

归藏-玄色 (ARCANE-Octarine)

算法规范及支撑文档

本提交材料来自归藏 (ARCANE) 团队，成员按字母顺序列出如下：

- Jose Maria Bermudo Mera, 比利时 PQShield 公司
- Joppe W. Bos, 比利时恩智浦半导体 (NXP Semiconductors)
- Angshuman Karmakar, 印度坎普尔理工学院 (Indian Institute of Technology Kanpur)
- Joost Renes, 美国恩智浦半导体 (NXP Semiconductors)
- Sujoy Sinha Roy, 奥地利格拉茨工业大学 (Graz University of Technology)
- Michiel Van Beirendonck, 比利时贝尔福特实验室 (Belfort Labs)
- Frederik Vercauteren, 比利时鲁汶大学 (KU Leuven)
- 王鹏 (Peng Wang), 中国科学院大学密码学院
- 王跃武 (Yuewu Wang), 中国科学院大学密码学院
- 郑昉昱 (Fangyu Zheng), 中国科学院大学密码学院

联系人： 王鹏 (Peng Wang)

电子邮箱： p-wang@ucas.ac.cn

网站： <https://soc.ucas.ac.cn/index.php/zh/ARCANE>
<https://www.pq-arcane.org>

Contents

1	归藏 (ARCANE) 算法套件	1
2	引言	4
2.1	基本方法概述	4
2.2	与现有工作的关系	5
2.3	安全性	6
3	预备知识	7
3.1	符号约定	7
3.2	格密码学 (Lattice-based Cryptography)	8
3.2.1	误差学习 (Learning With Errors)	8
3.2.2	舍入学习 (Learning With Rounding)	9
3.2.3	舍入学习的安全性	9
3.3	作为结构化模的环	10
3.4	根式环舍入学习 (Radical Ring-LWR)	12
4	算法描述	12
5	设计原理	22
5.1	参数选择准则	22
5.2	正确性与拒绝概率分析	22
5.3	其他考虑	23
6	安全性声明与分析	24
6.1	理论安全性	24
6.1.1	RR-LWR 困难问题	24
6.1.2	玄色 (Octarine) 的不可伪造性	26
6.1.3	超越不可伪造性的性质	30
6.2	实践安全性	31
6.2.1	物理安全性	31
7	性能评估	32
7.1	算术实现	32
7.2	性能分析	33
7.3	性能测试	34
A	SageMath 脚本	37
A.1	安全性估计	37
B	检查清单	41

1 归藏 (ARCANE) 算法套件

后量子密码学 (post-quantum cryptography) 要求算法能够抵抗量子敌手, 并可在密钥交换 (或封装)、数字签名等应用中取代 RSA、椭圆曲线密码学 (Elliptic Curve Cryptography, ECC) 等传统方案. 归藏 (ARCANE) 算法套件包含两个算法:

- 归藏-软猬 (ARCANE-Mithril) : 一种后量子密钥封装机制 (Key Encapsulation Mechanism, KEM);
- 归藏-玄色 (ARCANE-Octarine) : 一种后量子数字签名 (Digital Signature, DS) 方案.

下面重点介绍 软猬 (Mithril) 与 玄色 (Octarine) 的共有特性.

创新性. 其安全性基于根式环舍入学习 (Radical Ring Learning-With-Rounding, RR-LWR) 问题 [BRVW26], 该问题是根式数域上的环舍入学习 (Ring-LWR) 变体. 在这一变体中, 舍入是在迁移后的单项式基中逐系数进行的; 该基即根式环单项式基在定理 3 所形式化的数域同构下的像. 它可作为模舍入学习 (Module Learning-With-Rounding, Module-LWR) 的替代方案, 在保留 Ring-LWR 大部分性能优势的同时, 继承 Module-LWR 参数选择灵活的优点. KEM 和 DS 方案均在如下形式的根式环上运算:

$$\mathbf{S}_{q,n,k} = (\mathbb{Z}_q[y]/(y^n + 1))[x]/(x^k - y - 2), \quad (1)$$

其中, q 与 n 均为二的幂, k 为任意正整数. 它与数域表示 $\mathbf{S}_{q,n,k} \cong \mathbb{Z}_q[x]/((x^k - 2)^n + 1)$ 之间存在直接的同构 [BRVW26, 第 4 节], 由此得到本方案迁移基 Ring-LWR 变体所采用的根式环表示. 环 $\mathbf{S}_{q,n,k}$ 是第 $2n$ 个分圆域的 k 次根式扩张. 因此, 可按需采用数论变换 (Number Theoretic Transform, NTT) 等标准实现技术或其他多项式乘法例程; 与模结构相比, 根式扩张还能更高效地生成矩阵.

我们选取二的幂作为 q , 而非素数. 这使 软猬 (Mithril) 在设计上与 Saber [DKR⁺20, 第 A.2 节] 相近, 但二者选用的理想格不同 (本方案采用 RR-LWR, 而非 Module-LWR). 我们还定义了数字签名方案 玄色 (Octarine); 该方案遵循成熟的带中止 Fiat-Shamir (Fiat-Shamir with Aborts) 范式 [Lyu09, Lyu12], 并将其适配到 RR-LWR 场景. Okada 等人此前提出过基于 LWR 的签名方案 MLWRSign [OTF⁺20]. 我们的做法有所不同: 将 $\mathbf{s}_2$ 定义为由公钥确定的确定性误差 $\mathbf{s}_2 = \mathbf{A}\mathbf{s}_1 - \frac{q}{p} \lfloor \frac{p}{q} \mathbf{A}\mathbf{s}_1 \rfloor$, 而不是 MLWRSign 中的有理数值而 MLWRSign 使用 $\lfloor \frac{p}{q} \mathbf{A}\mathbf{s}_1 \rfloor - \frac{p}{q} \mathbf{A}\mathbf{s}_1$. 此外, 我们在密钥生成阶段而非签名生成阶段计算 $\mathbf{s}_2$. 这简化了设计, 使整体构造较 MLWRSign 更接近 CRYSTALS-Dilithium [DKL⁺18].

简洁性. 我们采用根式数域上的 Ring-LWR 变体, 并在上述迁移后的单项式基中逐系数舍入; 具体估计计入了相对于通常典范嵌入视角所产生的基失真. 所得 KEM 和数字签名方案以较为保守的安全概念为目标: KEM 达到自适应选择密文攻击下的不可区分性 (IND-CCA2) [BDPR98], 数字签名则在随机预言机模型 (Random Oracle Model, ROM) 中达到适应性选择消息攻击下的强不可伪造性 (SUF-CMA) [ADR02].

