let $\mathcal{L} \subset \mathbb{R}^d$ be a lattice of rank d. For any $\varepsilon > 0$, $s \geq \eta_\varepsilon(\mathcal{L})$, and $\mathbf{c} \in \mathbb{R}^d$, it holds that $H_\infty(\mathcal{D}_{\mathcal{L},s,\mathbf{c}}) \geq d \log_2 s - \log_2(\mathsf{Vol}\ \mathcal{L}) + \log_2(1-\varepsilon)$. In particular, when $\mathcal{L} = \mathbb{Z}^d$ and $\varepsilon \leq 1/2$, it yields $H_\infty(\mathcal{D}_{\mathbb{Z}^d,s}) \geq d \log_2 s - 1$.*

Proof. Let $\mathcal{L} \subset \mathbb{R}^d$ be a lattice of rank d, $\varepsilon > 0$, $s \geq \eta_\varepsilon(\mathcal{L})$ and $\mathbf{c} \in \mathbb{R}^d$. We look at $\rho_{s,\mathbf{c}}(\mathcal{L})$. By the Poisson summation formula, it holds that

$$\rho_{s,\mathbf{c}}(\mathcal{L}) = s^d (\mathsf{Vol}\ \mathcal{L})^{-1} \sum_{\mathbf{x} \in \mathcal{L}^*} e^{-i \cdot 2\pi \mathbf{x}^T \mathbf{c}} \rho_{1/s}(\mathbf{x}).$$

Yet, it holds that $\left| \sum_{\mathbf{x} \in \mathcal{L}^*} e^{-i \cdot 2\pi \mathbf{x}^T \mathbf{c}} \rho_{1/s}(\mathbf{x}) - 1 \right| \leq \rho_{1/s}(\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon$, as $s \geq \eta_\varepsilon(\mathcal{L})$. Since the sum is a positive real, it yields that the latter is bounded below by $1 - \varepsilon$. Thence,

$$\rho_{s,\mathbf{c}}(\mathcal{L}) \geq s^d (\mathsf{Vol}\ \mathcal{L})^{-1} (1 - \varepsilon).$$

Since $\rho_{s,\mathbf{c}}(\mathbf{x}) \leq 1$ for all $\mathbf{x} \in \mathcal{L}$, we have that $H_\infty(\mathcal{D}_{\mathcal{L},s,\mathbf{c}}) \geq \log_2 \rho_{s,\mathbf{c}}(\mathcal{L})$, which gives the desired inequality. When $\mathcal{L} = \mathbb{Z}^d$ and $\varepsilon \leq 1/2$, we have $\mathsf{Vol}\ \mathcal{L} = 1$ and $\log_2(1-\varepsilon) \geq -1$, which yields the claim. □

C.1.1 Algebraic Number Theory.

Another way to embed R (or its fraction field K) is the canonical embedding, which we denote by σ. More precisely, K

has exactly n field homomorphisms $\sigma_1, \ldots, \sigma_n$ from K to $\mathbb{C}$ which are characterized by the fact that each σ_i maps ζ to one of the distinct roots α_i of the minimal polynomial of ζ. The canonical embedding of K is then the ring homomorphism $\sigma(\cdot) = [\sigma_1(\cdot)| \ldots |\sigma_n(\cdot)]^T$ from K to $\mathbb{C}^n$ (with entry-wise addition and multiplication of vectors). The canonical and coefficient embeddings are linked linearly by the Vandermonde matrix $\mathbf{V}$ of the α_i, i.e., $\sigma(\cdot) = \mathbf{V}\tau(\cdot)$ with $\mathbf{V} = [\alpha_i^{j-1}]_{i,j\in[n]}$. We note that since the α_i are the n-th roots of -1, they all have magnitude 1. Additionally, in this power-of-two cyclotomic ring, $\mathbf{P} = \mathbf{V}/\sqrt{n}$ is a unitary matrix, i.e., $\mathbf{P}^H\mathbf{P} = \mathbf{I}_n$.

Similarly to the coefficient embedding, we can define a multiplication matrix map in the canonical embedding. More precisely, we have $\sigma(rs) = M_\sigma(r)\sigma(s)$ where $M_\sigma(r) = \mathrm{diag}(\sigma_1(r), \ldots, \sigma_n(r)) \in \mathbb{C}^{n\times n}$. The link between σ and τ implies that $M_\tau(\cdot) = \mathbf{V}^{-1}M_\sigma(\cdot)\mathbf{V} = \mathbf{P}^H M_\sigma(\cdot)\mathbf{P}$.

One result which we need for our aggregate signature is that the weighted sum of discrete Gaussian vectors over R is also a discrete Gaussian. The result is due to [MP13, Thm. 3.3] which was adapted to the ring setting in [BTT22, Lem. 2.7]. The latter is however formulated with constraints in the canonical embedding σ with respect to the rescaled norm $\|\sigma(\cdot)\|_2/\sqrt{n}$, yielding the same distribution as Gaussians in the coefficient embedding. The proof is exactly the same but we adapt the lemma statement to use the coefficient embedding in the constraints instead. This just relies on the fact that in power-of-two cyclotomic fields, we have $\left\|\sqrt{zz^*}\right\| = \|\sigma(z)\|_2/\sqrt{n}$ $(= \|\tau(z)\|_2)$ and $M_\tau(\sum_{i\in[N]} z_i z_i^* \cdot s^2) = \sum_{i\in[N]} s^2 M_\tau(z_i)M_\tau(z_i)^T$, where the left-hand side of the equations are the notations from [BTT22].

Lemma C.2 (Adapted from [BTT22, Lem. 2.7]). *Let d and N be positive integers. Let $e_1, \ldots, e_N$ be arbitrary elements of R, and $s > 0$ such that $s \geq \sqrt{2}\eta_\delta(\mathbb{Z}^{nd}) \cdot \max_{j\in[N]}\|\tau(e_j)\|_2$ for a negligible δ. Then it holds that*

$$\Delta\left(\sum_{i\in[N]} e_i\mathcal{D}_{R^d,s}, \mathcal{D}_{\mathcal{L}_e,\sqrt{\mathbf{S}}}\right) \leq \mathsf{negl}(\lambda),$$

where $\mathbf{S} = \mathbf{I}_d \otimes \sum_{i\in[N]} s^2 M_\tau(e_i)M_\tau(e_i)^T$ and $\mathcal{L}_e = \sum_{i\in[N]} e_i R^d$ is a submodule of R^d.

C.1.2 General Forking Lemma.

We give here the general forking lemma from Bellare and Neven [BN06] in Lemma C.3 and the forking algorithm $\mathcal{F}_\mathcal{B}$ in Algorithm C.1. We later need this result to prove the security of our aggregate signature scheme in Section C.4.

Lemma C.3 ([BN06, Lem. 1]). *Let Q_e be a positive integer and $\mathcal{C}$ a set of size at least 2. Let $\mathcal{B}$ be a randomized algorithm that on input $x, h_1, \ldots, h_{Q_e}$ returns a pair consisting of an integer in $\{0, \ldots, Q_e\}$ and a second element referred to as a side output. Let IG be a randomized algorithm that we call input generator.*

We define the accepting probability as

$$\mathsf{acc} = \mathbb{P}[j \geq 1 : x \leftarrow \mathsf{IG}; h_1, \ldots, h_{Q_e} \hookleftarrow U(\mathcal{C}); (j, \mathsf{out}) \leftarrow \mathcal{B}(x, h_1, \ldots, h_{Q_e})].$$

The forking algorithm $\mathcal{F}_{\mathcal{B}}$ associated to $\mathcal{B}$ takes as input x and is described in Algorithm C.1. We define the probability

$$\mathsf{frk} = \mathbb{P}[b = 1 : x \leftarrow \mathsf{IG}; (b, \mathsf{out}, \mathsf{out}') \leftarrow \mathcal{F}_{\mathcal{B}}(x)].$$

Then, it holds that $\mathsf{acc} \leq Q_e/|\mathcal{C}| + \sqrt{Q_e \cdot \mathsf{frk}}$

Algorithm C.1: Forking $\mathcal{F}_{\mathcal{B}}$

On input x, proceed as follows.

1. Pick random coins ρ for $\mathcal{B}$
2. $h_1, \ldots, h_{Q_e} \hookleftarrow U(\mathcal{C})$
3. $(j, \mathsf{out}) \leftarrow \mathcal{B}(x, h_1, \ldots, h_{Q_e}; \rho)$
4. **if** $j = 0$, **return** $(0, \perp, \perp)$
5. $h'_j, \ldots, h'_{Q_e} \hookleftarrow U(\mathcal{C})$
6. $(j', \mathsf{out}') \leftarrow \mathcal{B}(x, h_1, \ldots, h_{j-1}, h'_j, \ldots, h'_{Q_e}; \rho)$
7. **if** $(j = j') \wedge (h_j \neq h'_j)$, **return** $(1, \mathsf{out}, \mathsf{out}')$
8. **else return** $(0, \perp, \perp)$.

C.2 Aggregate Signature Schemes

An aggregate signature is a regular signature scheme $\{\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}\}$ which also enables public aggregation of different signatures on different messages and under different signing keys. The regular signature is thus completed with two algorithms $\mathsf{AggSign}$ and $\mathsf{AggVerify}$. The former takes as input a sequence of messages $(\mathbf{m}_i)_{i\in[N]}$, of public keys $(\mathsf{pk}_i)_{i\in[N]}$ and of signatures $(\mathsf{sig}_i)_{i\in[N]}$ of said messages under the corresponding keys, and outputs a single signature $\mathsf{sig}_{\mathsf{agg}}$. The $\mathsf{AggVerify}$ algorithm then takes the same inputs except that it gets $\mathsf{sig}_{\mathsf{agg}}$ instead of the individual signatures, and returns 1 if the aggregate signature is valid and 0 otherwise. An aggregate signature scheme is expected to be correct, i.e., honestly generated signatures and aggregate signatures verify using Verify and $\mathsf{AggVerify}$ respectively, and secure in a security model introduced by [BGLS03] which we recall in Section C.4.

The goal of aggregate signatures is to perform batch verification of several independent signatures, albeit sharing the same public parameters. The naive solution is to define $\mathsf{sig}_{\mathsf{agg}}$ as the concatenation of the $(\mathsf{sig}_i)_{i\in[N]}$ and perform verification individually but the resulting construction is meaningless, except perhaps to show that aggregate signatures trivially exist. In practice, we are therefore interested in aggregate signature schemes that perform better than the naive concatenation.

As explained above, several aggregate signatures gathering such features have been proposed in the classical setting, but it was yet open to propose a post-quantum construction. A first attempt over lattices was proposed by Döroz et al. [DHSS20], but had major drawbacks either in performance (MMSA) or

security (MMSAT/MMSATK), and was based on a non-standard assumption called Vandermonde-SIS (or Partial Fourier Recovery). Boudgoust and Roux-Langlois [BR21] then proposed another lattice-based aggregate signature based on the FSwA paradigm, which unfortunately ended up being larger than the trivial concatenation. One explanation of this lack of compression is the half aggregation and the peculiarities of aggregate signatures which in the end make the parameters slightly worse than for the standalone signature. In particular, FSwA signatures are composed of two parts $(\mathsf{sig}_1, \mathsf{sig}_2)$ and only one of them can be aggregated, i.e., the aggregate signature is of the form $\mathsf{sig}_{\mathsf{agg}} = (\mathsf{sig}_1, (\mathsf{sig}_{2,i})_{i\in[N]})$ where $(\mathsf{sig}_{1,i}, \mathsf{sig}_{2,i})_{i\in[N]}$ are the signatures to be aggregated. Unfortunately, one needs larger parameters to prove the security of the aggregate signature scheme. As a result the size of the non-aggregated part $\mathsf{sig}_{2,i}$ becomes larger than the size of a full FSwA signature with the smaller parameters. Hence, $\mathsf{sig}_{\mathsf{agg}}$ is always larger than the concatenation of standalone signatures in the case of [BR21], regardless of the value of N.

We now present a lattice-based aggregate signature scheme that supports public aggregation, whose security is proven in the aggregate chosen-key model based on standard (module) lattice assumptions, and that performs better than the naive solution. This answers positively to the open problem left by Boudgoust and Roux-Langlois in [BR21], and provides, to the best of our knowledge, the first post-quantum aggregate signature combining all such features.

C.3 Our Construction

Our aggregate signature scheme is based on the GPV hash-and-sign framework [GPV08], with MP trapdoors [MP12] and the LW preimage sampling algorithm [LW15] with our new parameter analysis presented in Section 3. We present our scheme over module lattices.

As explained in Section 4, the combination of the GPV signature and MP trapdoors produces signatures $\mathsf{sig} = \mathbf{v}$ on messages $\mathbf{m}$ by sampling the preimage $\mathbf{v}$ of $\mathcal{H}(\mathbf{m})$ by $[\mathbf{A}|\mathbf{G} - \mathbf{AR}]$ mod qR. The function $\mathcal{H}$ is modeled by a random oracle, the matrix $\mathbf{A}$ is part of the public key, while $\mathbf{R}$ is a short matrix constituting the secret key. The matrix $\mathbf{B} = \mathbf{AR}$ is also part of the public key. For different users, each user i would have a set of keys $\mathsf{pk}_i = (\mathbf{A}_i, \mathbf{B}_i = \mathbf{A}_i\mathbf{R}_i)$ and $\mathsf{sk}_i = \mathbf{R}_i$. An intuitive way of aggregating signatures sig_i is to sum them, but this becomes tricky when the public matrices involved in verification, i.e., $[\mathbf{A}_i|\mathbf{G} - \mathbf{B}_i]$, are all different. We can however force all the $\mathbf{A}_i$ to be the same matrix $\mathbf{A}$ for all i, making sure $\mathbf{A}$ is honestly generated, i.e., without embedding an illicit trapdoor. This can for example be done by setting $\mathbf{A}$ as the hash of some public parameters. Each user would thus share the same $\mathbf{A}$ and would have their own public key $\mathbf{B}_i = \mathbf{AR}_i$. Hence, by summing the verification equations, we would obtain $\mathbf{A} \cdot \sum_{i\in[N]} \mathbf{v}_{1,i} + \sum_{i\in[N]}(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} = \sum_{i\in[N]} \mathcal{H}(\mathbf{m}_i)$. The aggregate signature could then be $(\sum_i \mathbf{v}_{1,i}, (\mathbf{v}_{2,i})_i)$, meaning we would only be aggregating the $\mathbf{v}_{1,i}$ and providing the individual $\mathbf{v}_{2,i}$.

As in the previous attempts [DHSS20,BR21], it seems difficult to achieve full aggregation due to the fact that $\mathbf{v}_{2,i}$ faces $\mathbf{B}_i$, which must differ for every

user. As a result, the bit size of the first half $\sum_i \mathbf{v}_{1,i}$ would grow logarithmically with N, while that of the second half $(\mathbf{v}_{2,i})_i$ would grow linearly with N. Similarly to FSwA signatures, as described in Section C.2, if the increased complexity of aggregate signature security results in $\mathbf{v}_{2,i}$ being larger that a full standalone signature $(\mathbf{v}_1, \mathbf{v}_2)$, the aggregate signature scheme would be vacuous. Fortunately, based on our new assessment, the preimage sampler recalled in Section 3.2 moves the bulk of the signatures in the $\mathbf{v}_{1,i}$ while minimizing the size of $\mathbf{v}_{2,i}$ which makes the concatenation of the $\mathbf{v}_{2,i}$ minimal. It therefore amortizes the linear cost of the aggregate signature, and each $\mathbf{v}_{2,i}$ in the aggregate signature stays sufficiently below the size of a full LW^* signature to allow for relevant compression.

Unfortunately, this aggregate signature is not secure as it is. Indeed, user j can produce a forged aggregate signature on behalf of the set of users $1, \ldots, N$ as follows:

1. Select a set of messages $\mathbf{m}_i$, for $i \in [N]$.
2. Select $\mathbf{v}_{2,i}$, for $i \neq j$, distributed as in a normal signature.
3. Compute $\mathbf{v}_{2,j}$ such that $\mathbf{G}\mathbf{v}_{2,j} = -\sum_{i \neq j}(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} + \sum_{i \in [N]} \mathcal{H}(\mathbf{m}_i)$.
4. Set $\mathbf{v}_1 = \mathbf{R}_j \mathbf{v}_{2,j}$.

The resulting aggregate signature $(\mathbf{v}_1, (\mathbf{v}_{2,i})_i)$ is indeed valid on $(\mathbf{m}_i)_i$ under public keys $(\mathbf{B}_i)_i$ since

$$\begin{aligned}
\mathbf{A} \cdot \mathbf{v}_1 + \sum_{i \in [N]} (\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} &= \mathbf{A} \cdot \mathbf{v}_1 + (\mathbf{G} - \mathbf{B}_j)\mathbf{v}_{2,j} + \sum_{i \neq j} (\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} \\
&= \mathbf{G}\mathbf{v}_{2,j} + \sum_{i \neq j} (\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} \\
&= \sum_{i \in [N]} \mathcal{H}(\mathbf{m}_i).
\end{aligned}$$

Intuitively, the problem stems from the fact that the rogue signer is able to compute its own signature after seeing/selecting the other components. It can thus use its own trapdoor to select a preimage that will cancel all these components. To solve this problem, we rely on a countermeasure reminiscent of the one used against rogue key attacks. We tweak the verification equation with small random weights e_i that deterministically depend on the full set $\{(\mathbf{m}_i, \mathbf{v}_{2,i}, \mathbf{B}_i)\}_i$. This therefore forces the adversary to commit to each $\mathbf{v}_{2,i}$ before seeing the verification equation it must satisfy, which thwarts the previous attack.