我们从多个方面简化现有设计. 首先, 采用 (根式) 环结构而非模结构. 其次, 通过使用二的幂模数 q , 尽量降低各种参数选择的复杂性. 所有采样均使用模二的幂 α 的区间 $[-\alpha, \alpha - 1]$ 上的均匀分布, 并尽可能在各参数集中统一 α 的取值. 只需生成 $\log_2(2\alpha)$ 个随机比特, 即可产生 $[-\alpha, \alpha - 1]$ 中的随机元素. 因此, 在对秘密值、公有矩阵及掩蔽向量的系数采样时无需拒绝采样, 也无需丢弃任何 (伪) 随机生成的比特. 我们选择的参数使失败概率始终可忽略不计. 尤

其对于数字签名方案，分解参数和 Fiat-Shamir 中止拒绝循环的参数均基于二的幂，因此只需检查是否发生进位即可轻松完成边界检查。我们还减少了签名生成中的拒绝次数，以降低签名耗时的方差，从而可能更便于应用协议采用。例如，玄色 (Octarine) -128 完成一次签名平均需要 2.13 轮，标准差为 1.55；相比之下，Dilithium2 平均需要 4.25 轮，标准差为 3.72。

灵活性. 本方案继承了格密码系统的若干优点。选择 13 比特或 24 比特的二的幂作为 q ，可使模 q 的值容纳在大多数现代 CPU 字中，从而易于实现。我们作出以下设计选择，以便在不同体系结构上实现安全而灵活的实现：

- 尽管素数模数 q 的 LWR 可借助渐近高效的 NTT 获得快速实现优势，已有研究表明，二的幂 q 更适合掩蔽 [CJRR99, PR13]。为防御侧信道攻击和故障攻击，通常需要采用掩蔽，这对嵌入式系统及高保障实现尤为重要。构造掩蔽实现时常用的算术到布尔转换（及其逆转换）[CGV14, BBE⁺18] 在模数为二的幂时复杂度更低；无需拒绝采样的采样例程亦是如此。嵌入式系统本就受到资源约束，简化侧信道与故障防护对策有助于降低实现成本。
- 使用根式环可缩小公有矩阵，直接降低存储需求。例如，与 ML-DSA-87 的矩阵相比，本方案矩阵的大小几乎缩小到七分之一：若每个系数使用 32 比特，ML-DSA-87 的矩阵需要超过 57.3 kB；采用紧凑编码存储时仍需超过 41.2 kB。相比之下，本方案公有矩阵在每系数 32 比特时约需 8.2 kB，紧凑编码后仅需 6.1 kB。这不仅降低了矩阵生成成本，也减少了矩阵在多次使用期间（尤其是生成签名时）的存储开销，从而为受限系统提供更好的时间-存储权衡。
- 我们在标准域中表示所有变量，以避免算法描述依赖特定实现或算法技术。以特定形式（如 NTT 域）表示元素虽可能带来性能优势，却可能限制算法面向不同平台时的可移植性。归藏算法套件的算术可使用 NTT [CT65, Pol71]、Toom-Cook [Too63, Coo66]、Karatsuba [KO62] 高效实现，也可利用现有硬件执行 Kronecker 代换 [Kro82, BRvV22]。我们的结果表明，其性能相对于先进格密码系统仍具竞争力。特别地，将软猬 (Mithril) 私钥多项式 s 表示在普通域而非 NTT 域中时，每个系数只需 $\log_2(2\eta)$ 比特；Kyber 则需 $\log_2(q)$ 比特，因而本方案可显著缩小私钥。

兼容性. 该 RR-LWR 族的安全级别主要由格维数决定，而格维数取决于多项式次数 n 和根式扩张次数 k 。增大取二的幂的 n 可提高算术运算的总体效率；改变 k 则可像 Module-LWR 一样，更细粒度地控制安全级别。虽然 $n = 256$ 是 KEM 的常见选择，但此时格维数只能选为 512 或 768，因而难以恰当地达到 128 比特安全级别：维数 512 可能安全性不足（或安全余量极小），而维数 768 通常又会提供过高的安全级别。因此，我们对 KEM 采取保守设计，选择 $n = 128$ 、 $k = 5$ ，使用维数为 640 的格。具体而言，在精化攻击模型下，这将比特安全性从 Kyber 和 Saber 的 140 比特提高到 153 比特。基于根式扩张的理想格可由大小随 k 线性增长的元素表示，而 Module-LWE/LWR 则随 k 二次增长，因而我们减轻了矩阵生成成本所受的部分影响。与使用较小格相比，这一设计策略会略微增大密钥和密文，并影响总体性能；但为提高对所达到的 128 比特安全级别的信心，我们认为这是可以接受的。

我们在所有 KEM 参数集中统一使用 $n = 128$ ，以简化软硬件实现。对数字签名，我们在所有参数集中选择 $n = 1024$ ，并令 $k = 1, 2, 5$ ，以最大限度发挥分圆子环结构的性能。此外还有许多可调参数，会影响性能、密钥大小、拒绝率及签名耗时方差等。我们的参数选择使公钥和签名大小与其他格方案相近，同时提高若干常用参数集的性能并降低签名生成方差。例如，与 CRYSTALS-Dilithium-2 [DKL⁺18] 相比，归藏-玄色 (ARCANE-Octarine) -128 生成签名的期望重复次数约减半（分别为 2.1 次与 4.3 次）。

为最大限度复用数字签名与 KEM 的实现，我们将所有采样分布选为模二的幂 α 的区间 $[-\alpha, \alpha - 1]$ 上的均匀分布。数字签名和 KEM 均默认选择 $\alpha = 2$ ，仅有一个参数集例外，因为继续采用该值会造成过大的安全性影响。基于二的幂的打包和舍入操作效率高，并能最优利用所需比特。两种算法都选取适合 NTT 的次数 n （即二的幂），从而可在 NTT 对目标平台最优时采用 NTT 算术。尽管 KEM 与数字签名的 n 不同，仍可将 $n = 128$ 的乘法用作 1024 次乘法的子例程，得到共享实现。例如，KEM 取 $q = 2^{13}$ ，数字签名取 $q = 2^{24}$ 时，可选择适合 NTT 的 32 比特素数 p_1 。该素数既可用于 KEM，也可借助环 $\mathbb{Z}_{p_1 p_2} \cong \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$ 用于数字签名。实现时，可用单个模块在任意素数 p 对应的 $\mathbb{Z}_p$ 上运算，复用绝大部分代码；其上只需增加一个小型双素数中国剩余定理（Chinese Remainder Theorem, CRT）计算层，即可从 $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$ 转换回 $\mathbb{Z}_p$ 。参考实现和优化实现展示了这种方法及其效率。