However, if we follow the standard approach where $e_i \leftarrow \mathcal{H}(\mathbf{B}_1, \mathbf{v}_{2,1}, \mathbf{m}_1, \ldots, \mathbf{B}_N, \mathbf{v}_{2,N}, \mathbf{m}_N, i)$ for some hash function $\mathcal{H}$, we will end up with the same problem as in [BR21]: we could only ensure unforgeability for the last signature (the one generated under public key $\mathbf{B}_N$). This has led the authors in [BR21] to use a specific security model, where the challenge key must necessarily be the last one, but the real-world security assurances provided by this model are questionable. Informally, the problem is related to the forking lemma: at some point in the security proof we need to rewind and change the weight e_j associated

with the challenge public key $\mathbf{B}_j$. However, the proof works only if e_j is the last weight to be queried to the random oracle, hence the restriction in the model of [BR21]. Otherwise, the adversary could change the other weights after the rewinding, which would completely invalidate the proof strategy. Here, we stress that one cannot simply run the simulation several times until this event (e_j is the last queried weight) happens because j is known to the adversary (it is the index corresponding to the challenge public key). Therefore, an adversary could systematically initiate its queries with e_j, leading this probabilistic approach to fail.

We show that we can circumvent this issue at almost no cost by generating the small elements e_i in two steps. Concretely, we first compute f as the output of hash function $\mathcal{H}_f$ taking as input $\{\mathbf{B}_j, \mathbf{v}_{2,j}, \mathbf{m}_j\}_j$. The output space is denoted by F but there are no restrictions on it because f is then fed to another random oracle. The only constraint is that $|F|$ must be exponential in the security parameter to avoid simple guessing or collision-finding attacks. Then, each e_i is generated as the output of another hash function $\mathcal{H}_e$ run on (f, i). Here, the output of the random oracle shall be small polynomials. We typically use ternary polynomials e_i with fixed Hamming weight, i.e., in $\mathcal{C} = \{e \in S_1 : \|e\|_1 = w\}$. Intuitively, resorting to two successive random oracles $\mathcal{H}_f, \mathcal{H}_e$ enables the simulation to anticipate the weight queries and, more importantly, to control their order. This way, we can rely on the forking lemma without placing any contrived restrictions on the model, at the cost of only one hash evaluation for the whole aggregate signature.

The sampler from [LW15] given in Algorithm 3.1 can be instantiated so that it samples the $\mathbf{v}_{1,i}$ close to a Gaussian distribution, which is the object of Corollary 3.1. Although [LW15] can be used for a broader class of distributions such as uniform over a hypercube, the properties of Gaussian distributions lead to tighter verification bounds and in turn a smaller M-SIS bounds and thus smaller parameters. More precisely, the weighted sum $\mathbf{v}_1 = \sum_{i\in[N]} e_i\mathbf{v}_{1,i}$ follows a Gaussian distribution by Lemma C.2, and the tail bound thus gives $\|\mathbf{v}_1\|_2 \leq w \cdot \sqrt{N} \cdot s\sqrt{2nd}$. For other distribution, one would use the triangle inequality and get $\|\mathbf{v}_1\|_2 \leq \sum_{i\in[N]} \|e_i\|_1 \|\mathbf{v}_{1,i}\|_2 \leq w \cdot N \cdot B$ where B would be the norm bound on each $\mathbf{v}_{1,i}$ for a single signature. The dependency in N is therefore optimized in the case of Gaussian distributions.

Finally, as in Section 4, we can consider the matrix $\mathbf{A}$ in Hermite Normal Form, i.e., $\mathbf{A} = [\mathbf{I}_d|\mathbf{A}']$ with $\mathbf{A}' \sim U(R_q^{d\times d})$. If each $\mathbf{v}_{1,i}$ is parsed as $[\mathbf{v}_{1,1,i}^T|\mathbf{v}_{1,2,i}^T]^T$ with $\mathbf{v}_{1,1,i}, \mathbf{v}_{1,2,i} \in R^d$, this allows us to only aggregate the $\mathbf{v}_{1,2,i}$ as $\mathbf{v}_{1,2} = \sum_{i\in[N]} e_i\mathbf{v}_{1,2,i}$. The other part, i.e., $\mathbf{v}_{1,1} = \sum_{i\in[N]} e_i\mathbf{v}_{1,1,i}$ can be recovered during verification as

$$\mathbf{v}_{1,1} = \sum_{i\in[N]} e_i\mathcal{H}(\mathbf{m}_i) - \mathbf{A}'\mathbf{v}_{1,2} - \sum_{i\in[N]} e_i(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i}.$$

Although this does not have a tremendous impact on the aggregate signature size when N is large, as the bulk of it is due to the concatenation of the $\mathbf{v}_{2,i}$, it

leads to a more compact signature and gives a fair comparison with concatenated LW* signatures.

The Scheme. In what follows, we work over the $2n$-th cyclotomic ring denoted by R for n a power of two, as defined in Section 2.3. Although we have seen that the optimal base for the sampler from [LW15] seems to be $b = 2$, we present the scheme for an arbitrary b to be more general. The aggregate signature is described by Algorithms C.2 to C.7. We present it in the stateful version but it can be made stateless by using salts as usual. The only caveat is that the salts cannot be aggregated. As it remains far below the size of $\mathbf{v}_2$, it would be acceptable. Below, we let c' be a slack which implicitly depends on $2nd$ and λ. It is set so that the tail bound from Lemma 2.1 with $c = c'\sqrt{2\pi}$ is verified with probability $1 - 2^{-4\lambda}$. For $n = 256, d = 7, \lambda = 128$, we have $c' \approx 0.53$.

Algorithm C.2: Setup

Input: Security parameter λ, Maximal number of signers N.

1. Choose positive integers d, q, w, b.
2. $\mathcal{C} \leftarrow \{e \in S_1 : \|e\|_1 = w\}$. ▷ Hash space for weights, such that $|\mathcal{C}| \geq 2^{2\lambda}$
3. $k \leftarrow \lceil \log_b(\lceil (q-1)/2 \rceil + 1) \rceil$.
4. $\mathbf{G} = \mathbf{I}_d \otimes [1 \cdots b^{k-1}] \in R_q^{d \times dk}$. ▷ Gadget matrix
5. $t \leftarrow \sqrt{\lambda/(\pi \log_2 e)}$. ▷ $t \approx 5.4$
6. $\varepsilon \leftarrow 1/Q$ ▷ Rejection sampling loss
7. Choose $M > 1$. ▷ Repetition rate
8. $\alpha \leftarrow \frac{\sqrt{\pi}}{\ln M}(\sqrt{\ln \varepsilon^{-1} + \ln M} + \sqrt{\ln \varepsilon^{-1}})$. ▷ Rejection sampling slack
9. $s \leftarrow \max(\alpha(b-1)\sqrt{ndk}(\sqrt{2nd} + \sqrt{ndk} + t), \sqrt{2w}\eta_\delta(\mathbb{Z}^{2nd}))$. ▷ Width
10. $\mathbf{A}' \hookleftarrow U(R_q^{d \times d})$.

Output: $\mathsf{pp} = (\mathbf{A}'; \mathbf{G}; \lambda, N, n, q, d, b, k, w, s, M)$.

Algorithm C.3: KeyGen

Input: Public parameters pp as in Algorithm C.2.

1. $\mathbf{R} \hookleftarrow U(S_1^{2d \times dk})$ such that $\|\mathbf{R}\|_2 \leq \sqrt{2nd} + \sqrt{ndk} + t$.
2. $\mathbf{B} \leftarrow [\mathbf{I}_d | \mathbf{A}']\mathbf{R} \bmod qR \in R_q^{d \times dk}$

Output: $\mathsf{pk} = \mathbf{B}$, and $\mathsf{sk} = \mathbf{R}$. ▷ pp stored with pk for simplicity

Algorithm C.4: Sign

Input: Secret key sk, Message $\mathbf{m} \in \{0,1\}^*$, Public key pk.

1. **if** $(\mathbf{m}, \mathbf{v})$ is stored **then** look-up $\mathbf{v}$
2. **else** $\mathbf{v} \leftarrow \mathsf{SamplePre}(\mathbf{R}; \mathbf{A}', \mathbf{I}_d, \mathcal{H}(\mathbf{m}), s)$. ▷ Algorithm 3.1
3. Store $\mathbf{v}$. Parse $\mathbf{v}$ as $[\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T | \mathbf{v}_2^T]^T$ with $\mathbf{v}_{1,1}, \mathbf{v}_{1,2} \in R^d$ and $\mathbf{v}_2 \in R^{dk}$.

Output: $\mathsf{sig} = (\mathbf{v}_{1,2}, \mathbf{v}_2)$.

Algorithm C.5: Verify

Input: Public key pk, Message $\mathbf{m} \in \{0,1\}^*$, Signature sig.

1. $\mathbf{v}_{1,1} \leftarrow \mathcal{H}(\mathbf{m}) - \mathbf{A}'\mathbf{v}_{1,2} - (\mathbf{G} - \mathbf{B})\mathbf{v}_2 \in R^d$
2. $\mathbf{v}_1 \leftarrow [\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T]^T \in R^{2d}$.
3. $\gamma \leftarrow (\|\mathbf{v}_1\|_2 \leq c' s\sqrt{2nd}) \wedge (\|\mathbf{v}_2\|_\infty \leq b-1)$

Output: γ. ▷ $\gamma = 1$ if valid, 0 otherwise

Algorithm C.6: AggSign

Input: Public keys $(\mathbf{B}_i)_{i\in[N]}$, Signatures $(\mathbf{v}_{1,2,i}, \mathbf{v}_{2,i})_{i\in[N]}$, Messages $(\mathbf{m}_i)_{i\in[N]}$

1. $f \leftarrow \mathcal{H}_f(\mathbf{B}_1, \mathbf{v}_{2,1}, \mathbf{m}_1, \ldots, \mathbf{B}_N, \mathbf{v}_{2,N}, \mathbf{m}_N) \in F$ ▷ $|F| \geq |\mathcal{C}| \geq 2^{2\lambda}$
2. $\forall i \in [N], e_i \leftarrow \mathcal{H}_e(f, i) \in \mathcal{C}$.
3. $\mathbf{v}_{1,2} \leftarrow \sum_{i\in[N]} e_i \mathbf{v}_{1,2,i}$.

Output: $\mathsf{sig}_{\mathsf{agg}} = (\mathbf{v}_{1,2}, (\mathbf{v}_{2,i})_{i\in[N]})$.

Algorithm C.7: AggVerify

Input: Public keys $(\mathbf{B}_i)_{i\in[N]}$, Aggregate Signature $(\mathbf{v}_{1,2}, (\mathbf{v}_{2,i})_{i\in[N]})$, Messages $(\mathbf{m}_i)_{i\in[N]}$

1. $f \leftarrow \mathcal{H}_f(\mathbf{B}_1, \mathbf{v}_{2,1}, \mathbf{m}_1, \ldots, \mathbf{B}_N, \mathbf{v}_{2,N}, \mathbf{m}_N) \in F$
2. $\forall i \in [N], e_i \leftarrow \mathcal{H}_e(f, i) \in \mathcal{C}$.
3. $\mathbf{v}_{1,1} \leftarrow \sum_{i\in[N]} e_i \mathcal{H}(\mathbf{m}_i) - \mathbf{A}'\mathbf{v}_{1,2} - \sum_{i\in[N]} e_i(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i}$
4. $\mathbf{v}_1 \leftarrow [\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T]^T \in R^{2d}$.
5. $\gamma_1 \leftarrow (\|\mathbf{v}_1\|_2 \leq c' w s \sqrt{N \cdot 2nd})$.
6. $\gamma_2 \leftarrow (\forall i \in [N], \|\mathbf{v}_{2,i}\|_\infty \leq b - 1)$

Output: $\gamma_1 \wedge \gamma_2$. ▷ 1 if valid, 0 otherwise

We give prove the correctness of our scheme in the following theorem.

Theorem C.1 (Correctness). *The aggregate signature scheme (Setup, KeyGen, Sign, Verify, AggSign, AggVerify) described in Section C.3 is correct. Formally, for all security parameters λ and number of signers N, the following hold.*

Single signature correctness. *For all* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, N)$, *for all* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, *for all* $\mathbf{m} \in \{0,1\}^*$,

$$\mathbb{P}[\mathsf{Verify}(\mathsf{pk}, \mathbf{m}, \mathsf{Sign}(\mathsf{sk}, \mathbf{m}; \mathsf{pk})) \neq 1] < \frac{2^{-4\lambda}}{1-\varepsilon} = \mathsf{negl}(\lambda).$$

Aggregate signature correctness. *For all* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, N)$, *for all* $i \in [N]$ *and for all* $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathbf{m}_i \in \{0,1\}^*$, $\mathsf{sig}_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, \mathbf{m}_i; \mathsf{pk}_i)$,

$$\mathbb{P}[E] < \left(2^{-4\lambda} + \mathsf{negl}(\lambda)\right) \cdot \frac{1}{(1-\varepsilon)^N} = \mathsf{negl}(\lambda),$$

where $E = \{\mathsf{AggVerify}(\mathbf{PK}, \mathsf{AggSign}(\mathbf{PK}, \mathbf{SIG}, \mathbf{M}), \mathbf{M}) \neq 1\}$, *and* $\mathbf{PK} = (\mathsf{pk}_i)_{i\in[N]}$, $\mathbf{SIG} = (\mathsf{sig}_i)_{i\in[N]}$ *and* $\mathbf{M} = (\mathbf{m}_i)_{i\in[N]}$.

Proof. We first look at the single signature correctness. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, N)$, $(\mathbf{B}, \mathbf{R}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathbf{m} \in \{0,1\}^*$, and $(\mathbf{v}_{1,2}, \mathbf{v}_2) \leftarrow \mathsf{Sign}(\mathbf{R}, \mathbf{m}; \mathbf{B})$. We reconstruct $\mathbf{v}_{1,1} \leftarrow \mathcal{H}(\mathbf{m}) - \mathbf{A}'\mathbf{v}_{1,2} - (\mathbf{G} - \mathbf{B})\mathbf{v}_2$ and $\mathbf{v}_1 = [\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T]^T$. It thus holds that $[\mathbf{v}_1^T | \mathbf{v}_2^T]^T$ was obtained using $\mathsf{SamplePre}$. Using the parameters of Algorithm C.2, Corollary 3.1 gives that $RD_\infty(\mathbf{v}_1 \| \mathcal{D}_{R^{2d},s}) \leq 1/(1-\varepsilon)$ obtained when $a = +\infty$. By the probability preservation, we have

$$\begin{aligned}\mathbb{P}_{\mathbf{v}_1}[\|\mathbf{v}_1\|_2 > c's\sqrt{2nd}] &\leq \mathbb{P}_{\mathbf{v}_1' \sim \mathcal{D}_{R^{2d},s}}[\|\mathbf{v}_1'\|_2 > c's\sqrt{2nd}] \cdot RD_\infty(\mathbf{v}_1 \| \mathcal{D}_{R^{2d},s}) \\ &\leq \frac{2^{-4\lambda}}{1-\varepsilon},\end{aligned}$$

where the last inequality holds by Lemma 2.1 and definition of c'. Additionally, by construction it holds that $\mathbf{v}_2 \in S_{b-1}^{dk}$. We then get

$$\mathbb{P}[\mathsf{Verify}(\mathbf{B}, \mathbf{m}, \mathbf{v}) = 1] \geq 1 - \frac{2^{-4\lambda}}{1-\varepsilon} = 1 - \mathsf{negl}(\lambda).$$

Let us now investigate the correctness of our aggregate signature. Let $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, N)$, and for all $i \in [N]$ let $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathbf{m}_i \in \{0,1\}^*$, $\mathsf{sig}_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, \mathbf{m}_i; \mathsf{pk}_i)$. Let $\mathsf{sig}_{\mathsf{agg}} \leftarrow \mathsf{AggSign}(\mathbf{PK}, \mathbf{SIG}, \mathbf{M})$ and parse it as $(\mathbf{v}_{1,2}, (\mathbf{v}_{2,i})_{i\in[N]})$. From the single signature correctness above, we directly have that $\gamma_2 = 1$, namely that $\mathbf{v}_{2,i} \in S_{b-1}^{dk}$ for all $i \in [N]$.

We reconstruct $\mathbf{v}_{1,1} \leftarrow \sum_{i\in[N]} e_i \mathcal{H}(\mathbf{m}_i) - \mathbf{A}'\mathbf{v}_{1,2} - \sum_{i\in[N]} e_i(\mathbf{G}-\mathbf{B})\mathbf{v}_{2,i}$ and $\mathbf{v}_1 = [\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T]^T$. Since the signatures were honestly generated, it holds that $\mathbf{v}_1 = \sum_{i\in[N]} e_i \mathbf{v}_{1,i}$ where $[\mathbf{v}_{1,i}^T | \mathbf{v}_{2,i}^T]^T$ was obtained using $\mathsf{SamplePre}$.