应注意，素数的选取以及是否使用 NTT 均不属于本规范，而纯属实现选择；若其他实现方法性能更优，也可采用。

可扩展性。 这些构造不仅用于本文规定的具体实例，还旨在充当可复用的构件。在外围协议提供所需的认证、密钥确认和组合安全保证的前提下，KEM 可集成到更高层协议中，用于临时密钥建立、KEM-数据封装机制（Data Encapsulation Mechanism, DEM）加密以及认证密钥建立机制。它还可按混合模式与传统或其他后量子密钥建立机制结合，以支持迁移期间的算法敏捷性。

在套件层面，方案的模块化组织方式也为未来变体留出了空间。具体而言，可在不改变底层根式环运算的情况下考虑其他参数配置、哈希/可扩展输出函数（Extendable-Output Function, XOF）/密钥派生函数（Key Derivation Function, KDF）实例或实现配置。不过，此类变体仍应附有更新后的正确性界、具体安全性估计和实现验证。

玄色（Octarine）支持通用消息签名、代码签名、数字证书和安全证明等标准签名应用。此外，由于该方案遵循 Fiat-Shamir 范式，相同的代数环境还可作为更高级 Fiat-Shamir 类签名构造（包括门限签名 [PN25]）的起点。

因此，RR-LWR 框架不仅应被视为一组固定的 KEM 和签名实例，还应被视为一个平台，用于未来的协议集成以及复用相同根式环结构、经过审慎分析的变体。

不同平台上的性能优势。 归藏（ARCANE）套件采用保守的设计选择，因而可针对广泛的目标平台优化实现。特别地，本规范有意将多项式算术方法的选择与算法本身解耦：实现可按需采用基于 NTT、Toom-Cook、Karatsuba、朴素乘法或体系结构专用的技术。

为与成熟格方案进行有意义的比较，性能章节除报告基于指定 `pseudoXOF` 实例的提交测量结果外，还在适用时报告基于 SHAKE 的基准配置。在 KEM 场景下，基于 SHAKE 的 AVX2 比较涵盖 软猬（Mithril）-128 和 软猬（Mithril）-256；软猬（Mithril）-128 的 AVX2 优化实现比 ML-KEM-512 慢约 1.4 倍。造成这一差异的部分原因是，软猬（Mithril）-128 的估计安全余量高于 ML-KEM-512。在更高安全级别下，软猬（Mithril）-256 与 ML-KEM-1024 的性能差距缩小到约 1.1 倍，同时 软猬（Mithril）仍提供更大的估计安全余量。

在数字签名场景下，玄色（Octarine）-128 的密钥生成和签名速度快于 ML-DSA-44，而 ML-DSA-44 的验证速度仍更快。在更高安全级别下，玄色（Octarine）-256 在所测签名与验证操作上均优于 ML-DSA-87，签名和验证加速比分别为 1.13 和 1.23。这些结果表明，归藏（ARCANE）的设计在保留保守安全余量的同时实现了有竞争力的性能，并为平台专用优化留下了充分的灵活性。

应注意，归藏（ARCANE）方案在降低存储消耗及减少公钥矩阵伪随机数生成等方面的优

势，对于 XOF 成本占主导的较小型 Arm 或 RISC-V 平台内核会愈发显著。此外，简洁的设计选择应能显著便利侧信道与故障防护实现；Saber 相对 Kyber 的优势即是一个例证。同时，它也为可基于各类乘法器的硬件设计保留了灵活性。我们计划在本规范的后续版本中量化这些改进。

2 引言

本文以下部分重点给出数字签名方案 归藏-玄色（ARCANE-Octarine）（简称 玄色（Octarine））的规范。

2.1 基本方法概述

带中止 Fiat-Shamir 范式最早由文献 [Lyu09] 提出，是若干实用格签名方案的基础。下面从较高层面说明该方法在本方案中的工作方式，详细算法规范见算法 3。

密钥生成. 签名者从种子 ρ 采样公有矩阵 $\mathbf{A}$ ，并抽取系数有界的秘密向量 $\mathbf{s}_1$ 。LWR 舍入值 $\mathbf{b} = \lfloor \frac{p}{q} \mathbf{A} \mathbf{s}_1 \rfloor$ 构成公钥的主要部分；舍入误差 $\mathbf{s}_2 = \mathbf{A} \mathbf{s}_1 - \frac{q}{p} \mathbf{b}$ 存入私钥。直观而言， $\mathbf{s}_2$ 记录 LWR 舍入引入的偏差；签名时，它允许签名者在形成验证者重构的承诺时补偿该偏差，同时避免有理数算术。该定义的动机见第 5 节。

Dilithium [DKL⁺18]（后成为 ML-DSA 标准）引入了一种使用 Power2Round 函数并在签名中包含提示向量的公钥压缩方法。我们也采用该技术，将签名再缩小 d 比特；不同参数集取 $d \in \{11, 13, 14\}$ 。

签名. 签名者抽取系数均匀随机且以 γ_1 为界的掩蔽向量 $\mathbf{y}$ ，计算承诺 $\mathbf{w} = \mathbf{A} \mathbf{y}$ ，并提取高位 $\mathbf{w}_1 = \lfloor \mathbf{w} / (2^{\gamma_2}) \rfloor$ ；由于 γ_2 为二的幂，该除法等价于简单右移。将消息摘要 μ 与 $\mathbf{w}_1$ 一同哈希，导出挑战种子 $\tilde{c}$ ，计算挑战多项式 $c = \text{SampleInBall}(\tilde{c})$ ，并形成候选响应 $\mathbf{z} = \mathbf{y} + c \mathbf{s}_1$ 。为防止 $\mathbf{z}$ 泄漏私钥信息，签名者使用拒绝采样。第一次拒绝测试限制 $\|\mathbf{z}\|_\infty$ ，确保接受响应的分布与 $\mathbf{s}_1$ 足够独立；第二次限制 $\mathbf{w} + c \mathbf{s}_2$ 的低位部分，确保验证者能借助公钥和提示比特恢复相同高位承诺。参数使期望重复次数较小：玄色（Octarine）-128、玄色（Octarine）-256、玄色（Octarine）-512 分别为 2.1、2.3、3.6 次。