We now look at the norm bound on $\mathbf{v}_1$. The idea is that $\mathbf{v}_1$ behaves as a discrete Gaussian over a lattice that depends on the weights e_i and its covariance depends on the size of the e_i. Using the Gaussian tail bound of Lemma 2.1 yields the correct bound. We now give more details. First, since $\mathbf{v}_1$ is a weighted sum of discrete Gaussian vectors, Lemma C.2 yields

$$\Delta(\sum_{i\in[N]} e_i \mathcal{D}_{R^{2d},s}, \mathcal{D}_{\sum_{i\in[N]} e_i R^{2d}, \sqrt{\mathbf{S}}}) \leq \mathsf{negl}(\lambda),$$

where $\mathbf{S} = \mathbf{I}_{2d} \otimes \sum_{i\in[N]} s^2 M_\tau(e_i) M_\tau(e_i)^T$, as long as the Gaussian width verifies $s \geq \sqrt{2}\eta_\delta(\mathbb{Z}^{2nd}) \cdot \max_{i\in[N]} \|\tau(e_i)\|_2$. Since the $\tau(e_i)$ are ternary vectors with Hamming weight w, we have $\|\tau(e_i)\|_2 = \sqrt{w}$ The condition thus becomes $s \geq \sqrt{2w}\eta_\delta(\mathbb{Z}^{2nd})$, which is encompassed by our parameter choice. Then, using the fact that each e_i has weight $w \neq 0$, it holds that $e_i \neq 0$ in the field K and in turn that all the $M_\tau(e_i)$ are invertible. As a result, the final covariance matrix $\mathbf{S}$ is positive definite. Using [GMPW20, Lem. 2.3], we obtain that

$$\mathcal{D}_{\sum_{i\in[N]} e_i R^{2d}, \sqrt{\mathbf{S}}} = \sqrt{\mathbf{S}} \mathcal{D}_{\sqrt{\mathbf{S}}^{-1} \sum_{i\in[N]} e_i R^{2d}, 1},$$

and we can therefore apply Lemma 2.1 and get

$$\begin{aligned}
&\mathbb{P}_{\mathbf{v}_1 \sim \mathcal{D}_{\sum_i e_i R^{2d}, \sqrt{\mathbf{S}}}} \left[\|\mathbf{v}_1\|_2 > c' \left\|\sqrt{\mathbf{S}}\right\|_2 \sqrt{2nd} \right] \\
&\qquad = \mathbb{P}_{\mathbf{x} \sim \mathcal{D}_{\sqrt{\mathbf{S}}^{-1} \sum_i e_i R^{2d}, 1}} \left[\left\|\sqrt{\mathbf{S}}\mathbf{x}\right\|_2 > c' \left\|\sqrt{\mathbf{S}}\right\|_2 \sqrt{2nd} \right] \\
&\qquad \leq \mathbb{P}_{\mathbf{x} \sim \mathcal{D}_{\sqrt{\mathbf{S}}^{-1} \sum_i e_i R^{2d}, 1}} \left[\|\mathbf{x}\|_2 > c' \sqrt{2nd} \right] \\
&\qquad \leq 2^{-4\lambda},
\end{aligned}$$

where the first inequality follows by inclusion of events. We now only need to bound $\left\|\sqrt{\mathbf{S}}\right\|_2$. The latter corresponds to $\sqrt{\lambda_{\max}(\mathbf{S})}$ which itself equals $\sqrt{\lambda_{\max}(\mathbf{S}')}$

with $\mathbf{S}' = s^2 \sum_{i\in[N]} M_\tau(e_i) M_\tau(e_i)^T$, and where $\lambda_{\max}$ denotes the largest eigenvalue. Recalling from Section 2.3 that $M_\tau = \mathbf{P}^H M_\sigma \mathbf{P}$ with $\mathbf{P}$ a unitary matrix, we get

$$\mathbf{S}' = s^2 \mathbf{P}^H \mathrm{diag}\left(\sum_{i\in[N]} |\sigma_1(e_i)|^2, \ldots, \sum_{i\in[N]} |\sigma_n(e_i)|^2 \right) \mathbf{P},$$

where the σ_i are the individual field embeddings. It thus proves that

$$\lambda_{\max}(\mathbf{S}') = s^2 \max_{k\in[n]} \sum_{i\in[N]} |\sigma_k(e_i)|^2.$$

For all (k,i), we have $|\sigma_k(e_i)| \leq \|\sigma(e_i)\|_\infty = \|M_\sigma(e_i)\|_2 = \|M_\tau(e_i)\|_2$, where the first equality is due to the diagonal form of M_σ, and the last equality is due to [BJRW23, Lem. 2.3]. Due to the specific form of $M_\tau(e_i)$ as described in Section 2.3, it holds by e.g. [BJRW23, Lem. 2.2] that $\|M_\tau(e_i)\| \leq \|\tau(e_i)\|_1 = w$. As a result, we obtain $\lambda_{\max}(\mathbf{S}') \leq s^2 N w^2$. Combining the rejection sampling, the multiplicativity of the Rényi divergence, the weighted sum of Gaussians, the tail bound and the spectral bound on $\lambda_{\max}(\mathbf{S})$, it proves that

$$\begin{aligned}
&\mathbb{P}_{\mathbf{v}_1}[\|\mathbf{v}_1\|_2 > c' w s \sqrt{N \cdot 2nd}] \\
&\qquad \leq \mathbb{P}_{(\mathbf{v}_{1,i})_i \sim \mathcal{D}_{R^{2d},s}} \left[\left\| \sum_i e_i \mathbf{v}_{1,i} \right\|_2 > c' w s \sqrt{N \cdot 2nd} \right] \cdot \frac{1}{(1-\varepsilon)^N} \\
&\qquad \leq \left(\mathbb{P}_{\mathbf{v}_1 \sim \mathcal{D}_{\sum_i e_i R^{2d}, \sqrt{\mathbf{S}}}} [\|\mathbf{v}_1\|_2 > c' w s \sqrt{N \cdot 2nd}] + \mathsf{negl}(\lambda) \right) \cdot \frac{1}{(1-\varepsilon)^N} \\
&\qquad \leq \left(2^{-4\lambda} + \mathsf{negl}(\lambda) \right) \cdot \frac{1}{(1-\varepsilon)^N}
\end{aligned}$$

thus proving that $\gamma_1 = 1$ except with negligible probability, as $\varepsilon = 1/Q$, $N \ll Q$ and $N = \mathsf{poly}(\lambda)$. It then yields

$$\begin{aligned}
\mathbb{P}[\mathsf{AggVerify}(\mathbf{PK}, \mathbf{M}, \mathsf{sig}_{\mathsf{agg}}) = 1] &\geq 1 - \left(2^{-4\lambda} + \mathsf{negl}(\lambda) \right) \cdot \frac{1}{(1-\varepsilon)^N} \\
&= 1 - \mathsf{negl}(\lambda),
\end{aligned}$$

concluding the proof. □

C.4 Security

The *aggregate chosen-key* security model introduced by Boneh et al. [BGLS03] captures the idea that an adversary cannot produce an aggregate signature on behalf of N users, even if it colludes with (at most) $N-1$ of them. The adversary is given a challenge public key pk and the ability to query signatures on this key, and is asked to produce $N-1$ keys pk_i as well as an aggregate signature $\mathsf{sig}_{\mathsf{agg}}$ that verifies with these N public keys. We formally define this model by a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$ in three stages.

Setup Stage. $\mathcal{B}$ runs Setup and KeyGen to obtain pp, pk, and sk. It then gives pp and pk to $\mathcal{A}$.

Query Stage. $\mathcal{A}$ queries signatures on at most Q messages $\mathbf{m}^{(1)}, \ldots, \mathbf{m}^{(Q)}$, which are answered by $\mathcal{B}$ returning $\mathsf{sig}^{(i)} \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathbf{m}^{(i)}; \mathsf{pk})$.

Forgery Stage. $\mathcal{A}$ eventually provides a forgery $((\mathsf{pk}_i)_{i\in[N]}, (\mathbf{m}_i)_{i\in[N]}, \mathsf{sig}_{\mathsf{agg}})$.

The adversary wins the game if (1) there exists an $i^* \in [N]$ such that $\mathsf{pk}_{i^*} = \mathsf{pk}$, (2) for all $i \in [Q]$, $\mathbf{m}_{i^*} \neq \mathbf{m}^{(i)}$, and (3) $\mathsf{AggVerify}((\mathsf{pk}_i)_{i\in[N]}, \mathsf{sig}_{\mathsf{agg}}, (\mathbf{m}_i)_{i\in[N]}) = 1$. The adversary's advantage is defined as $\mathrm{Adv}[\mathcal{A}] = \mathbb{P}[\mathcal{A} \text{ wins}]$, where the probability is over all the random coins. We say that the aggregate signature scheme is secure in the aggregate chosen-key model if for all probabilistic polynomial time (PPT) adversary $\mathcal{A}$, $\mathrm{Adv}[\mathcal{A}]$ is negligible in the security parameter λ.

We note that in [BGLS03], the challenge key is set to be pk_1. In the context of their construction in bilinear groups, this can be assumed without loss of generality because the order of the signatures that are aggregated does not matter. In our case, each (half) signature $\mathbf{v}_{1,i}$ is multiplied by a weight $e_i = \mathcal{H}_e(f, i)$ which depends on the position i and also the order of the signatures because of $f = \mathcal{H}_f(\mathbf{B}_1, \mathbf{v}_{2,1}, \mathbf{m}_1, \ldots, \mathbf{B}_N, \mathbf{v}_{2,N}, \mathbf{m}_N)$. These weights are necessary in the lattice setting to avoid the attack we described in Section C.3. As a result, in the security proof, the challenger has to guess the position i^* of the challenge key in order to exploit the forgery to break the underlying computational assumption.

Theorem C.2 (Security). *The aggregate signature scheme (Setup, KeyGen, Sign, Verify, AggSign, AggVerify) described in Section C.3 is secure in the aggregate chosen-key model under the* M-SIS *and* M-LWE *assumptions. More formally, for any PPT adversary $\mathcal{A}$ against the aggregate chosen-key security, it holds that*

$$Adv[\mathcal{A}] \leq \frac{N}{(1-1/Q)^Q} \cdot \left(\varepsilon_{\text{M-LWE}} + \frac{Q_e}{|\mathcal{C}|} + \sqrt{Q_e \varepsilon_{\text{M-SIS}}}\right) + \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda),$$

where $\varepsilon_{\text{M-LWE}}$ is the hardness bounds of $\text{M-LWE}^{dk}_{n,d,d,q,U(S_1)}$*, and $\varepsilon_{\text{M-SIS}}$ is that of* $\text{M-SIS}_{n,d,d(2+k),q,\beta}$ *with* $\beta = \sqrt{(2w(\sqrt{N}+1)c's\sqrt{2nd})^2 + (4w(b-1)\sqrt{ndk})^2}$.

Proof. We proceed by a sequence of games that we prove indistinguishable from the aggregate chosen-key game. In the final game, we use the general forking lemma in order to deduce a solution of M-SIS. We first denote by Q the maximal number of signature queries, and by Q_e the maximal number of queries to $\mathcal{H}_e$.

Game G_0. We change the original aggregate chosen-key game by programming the random oracles in a certain way. The challenger $\mathcal{B}$ starts by sampling $i^+ \hookleftarrow U([N])$, which later acts as a guess on the position of the challenge key in the forgery. $\mathcal{B}$ is also provided with some random inputs $h_j \hookleftarrow U(\mathcal{C})$ for all $j \in [Q_e]$. Additionally, $\mathcal{B}$ keeps four tables $\mathcal{T}_s, \mathcal{T}_f, \mathcal{T}_e, \mathcal{T}_m$ that will be used to store the corresponding queries, and which are all empty at the outset of the game. Finally, it further stores an index j_e, initially set to 0.

Setup. $\mathcal{B}$ computes $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathbf{B}, \mathbf{R}) = (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$. It then sends pp, pk to $\mathcal{A}$.

Queries to $\mathcal{H}$. On input $\mathbf{m} \in \{0,1\}^*$ given by $\mathcal{A}$, $\mathcal{B}$ first checks whether $\mathbf{m}$ is already stored in $\mathcal{T}_m$. If so, it directly outputs the $\mathbf{u}$ from $\mathcal{T}_m$ corresponding to $\mathbf{m}$. If not, it samples $\mathbf{u} \hookleftarrow U(R_q^d)$, stores $(\mathbf{m}, \mathbf{u})$ in $\mathcal{T}_m$ and sends $\mathbf{u}$ to $\mathcal{A}$.

Queries to $\mathcal{H}_f$. On input $(\mathbf{B}_i, \mathbf{v}_{2,i}, \mathbf{m}_i)_{i\in[N]}$ given by $\mathcal{A}$, $\mathcal{B}$ first checks whether it already appears in $\mathcal{T}_f$. If so, it directly outputs the f in $\mathcal{T}_f$ corresponding to the input. If not, it samples $f \hookleftarrow U(F)$, stores $((\mathbf{B}_i, \mathbf{v}_{2,i}, \mathbf{m}_i)_{i\in[N]}, f)$ in $\mathcal{T}_f$ and sends f to $\mathcal{A}$. Additionally, for all $i \in [N] \setminus \{i^+\}$, $\mathcal{B}$ samples $e_i \hookleftarrow U(\mathcal{C})$ and stores (f, i, e_i) in $\mathcal{T}_e$.

Queries to $\mathcal{H}_e$. On input (f, i) given by $\mathcal{A}$, $\mathcal{B}$ first checks whether it already appears in $\mathcal{T}_e$. If so, it outputs the e_i from $\mathcal{T}_e$ corresponding to (f, i). If (f, i) does not appear in $\mathcal{T}_e$, then either f does not appear in $\mathcal{T}_f$ or $i = i^+$. Without loss of generality, we can assume that f has previously been obtained by a query to $\mathcal{H}_f$, and therefore we necessarily have $i = i^+$. Then, $\mathcal{B}$ increments j_e to $j_e + 1$ and sends h_{j_e} to $\mathcal{A}$. It also stores (f, i^+, h_{j_e}) in $\mathcal{T}_e$. Notice that $\mathcal{H}_e(f, i^+)$ is therefore set after all the other $\mathcal{H}_e(f, i)$ for $i \neq i^+$.

Signature queries. On input $\mathbf{m}$, $\mathcal{B}$ first checks if $\mathbf{m}$ appears in $\mathcal{T}_s$. If so, it outputs the $\mathbf{v}$ from $\mathcal{T}_s$ corresponding to $\mathbf{m}$. If not, it proceeds as follows. $\mathcal{B}$ checks if $\mathbf{m}$ is in $\mathcal{T}_m$. If not, it samples $\mathbf{u} \hookleftarrow U(R_q^d)$ and stores $(\mathbf{m}, \mathbf{u})$ in $\mathcal{T}_m$. Otherwise, it gets the corresponding syndrome $\mathbf{u}$. Then, it runs the legitimate signing algorithm Sign with $\mathsf{sk}, \mathsf{pk}, \mathsf{pp}$ by just replacing $\mathcal{H}(\mathbf{m})$ by $\mathbf{u}$, namely sampling $\mathbf{v} = (\mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2) \leftarrow \mathsf{SamplePre}(\mathbf{R}; \mathbf{A}', \mathbf{I}_d, \mathbf{u}, s)$. It then stores $(\mathbf{m}, \mathbf{v})$ in $\mathcal{T}_s$ and sends $(\mathbf{v}_{1,2}, \mathbf{v}_2)$ to $\mathcal{A}$.

Forgery. Eventually, $\mathcal{A}$ outputs $((\mathsf{pk}_i)_{i\in[N]}, (\mathbf{m}_i)_{i\in[N]}, \mathsf{sig}_{\mathsf{agg}})$ to $\mathcal{B}$ such that there exists $i^* \in [N]$ satisfying $\mathsf{pk}_{i^*} = \mathsf{pk}$, that $\mathbf{m}_{i^*}$ was not part of the signing queries, and such that $\mathsf{AggVerify}((\mathsf{pk}_i)_{i\in[N]}, \mathsf{sig}_{\mathsf{agg}}, (\mathbf{m}_i)_{i\in[N]}) = 1$. If these conditions are not met, then $\mathcal{B}$ outputs $(0, \perp)$. From now on, we assume that these conditions are met, which happens with probability $\mathrm{Adv}[\mathcal{A}]$ as everything is correctly distributed. Then, if $i^* \neq i^+$, then $\mathcal{B}$ also outputs $(0, \perp)$. Since i^+ is completely independent of the view of $\mathcal{A}$ as all the random oracle queries are identical as in the standard game, this happens with probability $1/N$. If $f = \mathcal{H}_f((\mathsf{pk}_i, \mathbf{v}_{2,i}, \mathbf{m}_i)_{i\in[N]})$ was not queried, then $\mathcal{A}$ would have had to guess the correct value of f to obtain the weights e_i, and thus the signature would verify with probability at most $1/|F|$. Noting that $1/|F| = \mathsf{negl}(\lambda)$, it would entail a negligible advantage for $\mathcal{A}$. So we assume that f has been queried. Similarly, if $\mathcal{H}_e(f, i^+)$ was not queried, then the probability that $\mathsf{AggVerify}$ passes is at most $1/|\mathcal{C}|$ as $\mathcal{A}$ would have had to guess the value of e_{i^+}. Since $1/|\mathcal{C}| = \mathsf{negl}(\lambda)$, then such an adversary $\mathcal{A}$ would have a negligible advantage. So we further assume, without loss of generality that $\mathcal{H}_e(f, i^+)$ was queried and is equal to some h_j for some counter index j. Then, $\mathcal{B}$ outputs (j, out) with $\mathsf{out} = ((\mathsf{pk}_i)_{i\in[N]}, (\mathbf{m}_i)_{i\in[N]}, \mathsf{sig}_{\mathsf{agg}}, (\mathcal{H}_e(f, i))_{i\in[N]})$. Further, we let p_k denote the probability that $\mathcal{B}$ does not output $(0, \perp)$ in game G_k. Here, we have

$$p_0 = \frac{1}{N}\mathrm{Adv}[\mathcal{A}]. \tag{5}$$

Game G_1. This game is identical to game G_0 except in the way signatures are generated. Instead, $\mathcal{B}$ simulates signatures without resorting to sk by using the simulator from Corollary 3.1. We thus change the way queries to $\mathcal{H}$ and signing queries are handled.