验证. 验证根据响应 $\mathbf{z}$ 、挑战 c 和压缩公钥重新计算原始承诺的近似值

$$\mathbf{w}'_{\text{Approx}} = \mathbf{A} \mathbf{z} - c \cdot \frac{q}{p} \mathbf{b}_1 \cdot 2^d,$$

其中 $\mathbf{b}_1$ 是公钥高位部分， d 是 Power2Round 压缩丢弃的比特数， c 从挑战哈希 $\tilde{c}$ 恢复。该值与签名者承诺的差异仅来自公钥压缩和 LWR 舍入产生的受控舍入项。提示向量 $\mathbf{h}$ 用于校正缺失低位 $\mathbf{b}_0$ 引起的进位，得到 $\mathbf{w}'_1$ 。若 $\mathbf{w}'_1 = \mathbf{w}_1$ ，即 $\tilde{c} = H(\mu \| \mathbf{w}'_1)$ ，且 $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ ，则接受签名。

参数以三个安全级别为目标，并针对底层 RR-LWR 与 SIS 型困难性假设目前已知最佳经典和量子攻击保守选取。方案具有模块化特征：调整环维数可在不改变算法整体结构的情况下调节大部分安全级别；但仍需谨慎选择其他参数（主要为 p 、 η 、 γ_1 、 γ_2 、 τ ），以平衡性能特征与确切安全级别。

2.2 与现有工作的关系

下面将 玄色 (Octarine) 与最相关的现有格数字签名方案比较。现有提案中, 最接近的参照是 CRYSTALS-Dilithium [DKL⁺18] (标准化为 ML-DSA) 和基于 LWR 的签名方案 MLWR-Sign [OTF⁺20]。其他重要数字签名方案还包括 Falcon [PFH⁺22] 和 HAETAE [CCD⁺24]。由于这些方案与本方案差异更大, 比较时主要关注性能和大小。

本构造遵循成熟的带中止 Fiat-Shamir 范式, 但结合了不同的代数环境和舍入误差处理方式。所得设计旨在改善概念简洁性、实现效率、存储用量与加固实现稳健性之间的总体平衡。目标不仅是获得有竞争力的渐近安全性和性能, 还要降低在受限及侧信道敏感环境中部署方案的工程复杂度。

CRYSTALS-Dilithium / ML-DSA. 玄色 (Octarine) 与 ML-DSA 具有相同的高层签名结构, 但在若干重要设计选择上有所不同。

第一, 本方案在根式环 $S_{q,n,k}$ 上实例化, 安全性基于 RR-LWR 假设而非 Module-LWE, 因而具有不同的实现特征。特别地, 选择二的幂作为 q 。这并不排斥包括 NTT 实现在内的快速乘法技术, 却避免设计依赖特定 NTT 友好素数模数。二的幂算术使算法结构更简洁、统一, 同时仍可依目标平台采用 NTT、Toom-Cook、Karatsuba 或其他乘法技术。从规范角度看, 这消除了大量复杂性, 降低了构造对特定实现策略的依赖。虽然会影响性能, 第 7 节表明: 玄色 (Octarine) -128 的密钥生成和签名快于 ML-DSA-44, 但验证较慢; 玄色 (Octarine) -256 在所有测量操作上都快于 ML-DSA-87。

第二, 所有目标安全级别的参数选择均由保守安全目标驱动。方案包含 512 比特目标安全级别, 并相应调整哈希域和输出长度。为在该级别保持统一, 将相关哈希输出 (如摘要 tr 、 μ 及每签名种子 ρ'') 扩为 1024 比特, 而挑战哈希 $\tilde{c}$ 保留依赖安全级别的长度 $\lambda/4$; 较低安全级别原可使用更短输出。这样避免规范和实现中特定于级别的特殊情况, 使各参数集的整体结构不变。更一般地, 128 与 256 比特参数集旨在针对相关 SIS 和 RR-LWR 攻击保留保守余量, 而非仅优化最小尺寸; 该理念面向长期评估和部署。

第三, 与 ML-DSA 相比, 简化了若干实践中非常重要的底层组件。秘密系数从 $[-2, 1]$ 或 $[-4, 3]$ 采样, 只需每次生成 2 或 3 比特, 无需拒绝。取 $q = 2^{24}$ 后, 公有矩阵系数也可由伪随机输出逐比特精确生成, 无需丢弃随机比特。这个小改动带来直接的常数时间采样、更好的跨实现可复现性, 并减少需加固的控制流变化; 非二的幂区间即便分布简单, 通常也需要额外拒绝逻辑。

第四, 有意将 γ_1 和 γ_2 选为二的幂, 以简单舍入和比特移位替代若干分解相关操作。因此, 对应 ML-DSA 的 HighBits、LowBits 和 Decompose 功能可通过更规则的比特级接口表达。这不仅简化软件, 也提高实现保障: 由移位、掩码和加法表达的算术通常比依赖模分解规则的算术更易验证、优化和加固。

第五, 将环次数提高到 $n = 1024$, 有两项优势: 一方面增加可供实现利用的环结构, 提高底层算术效率; 另一方面直接改善挑战生成。SampleInBall 在 n 次多项式中放置 τ 个非零系数, 较大的 n 可在保持所需挑战熵的同时减小 τ , 进而减小 $\beta = \tau\eta$, 扩大响应允许区间并提高 Fiat-Shamir 中止循环成功率。因此, 较大环维数不仅用于算术效率, 也提高签名效率并降低签名耗时方差。例如在 256 比特安全级别, 期望重复次数从约 3.85 降至约 2.3。较低拒绝率可降低平均签名成本、缩小计时变化, 并便于集成到延迟约束更严格的系统。

MLWRSign. Okada 等人曾提出基于 LWR 的签名方案 MLWRSign [OTF+20]. 主要区别在于舍入误差的处理.

MLWRSign 通过如下有理数值表示舍入偏差:

$$\mathbf{s}_2^{\text{MLWRSign}} = \lfloor \frac{p}{q} \mathbf{A} \mathbf{s}_1 \rfloor - \frac{p}{q} \mathbf{A} \mathbf{s}_1 \in \mathbf{S}_{q/p, n, k},$$

并由此引入辅助误差项

$$\xi_1 = \lfloor \frac{p}{q} \mathbf{A} \mathbf{y} \rfloor - \frac{p}{q} \mathbf{A} \mathbf{y}, \quad \xi_2 = \lfloor c \mathbf{s}_2 \rfloor - c \mathbf{s}_2, \quad \nu = \lfloor \xi_2 - \xi_1 \rfloor.$$

来表达签名和验证方程. 相比之下, 本方案定义

$$\mathbf{s}_2 = \mathbf{A} \mathbf{s}_1 - \frac{q}{p} \mathbf{b} \in \mathbf{S}_{q/p, n, k},$$

其中 $\mathbf{b} = \lfloor \frac{p}{q} \mathbf{A} \mathbf{s}_1 \rfloor$. 两个定义满足 $\mathbf{s}_2 = -\frac{q}{p} \mathbf{s}_2^{\text{MLWRSign}}$, 故 $\mathbf{s}_2$ 是 $\mathbf{S}_{q/p, n, k}$ 中的整数元素.

除已知缩放 $\mathbf{s}_2 = -\frac{q}{p} \mathbf{s}_2^{\text{MLWRSign}}$ 外, 两个定义包含相同信息, 但算法后果明显不同. 本表述完全避免有理数算术, 不再需要额外记账变量 ξ_1 、 ξ_2 、 ν , 使方案更接近标准带中止 Fiat-Shamir 模板.

第二项优势是舍入误差的分布作用更透明. $\mathbf{s}_2$ 的系数始终在整数区间 $[-q/(2p), q/(2p) - 1]$ 内. 启发式地, 若将 $\mathbf{A} \mathbf{s}_1$ 视为模 q 均匀, 则导出误差在该区间上接近均匀; 这一更强的均匀乘积启发式不同于对舍入值 $\mathbf{b}$ 的 RR-LWR 假设. 无论该启发式是否成立, 公钥 $(\mathbf{A}, \mathbf{b})$ 按构造就是具有确定性误差 $\mathbf{s}_2$ 的 RR-LWR 样本, 明确体现了算法与底层困难性假设的联系.

最后, MLWRSign 每次签名都重新计算舍入误差. 本方案因 $\mathbf{s}_2$ 完全由 $\mathbf{A}$ 和 $\mathbf{s}_1$ 决定, 只在密钥生成时计算一次并存入私钥. 这将少量工作和存储转移到很少执行的密钥生成, 避免在线签名期间重复计算. 存储 $\mathbf{s}_2$ 可降低每签名成本、简化签名代码路径并使实现动态行为更规则, 符合优先在线效率和实现稳健性、而非略微节省私钥大小的总体目标.

总体而言, 玄色 (Octarine) 将 ML-DSA 清晰的体系结构与比 MLWRSign 更简洁的 LWR 表述结合. 相对 ML-DSA, 主要优势是二的幂模数、更简单的采样和分解、通过降低中止率提高签名效率, 以及显著更小的矩阵表示; 相对 MLWRSign, 主要优势是消除有理数算术和辅助校正项、更直接地将公钥解释为 RR-LWR 样本, 以及在密钥生成时预计算舍入误差以降低签名成本. 这些差异构成了实质上更简洁、更面向实现的设计, 而非对现有技术的小幅改写.

2.3 安全性

在经典随机预言机模型 (ROM) 中, 证明方案达到 SUF-CMA 安全, 并在理想化高最小熵模型中具有 BUFF 所有权性质及相应的弱不可重签名声明. 不可伪造性归约使用 RR-LWR 假设和 SIS 型假设; 在所述哈希函数及编码假设下, BUFF 型性质由经过检查的挑战摘要推出. RR-LWR 是 Module-LWR 的根式环对应问题, 在第 6.1.1 节形式化定义. 第 6 节给出完整安全性分析, 包括 ROM 安全归约及各参数集的具体经典、量子攻击成本估计.

表 1 从高层面对其他格签名方案进行比较, 报告私钥恢复及签名伪造相关短解关系的经典和量子 Core-SVP 成本. 伪造列使用 EUF-CMA 估计; 相应 SUF-CMA 估计可能不同, 因为强不可伪造性可能需要额外的齐次短解关系, 这一差异对 HAETAE 尤为显著.

方案	攻击	安全性			大小 (字节)		
		BKZ (b)	C	Q	sk	pk	sig
玄色 (Octarine) -128	密钥恢复	424	124	112	2 432	1 344	2 564
	不可伪造性	439	128	116			
玄色 (Octarine) -256	密钥恢复	924	270	245	4 608	2 368	5 449
	不可伪造性	910	266	241			
ML-DSA-44	密钥恢复	433	126	115	2 560	1 312	2 420
	不可伪造性	423	124	112			
ML-DSA-87	密钥恢复	883	258	234	4 896	2 592	4 627
	不可伪造性	909	265	241			
Falcon-512	密钥恢复	458	133	121	1 281	897	666
	不可伪造性	411	120	108			
Falcon-1024	密钥恢复	936	273	248	2 305	1 793	1 280
	不可伪造性	952	277	252			
HAETAE-120	密钥恢复	428	125	109	1 376	992	1 463
	不可伪造性	409	119	105			
HAETAE-260	密钥恢复	988	288	253	2 720	2 080	2 908
	不可伪造性	878	256	225			

Table 1: 玄色 (Octarine) 与相关格方案的密钥恢复 (LWR)、存在性不可伪造性 (SIS) 安全估计及字节大小比较.

3 预备知识

本节给出理解本规范所需的全部背景知识.

3.1 符号约定

环元素. 整数环记为 $\mathbf{Z}$; 对于正整数 q , $\mathbf{Z}_q$ 表示商环 $\mathbf{Z}/q\mathbf{Z}$. 给定正整数 m , 第 m 个分圆多项式 (cyclotomic polynomial) 记为 $\Phi_m(x)$, 其定义为以所有 m 次本原单位根为根的唯一首一多项式, 即

$$\Phi_m(x) = \prod_{\substack{k=1 \\ \gcd(k,m)=1}}^m \left(x - e^{2\pi i \frac{k}{m}} \right),$$

其中满足 $\gcd(k, m) = 1$ 的 $e^{2\pi i \frac{k}{m}}$ 为 m 次本原单位根. 可以证明 $\Phi_m(x)$ 的系数均为整数, 并由定义可知 $\deg(\Phi_m(x)) = \varphi(m)$, 其中 $\varphi(m)$ 是 Euler 欧拉函数, 用于统计小于 m 且与 m 互素的正整数个数. 众所周知, 当 $m = 2n$ 且 n 本身为二的幂时, $\Phi_m(x) = x^n + 1$. 多项式商环记为 $\mathbf{R}_q = \mathbf{Z}_q[x]/(f(x))$, 其中 $f(x) \in \mathbf{Z}[x]$ 为首一多项式.