Queries to $\mathcal{H}$. On input $\mathbf{m} \in \{0,1\}^*$ given by $\mathcal{A}$, $\mathcal{B}$ first checks whether $\mathbf{m}$ is already stored in $\mathcal{T}_m$. If so, it directly outputs the $\mathbf{u}$ from $\mathcal{T}_m$ corresponding to $\mathbf{m}$. If not, it samples $\mathbf{v}_1 \hookleftarrow \mathcal{D}_{R^{2d},s}$, $\mathbf{v}_2 \hookleftarrow \mathbf{G}^{-1}(U(R_q^d))$, sets $\mathbf{v} = [\mathbf{v}_1^T|\mathbf{v}_2^T]^T \in R^{d(2+k)}$ and computes $\mathbf{u} = [\mathbf{I}_d|\mathbf{A}'|\mathbf{G}-\mathbf{B}]\mathbf{v} \bmod qR$. It rejects such a $\mathbf{v}, \mathbf{u}$ with probability $1 - 1/M$ and repeats the procedure until $\mathbf{v}, \mathbf{u}$ is kept. Then, $\mathcal{B}$ stores $(\mathbf{m}, \mathbf{u})$ in $\mathcal{T}_m$ and $(\mathbf{m}, \mathbf{v})$ in $\mathcal{T}_s$. It then sends $\mathbf{u}$ to $\mathcal{A}$.

Signature queries. On input $\mathbf{m} \in \{0,1\}^*$ given by $\mathcal{A}$, $\mathcal{B}$ first checks whether $\mathbf{m}$ is already stored in $\mathcal{T}_s$. If so, it directly outputs the $\mathbf{v}$ from $\mathcal{T}_s$ corresponding to $\mathbf{m}$. If not, it means that $\mathcal{H}$ was never queried on $\mathbf{m}$. In this case, $\mathcal{B}$ performs the query to $\mathcal{H}(\mathbf{m})$ on its own as above and fills $\mathcal{T}_m$ with $(\mathbf{m}, \mathbf{u})$ and $\mathcal{T}_s$ with $(\mathbf{m}, \mathbf{v})$. It then sends $\mathbf{v}$ to $\mathcal{A}$.

The simulation result of [LW15, Thm. 3.1] which we overhauled in Theorem 3.1 applies to the Gaussian case as stated in Corollary 3.1, yielding for $a = +\infty$

$$p_0 \leq p_1 \cdot RD_\infty(\mathcal{P}_1 \| \mathcal{P}_2)^Q \leq \frac{p_1}{(1-1/Q)^Q}. \tag{6}$$

Game G_2. Since sk is no longer used in game G_1, we define G_2 to be identical to G_1 except in the setup stage.

Setup. $\mathcal{B}$ computes $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ and samples $\mathbf{B}' \hookleftarrow U(R_q^{d \times dk})$. It then computes $\mathbf{B} \leftarrow \mathbf{G} - \mathbf{B}'$ and sets $\mathsf{pk} \leftarrow \mathbf{B}$. It then sends pp, pk to $\mathcal{A}$.

Since $\mathbf{B}'$ is uniform, then so is $\mathbf{B}$. By the M-LWE$^{dk}_{n,d,d,q,U(S_1)}$ assumption, $[\mathbf{I}_d|\mathbf{A}']\mathbf{R} \bmod qR$ in game G_1 is $\varepsilon_{\text{M-LWE}}$-indistinguishable from $\mathbf{B}$ in game G_2. As a result, it holds that

$$|p_1 - p_2| \leq \varepsilon_{\text{M-LWE}}. \tag{7}$$

Forking. We now aim at bounding p_2, using the general forking lemma recalled in Lemma C.3. We use the forking algorithm $\mathcal{F}_\mathcal{B}$ of Algorithm C.1 around $\mathcal{B}$ and we will invoke Lemma C.3. The input generator IG is defined by outputting $\overline{\mathbf{A}} = [\mathbf{I}_d|\mathbf{A}'|\mathbf{B}']$ where $[\mathbf{A}'|\mathbf{B}'] \hookleftarrow U(R_q^{d \times d(1+k)})$ and pp honestly generated (where $\mathbf{A}'$ is the same matrix as the one in pp). For clarity, we denote by $\mathbf{A}$ the matrix $[\mathbf{I}_d|\mathbf{A}']$. We call acc the accepting probability of $\mathcal{B}$, i.e., $\mathsf{acc} = p_2$, and frk the forking probability from Lemma C.3. Hence, with probability frk, the two calls to $\mathcal{B}$, and in turn $\mathcal{A}$ (which are both oblivious to the fact they are being rewound), return (j, out) and (j', out') with $j = j' \neq 0$ and $h_j \neq h'_j$. The output of $\mathcal{F}_\mathcal{B}$ is in this case $(1, \mathsf{out}, \mathsf{out}')$. We now use $\mathsf{out}, \mathsf{out}'$ to construct a solution to M-SIS on the matrix $\overline{\mathbf{A}}$.

By definition of the forking, we have that the random coins are the same up to the forking index j. As a result, $(f, i^+) = (f', i^+)$ and $e_{i^+} = h_j \neq h'_j = e'_{i^+}$. Because $f = f'$, this implies that $\mathsf{pk}_i = \mathsf{pk}'_i$, $\mathbf{v}_{2,i} = \mathbf{v}'_{2,i}$ and $\mathbf{m}_i = \mathbf{m}'_i$ for all $i \in [N]$. Additionally, due to the fact that e_{i^+} is set before all the e_i in the

queries to $\mathcal{H}_e$, we have that $e_i = e'_i$ for all $i \neq i^+$. Then, since $\mathsf{sig}_{\mathsf{agg}}$ and $\mathsf{sig}'_{\mathsf{agg}}$ both verify, by definition of the reconstructed vectors $\mathbf{v}_{1,1}, \mathbf{v}'_{1,1}$ in Algorithm C.7 and $\mathbf{v}_1, \mathbf{v}'_1$, we have

$$\mathbf{A}\mathbf{v}_1 + \sum_{i\in[N]} e_i(\mathbf{G} - \mathbf{B}_i)\mathbf{v}_{2,i} = \sum_{i\in[N]} e_i\mathcal{H}(\mathbf{m}_i) \bmod qR$$
$$\mathbf{A}\mathbf{v}'_1 + \sum_{i\in[N]} e'_i(\mathbf{G} - \mathbf{B}'_i)\mathbf{v}'_{2,i} = \sum_{i\in[N]} e'_i\mathcal{H}(\mathbf{m}'_i) \bmod qR,$$

such that $\|\mathbf{v}_1\|_2, \|\mathbf{v}'_1\|_2 \leq ws\sqrt{N \cdot 2nd}$. We call $\Delta e = e_{i^+} - e'_{i^+}$. With the prior observations, combining the above equations gives

$$\mathbf{A}(\mathbf{v}_1 - \mathbf{v}'_1) + \Delta e \cdot (\mathbf{G} - \mathbf{B})\mathbf{v}_{2,i^+} = \Delta e \cdot \mathcal{H}(\mathbf{m}_{i^+}) \bmod qR$$