设 n 为二的幂, k 为正整数. 定义分圆环 $\mathbf{R}_{q,n} = \mathbf{Z}_q[y]/(y^n + 1)$ 以及

$$\mathbf{S}_{q,n,k} = \mathbf{R}_{q,n}[x] / (x^k - y - 2)$$

作为 $\mathbf{R}_{q,n}$ 的 k 次根式扩张. $\mathbf{S}_{q,n,k}$ 中的元素表示为多项式 $\mathbf{g} = \sum_{i=0}^{k-1} \mathbf{g}_i x^i$, 其系数 $\mathbf{g}_i \in \mathbf{R}_{q,n}$. 普通小写字母表示 $\mathbf{Z}$ 或 $\mathbf{Z}_q$ 中的元素; 粗体小写字母表示 $\mathbf{R}_{q,n}$ 或 $\mathbf{S}_{q,n,k}$ 中的元素 (即多项式或环元素); 粗体大写字母表示矩阵.

模表示. 对于偶整数 α 和 $r \in \mathbf{Z}$, 记 $r' = r \bmod \alpha$ 为区间 $[-\alpha/2, \alpha/2)$ 中满足 $r' \equiv r \pmod{\alpha}$ 的唯一代表元; 该区间包含 $-\alpha/2$, 不包含 $\alpha/2$. 当 α 为二的幂时, 这等价于提取 r 的最低 $\log_2(\alpha)$ 个有效比特. 除非另有说明, 本文假定所有变量均在有符号区间内表示. 在使用 32 位或 64 位字的现代 CPU 上实现该操作时, 若丢弃高位比特, 须特别注意符号扩展. 该操作以逐系数方式扩展到 $\mathbf{S}_{q,n,k}$ 中的元素.

范数. 对于环元素 $\mathbf{g} = \sum_{i=0}^{k-1} \mathbf{g}_i x^i \in \mathbf{S}_{q,n,k}$ 以及 $\mathbf{g}_i = \sum_{j=0}^{n-1} \mathbf{g}_{i,j} y^j \in \mathbf{R}_{q,n}$, 定义 $\|\mathbf{g}\|_\infty = \max_{i,j} |\mathbf{g}_{i,j} \bmod q|$.

采样分布. 用 $D_{\mathbf{x}}(R)$ 表示从环 R 中采样变量 $\mathbf{x}$ 所用的分布. 若依据种子 σ 进行确定性采样, 则记为 $D_{\mathbf{x}}(R; \sigma)$. 若使用多个种子 $\sigma_1, \dots, \sigma_m$, 则记为 $D_{\mathbf{x}}(R; \sigma_1, \dots, \sigma_m)$, 并在采样前将它们串接为 $\sigma = \sigma_1 \mid \dots \mid \sigma_m$. 例如, 可用这种方式在各个种子后附加环索引.

哈希函数. 算法使用输出长度可变 (可扩展) 的哈希函数 $H(\text{input}, \text{outlen})$, 其中 outlen 以字节为单位. 每次调用 H 时均单独规定其输出长度. 安全性证明要求 H 表现为随机预言机, 并具有 $\lambda/2$ 比特碰撞抗性和 λ 比特原像抗性, 其中 λ 为目标安全级别. H 的具体实例将与最终参数集一并规定.

3.2 格密码学 (Lattice-based Cryptography)

理解舍入学习 (Learning With Rounding, LWR) 的理论与实践需要先介绍误差学习 (Learning With Errors, LWE) 问题 [Reg05] 及其较常用的结构化变体: 环误差学习 (Ring-LWE) [LPR10] 和模误差学习 (Module-LWE) [BGV12, LS15]. 感兴趣的读者可参阅有关 LWE 的综合性综述 [Reg10, Pei16].

3.2.1 误差学习 (Learning With Errors)

设模秩 d 为正整数, χ 表示 $\mathbf{R}$ 上的一个分布. 令 $s \in \mathbf{R}_q^d$ 为秘密向量, 定义分布 $A_{s,\chi}$ 为

$$A_{s,\chi} = (a, b = \langle a, s \rangle + e \bmod q) \in \mathbf{R}_q^d \times \mathbf{R}_q$$

其中, $a \leftarrow \mathcal{U}(\mathbf{R}_q^d)$ 为均匀随机采样值, $e \leftarrow \chi$ 为误差项.

搜索型 LWE 问题 (*search LWE problem*) 要求在获得来自 $A_{s,\chi}$ 的多项式数量的独立样本后恢复秘密 s ; 判定型 LWE 问题 (*decision LWE problem*) 则要求区分来自 $A_{s,\chi}$ 的样本与从 $\mathbf{R}_q^d \times \mathbf{R}_q$ 中均匀抽取的样本. 实践中常取 $\mathbf{R}_{q,n} = \mathbf{Z}_q[x]/(f(x))$, 其中 $f(x)$ 为分圆多项式

$$f(x) = \Phi_{2n}(x) = x^n + 1$$

其中 n 为二的幂. 在此 Ring-LWE 变体中, 所有运算均在环 $\mathbf{R}_q$ 内完成. 单个 Ring-LWE 样本可嵌入相当于 n 个普通 LWE (Plain-LWE) 样本的信息, 同时显著缩小密码密钥. 此外, 分圆多项式的选择允许使用数论变换 (NTT) [Pol71] 高效计算多项式乘法; NTT 是一种仅涉及整数算术的 Fourier 变换变体.

Module-LWE 问题 [BGV12, LS15] 同时推广了 Plain-LWE 和 Ring-LWE, 提供一种灵活的中间结构, 可在安全性与效率之间权衡. 在 Module-LWE 中, 秘密是多项式向量, 样本为 $(\mathbf{a}, b)$, 其中 $b = \langle \mathbf{a}, s \rangle + e$. 因此, Module-LWE 是 Plain-LWE 与 Ring-LWE 的共同推广. 令秩

$d = 1$ 即得到 Ring-LWE 问题；令环为 $\mathbf{Z}$ 、秩 $d = n$ 则得到 Plain-LWE 问题. 此外，对于给定的 Module-LWE 参数，只需调整模秩即可改变问题难度，也即方案安全性.