We note that $\mathbf{m}_{i^+}$ was not queried for a signature, but it must have been queried to $\mathcal{H}$ (otherwise $\mathcal{A}$ would have had a negligible advantage to begin with). Hence, $\mathcal{T}_s$ contains an entry $(\mathbf{m}_{i^+}, \mathbf{v}'')$ where $\mathbf{v}''$ was generated as in game G_2. Then, $\mathbf{v}''$ verifies $\mathbf{A}\mathbf{v}''_1 + (\mathbf{G} - \mathbf{B})\mathbf{v}''_2 = \mathcal{H}(\mathbf{m}_{i^+}) \bmod qR$. We then obtain

$$\mathbf{A}(\mathbf{v}_1 - \mathbf{v}'_1 - \Delta e \cdot \mathbf{v}''_1) + \Delta e \cdot (\mathbf{G} - \mathbf{B})(\mathbf{v}_{2,i^+} - \mathbf{v}''_2) = \mathbf{0} \bmod qR,$$

which can be written $\overline{\mathbf{A}}\mathbf{x} = \mathbf{0} \bmod qR$ for

$$\mathbf{x} = \begin{bmatrix} \mathbf{v}_1 - \mathbf{v}'_1 \\ \Delta e \cdot \mathbf{v}_{2,i^+} \end{bmatrix} - \Delta e \cdot \mathbf{v}'' \in R^{d(2+k)}.$$

The adversary $\mathcal{A}$ does not know $\mathbf{v}''$ but only $\overline{\mathbf{A}}\mathbf{v}'' \bmod qR$ which takes $2^{nd\log_2 q}$ possible values. By [DORS08, Lem. 2.2], the entropy of $\mathbf{v}''$ given $\overline{\mathbf{A}}\mathbf{v}'' \bmod qR$ is at least $H_\infty(\mathbf{v}'') - nd\log_2 q$. Since $\mathbf{v}''$ is sampled by the simulator, it holds that $\mathbf{v}''_1 \sim \mathcal{D}_{R^{2d},s}$ and $\mathbf{v}''_2 \sim \mathbf{G}^{-1}(U(R_q^d))$. As a result, $H_\infty(\mathbf{v}'') = H_\infty(\mathcal{D}_{R^{2d},s}) + nd\log_2 q$. Then, by Lemma C.1, we have that $H_\infty(\mathcal{D}_{R^{2d},s}) \geq 2nd\log_2 s - 1$ as $s \geq \eta_\delta(\mathbb{Z}^{2nd})$ for some negligible $\delta > 0$. We thus obtain that the entropy of $\mathbf{v}''$ given $\overline{\mathbf{A}}\mathbf{v}'' \bmod qR$ is at least $2nd\log_2 s - 1 \gg 4\lambda$, and then that $\mathbf{x} = \mathbf{0}$ only with negligible probability. Finally, we have

$$\begin{aligned}\|\mathbf{x}\|_2 &\leq \sqrt{(\|\mathbf{v}_1\|_2 + \|\mathbf{v}'_1\|_2 + \|\Delta e\|_1\|\mathbf{v}''_1\|_2)^2 + (\|\Delta e\|_1 \cdot (\left\|\mathbf{v}_{2,i^+}\right\|_2 + \|\mathbf{v}''_2\|_2))^2} \\ &\leq \sqrt{(2w \cdot (\sqrt{N}+1) \cdot c's\sqrt{2nd})^2 + (2w \cdot 2(b-1)\sqrt{ndk})^2} \\ &= \beta,\end{aligned}$$

except with probability $2^{-4\lambda}$ that is due to Lemma 2.1 and the definition of c'. Therefore, $\mathbf{x}$ is a solution to M-SIS$_{n,d,d(2+k),q,\beta}$ except with negligible probability. Since we assumed that the hardness bound of the latter was $\varepsilon_{\text{M-SIS}}$, it thus hold that

$$\mathsf{frk} \leq \varepsilon_{\text{M-SIS}} + \mathsf{negl}(4\lambda) \tag{8}$$

Combining Equation (8) with the result from the general forking lemma, we get

$$p_2 = \mathsf{acc} \leq \frac{Q_e}{|\mathcal{C}|} + \sqrt{Q_e(\varepsilon_{\text{M-SIS}} + \mathsf{negl}(4\lambda))}.$$

We can assume without loss of generality that $Q_e \leq 2^\lambda$, and recalling that $\mathcal{C}$ is chosen so that $|\mathcal{C}| \geq 2^{2\lambda}$, it holds $Q_e/|\mathcal{C}| = \mathsf{negl}(\lambda)$. Combined with Equations (5), (6), and (7), we get

$$\text{Adv}[\mathcal{A}] \leq \frac{N}{(1-1/Q)^Q} \cdot \left(\varepsilon_{\text{M-LWE}} + \frac{Q_e}{|\mathcal{C}|} + \sqrt{Q_e \varepsilon_{\text{M-SIS}}}\right) + \mathsf{negl}(\lambda),$$

as claimed. When $\varepsilon_{\text{M-LWE}}$ and $\varepsilon_{\text{M-SIS}}$ are negligible in λ and 3λ respectively, the bound is indeed negligible as $\frac{N}{(1-1/Q)^Q} \leq 4N = \mathsf{poly}(\lambda)$ whenever $Q \geq 2$. □

C.5 Performance Evaluation

We now evaluate the performance of our aggregate signature compared to the naive concatenation. For that we define the compression rate as

$$\mathsf{compression\ rate} = \max\left(0, 100 \cdot \left(1 - \frac{|\mathsf{sig_{agg}}|}{|\mathsf{concatenation}|}\right)\right) \%.$$

However, to obtain a fair comparison, we cannot simply compare the concatenation of signatures produced by Algorithm C.4 with the aggregate signature output by Algorithm C.6. Indeed, in the case of a mere concatenation, the parameters used in Algorithm C.4 would not be optimal, one would instead use those for single GPV signatures, as described in Section 4. We thus compare below the size of an aggregate signature with the concatenation of signatures generated with better parameters, tailored to the single signature use-case. Concretely, although we use the same ring $R = \mathbb{Z}[x]/\langle x^n + 1\rangle$, where $n = 256$, we select the optimal parameters from Table 4.3, that is $q \approx 2^{22.5}$, $d = 6$, $b = 2$, $s \approx 362140$ for single signatures, leading to signature size of $|\mathsf{sig_{LW^*}}| = 64461$ bits ≈ 7.87 KB. Hence, the concatenation of N signatures results in a naive aggregate signature of $|\mathsf{concatenation}| = N \cdot |\mathsf{sig_{LW^*}}| = N \cdot 64461$ bits.

We estimate the aggregate signature size for different values of N ranging from $N = 10$ to $N = 2000$. The bit-size of the aggregate signature is given by

$$|\mathsf{sig_{agg}}| = nd(1/2 + \log_2(ws\sqrt{N})) + N \cdot nd\lceil \log_2 q \rceil.$$

The parameters of our scheme are set according to Setup (Algorithm C.2) with $Q = 2^{40}$, where q, d and b are selected to guarantee sufficient security for the underlying M-SIS$_{n,d,d(2+k),q,\beta}$ and M-LWE$^{dk}_{n,d,d,q,U(S_1)}$ problems aiming for $\lambda = 128$ using the methodology from Section A, while minimizing the aggregate signature size. Since the parameters increase with N (typically the bound β), the values of q and d will naturally depend on N accordingly. Hence, when N increases the modulus q and rank d need to be increased to preserve the security of the

scheme, which results in lower compression rates. The higher N gets, the more we would have to increase q and d, and we thus expect that passed a certain threshold N the concatenation would become better than our aggregate signature. Nevertheless, in practical use cases of aggregate signatures the number of signers stays in the low hundreds which in our case offer up to 10% compression rate compared to the naive concatenation, as shown in Table C.1.

N	5	10	50	100	500	1000	2000
$N \cdot \lvert\mathsf{sig}_{\mathsf{LW}^*}\rvert$	39.3	78.7	393.4	786.9	3934.4	7868.8	15737.6
(d, q)	$(6, 2^{22.5})$	$(6, 2^{22.5})$	$(6, 2^{22.5})$	$(6, 2^{22.5})$	$(6, 2^{22.5})$	$(6, 2^{22.5})$	$(6, 2^{22.5})$
$\lvert\mathsf{sig}_{\mathsf{agg}}\rvert$	38.5	71.4	345.1	706.1	3615.8	7444.0	15319.1
(d, q)	$(7, 2^{29.7})$	$(7, 2^{30})$	$(7, 2^{30.99})$	$(7, 2^{31.6})$	$(7, 2^{32.7})$	$(7, 2^{33.6})$	$(7, 2^{34.99})$
Comp. Rate	2.26%	9.29%	12.29%	10.26%	8.10%	5.39%	2.66%

Table C.1. Comparison estimates of our aggregate signature and the concatenation of LW signatures over module lattices. Sizes of $N \cdot \lvert\mathsf{sig}_{\mathsf{LW}^*}\rvert$ and $\lvert\mathsf{sig}_{\mathsf{agg}}\rvert$ are expressed in KB. All the parameters are chosen for an M-SIS and M-LWE security of at least $\lambda = 128$ bits in the Core-SVP model.