3.2.2 舍入学习 (Learning With Rounding)

舍入学习由文献 [BPR12] 提出，可视为 LWE 的一种去随机化技术. 在 LWE 中计算 $\langle a, s \rangle + e \bmod q$ ，即将误差项 e 加到内积上 (模 q) . LWR 不采用加法，而是将该内积缩放并舍入到较小的群 $\mathbf{Z}_p$. 具体而言，舍入把 $\mathbf{Z}_q$ 中的元素划分为 p 个相邻区间，并将每个 $\mathbf{Z}_q$ 元素映射到其在 $\mathbf{Z}_p$ 中对应的索引. 必须满足 $p < q$ ，否则不会引入误差. 舍入后的内积 $\lfloor \frac{p}{q} \langle a, s \rangle \rfloor$ 记为

$$\lfloor \langle a, s \rangle \rfloor_p = \left\lfloor \frac{p}{q} \langle a, s \rangle \right\rfloor \in \mathbf{Z}_p.$$

此处 $\mathbf{Z}_q$ (相应地， $\mathbf{Z}_p$) 用 $[0, q[$ (相应地， $[0, p[$) 表示. 注意，当 $p \mid q$ 时， $\mathbf{Z}_q$ 上均匀分布经舍入后在 $\mathbf{Z}_p$ 上仍为均匀分布. 逐系数应用即可将舍入操作扩展到向量或多项式. 与误差学习的环变体和模变体类似，也可定义 LWR 的环版本与模版本.

舍入学习问题的判定版本定义如下.

定义 1 (判定型舍入学习，基于文献 [BPR12, 定义 3.1]). 设 $n \geq 1$ ，模数 $q > p \geq 2$ 均为整数， $\mathbf{R}_q = \mathbf{Z}_q[x]/f(x)$ 如上定义.

- (**Plain-LWR**): 对向量 $s \in \mathbf{Z}_q^n$ ，定义 LWR 分布 L_s 为 $\mathbf{Z}_q^n \times \mathbf{Z}_p$ 上的如下分布：均匀随机选取向量 $a \leftarrow \mathbf{Z}_q^n$ ，输出 $(a, b = \lfloor \langle a, s \rangle \rfloor_p)$.
- (**Ring-LWR**): 对多项式 $s \in \mathbf{R}_q$ ，定义环 LWR 分布 L_s 为 $\mathbf{R}_q \times \mathbf{R}_p$ 上的如下分布：均匀随机选取 $a \leftarrow \mathbf{R}_q$ ，输出 $(a, b = \lfloor a \cdot s \rfloor_p)$.
- (**Module-LWR**): 固定模秩 $d > 1$. 对 $s \in \mathbf{R}_q^d$ ，定义模 LWR 分布 L_s 为 $\mathbf{R}_q^d \times \mathbf{R}_p$ 上的如下分布：均匀随机选取 $a \leftarrow \mathbf{R}_q^d$ ，输出 $(a, \lfloor \langle a, s \rangle \rfloor_p)$.

判定型 (环/模) $LWR_{n,q,p}$ 问题要求以关于 n 不可忽略的优势，区分多项式数量的独立样本 $(a_i, b_i) \leftarrow L_s$ 与分别从 $\mathbf{Z}_q^n \times \mathbf{Z}_p$ 、 $\mathbf{R}_q \times \mathbf{R}_p$ 或 $\mathbf{R}_q^d \times \mathbf{R}_p$ 中均匀、独立抽取的同等数量样本.

3.2.3 舍入学习的安全性

论证 LWR 安全性的总体思路，是证明其安全性与求解判定型 LWE (即与随机量 $\mathbf{u}$ 区分) 的安全性在统计意义上接近 [AKPW13]:

$$(a, \lfloor \langle a, s \rangle \rfloor_p) \stackrel{\text{stat}}{\approx} (a, \langle a, s \rangle + e) \stackrel{\text{comp}}{\approx} (a, \mathbf{u}).$$

文献 [BPR12] 最早研究了 LWR 的安全性，并给出如下结果.

定理 1 (到 LWE 的归约 [BPR12, 定理 3.2]). 设 χ_n 为整数环 $\mathbf{Z}$ 上任意可高效采样的概率分布族，以安全参数 n 为索引，且为 $\beta(n)$ -有界 (即 $\Pr_{x \leftarrow \chi} [|x| > \beta] \leq \text{negl}(n)$). 令 $q \geq p \cdot \beta \cdot n^{\omega(1)}$. 则对于秘密 $s \in \mathbf{Z}_q^n$ 上的任意分布，求解判定型 $LWR_{n,q,p}$ 至少与在相同秘密分布下求解判定型 $LWE_{n,q,\chi}$ 一样困难.

定理 1 要求模数 $q \geq p \cdot \beta \cdot n^{\omega(1)}$ 为超多项式量级. 文献 [AKPW13] 对该结果进行了改进，给出适用于更广参数范围的安全归约：所用模数 q 只需为多项式量级，而非超多项式量

级 [AKPW13, 定理 4.1 与推论 4.2]. 具体而言, 若参数为 n', m, q, β 的 LWE 是困难的, 则只要

$$n \geq \frac{\log(q)}{\log(2\gamma)} \cdot n' \quad \text{且} \quad q \geq \gamma(nm\beta p)$$

对某个参数 $\gamma \geq 1$ 成立, 作者便证明参数为 n, m (样本数)、 q, p 的 LWR 是困难的. 为使归约成立, 模数 q 必须远大于 LWR 样本数 m . 因此, 当攻击者可自行选择 LWR 样本数 (即样本数并非预先固定) 时, 模数 q 又会变为超多项式量级. 模数 q 必须为素数; 否则, 上述不等式须对 q 的最大素因子 $q_{\max}$ 成立. 此外, 该结果要求 q 与 $q/q_{\max}$ 互素, 这意味着当模数 q 为二的幂时该结果不成立.

文献 [BGM⁺16] 的结果消除了这一限制. 首先, 模数 q 的界改进为

$$q \geq 2m\beta p$$

其次, 该结果成立时无需对 q 施加任何数论限制; 特别是, 当模数 q 为二的幂时也成立 [BGM⁺16, 定理 1 与定理 3].

定理 2 (LWR 的单向性 [BGM⁺16, 定理 1]). 设 p, q, n, m, β 为满足 $q > 2p\beta$ 的整数. 对于任意算法 Learn ,

$$\Pr_{A,s,e} [\text{Learn}(A, \lfloor As + e \rfloor_p) = s] \geq \frac{\Pr_{A,s} [\text{Learn}(A, \lfloor As \rfloor_p) = s]^2}{(1 + 2p\beta/q)^m},$$

其中 $A \leftarrow \mathbf{Z}_q^{m \times n}$; 噪声 e 的全部 m 个坐标相互独立, 并且每个坐标均为 β -有界且平衡; s 从支撑集位于 $\mathbf{Z}_q^{n*}$ 上的任意分布中选取.

该结果证明了样本数 m 有界时 LWR 的困难性. 较近期的文献 [NR23] 研究了无界情形, 并发现通过从 LWE 到 LWR 的“逐点”归约来证明安全性存在障碍. 该工作的主要结果及相关猜想要求模数 q 为素数 [NR23, 定理 1 与猜想 1], 而我们的设计并非如此. 需要强调的是, 文献 [NR23] 的结果仅表明证明无界样本情形的安全性需要新思路, 并不意味着 LWR 容易求解.

3.3 作为结构化模的环

在 Module-LWR 场景中, 单个样本形如 $\lfloor \langle \mathbf{A}, \mathbf{s} \rangle \rfloor_p$, 其中 $\mathbf{A}, \mathbf{s} \in \mathbf{R}_{q,n}^k$, k 为模秩. 样本的选取方式使 $\mathbf{A}$ 均匀分布, 而 $\mathbf{s}$ 与 $\mathbf{e}$ 从“小”分布中生成. 若选取 m 个样本, 并假定环结构与模结构无法被利用, 则其安全级别等价于矩阵大小为 $mn \times kn$ 的 Plain-LWR 问题. 取 $m = k$ 即得到典型 KEM 的情形, 此时舍入学习问题由方阵定义. 为简化叙述, 以下均假定 $m = k$.

Module-LWR 的主要优点是可以任意选择 k . 例如, Saber 选择 $k \in \{2, 3, 4\}$. 相比之下, 要达到相同安全级别, 基于分圆环的 Ring-LWR 需从 $R_{q,kn}$ 中采样, 因此必须把 k 限制为二的幂, 无法选择 $k = 3$. 另一方面, Ring-LWR 的优点是单个样本元素 $\mathbf{A}$ 仅含 kn 个不同的 $\mathbf{Z}_q$ 元素 (即一个 $\mathbf{A} \in R_{q,kn}$ 元素), 而 Module-LWR 则含 k^2n 个 (即 $\mathbf{R}_{q,n}^k$ 中元素的 k 个样本). Module-LWR 的二次增长使其效率低于 Ring-LWR, 但 k 的更高灵活性使具体参数集更容易实例化.

众所周知, Ring-LWR 可对应于 (或视为) 一个结构化 Module-LWR 实例. 对于每个满足 $k \mid n$ 的二的幂 k , 存在同构

$$\mathbf{R}_{q,n} = \mathbf{Z}_q[x]/(x^n + 1) \cong \left(\mathbf{Z}_q[y]/(y^{\frac{n}{k}} + 1) \right) [x]/(x^k - y) \quad (2)$$

该同构映射为

$$s(x) = \sum_{i=0}^n s_i x^i \mapsto \sum_{j=0}^{k-1} s_j(y) \cdot x^j, \quad \text{其中} \quad s_j(y) = \sum_{i=0}^{n/k-1} s_{ki+j} \cdot y^i.$$

这仅是将多项式 s 划分为 k 个部分 $s_j(y)$, 每个 $s_j(y)$ 包含次数模 k 等于 j 的系数. 例如, 当 $k=2$ 时, $s_0(y)$ 包含偶数次项的系数, $s_1(y)$ 包含奇数次项的系数. 取两个元素 $\mathbf{A}, \mathbf{s} \in \mathbf{R}_{q,n}$, 在此同构下有

$$\mathbf{A} \cdot \mathbf{s} = \begin{bmatrix} a_0(y) & ya_1(y) \\ a_1(y) & a_0(y) \end{bmatrix} \begin{bmatrix} s_0(y) \\ s_1(y) \end{bmatrix},$$

其中每项运算均在环 $\mathbf{Z}_q[y]/(y^{\frac{n}{2}} + 1)$ 中进行; 特别地, ya_1 对应于 $a_1(y)$ 系数的负循环移位. 这是因为

$$\begin{aligned} \mathbf{A} \cdot \mathbf{s} &= (a_0(y) + a_1(y)x) \cdot (s_0(y) + s_1(y)x) \\ &= a_0s_0(y) + (a_0s_1(y) + a_1s_0(y))x + a_1s_1(y)x^2 \\ &= a_0s_0(y) + ya_1s_1(y) + (a_0s_1(y) + a_1s_0(y))x \bmod (x^2 - y). \end{aligned}$$

当 $k=4$ 时, 则得到

$$\mathbf{A} \cdot \mathbf{s} = \begin{bmatrix} a_0(y) & y \cdot a_3(y) & y \cdot a_2(y) & y \cdot a_1(y) \\ a_1(y) & a_0(y) & y \cdot a_3(y) & y \cdot a_2(y) \\ a_2(y) & a_1(y) & a_0(y) & y \cdot a_3(y) \\ a_3(y) & a_2(y) & a_1(y) & a_0(y) \end{bmatrix} \begin{bmatrix} s_0(y) \\ s_1(y) \\ s_2(y) \\ s_3(y) \end{bmatrix}, \quad (3)$$

此时运算在 $\mathbf{Z}_q[y]/(y^{\frac{n}{4}} + 1)$ 中进行. 更一般地, 容易看出 $\mathbf{A} \cdot \mathbf{s}$ 对应于一个 $k \times k$ 循环矩阵与向量的乘法, 其中矩阵右上三角部分乘以 y . 作为比较, 当 $k=4$ 时, Module-LWR 矩阵乘法的结果为

$$\begin{bmatrix} a_{0,0}(y) & a_{0,1}(y) & a_{0,2}(y) & a_{0,3}(y) \\ a_{1,0}(y) & a_{1,1}(y) & a_{1,2}(y) & a_{1,3}(y) \\ a_{2,0}(y) & a_{2,1}(y) & a_{2,2}(y) & a_{2,3}(y) \\ a_{3,0}(y) & a_{3,1}(y) & a_{3,2}(y) & a_{3,3}(y) \end{bmatrix} \begin{bmatrix} s_0(y) \\ s_1(y) \\ s_2(y) \\ s_3(y) \end{bmatrix},$$

其中算术同样在环 $\mathbf{Z}_q[y]/(y^{\frac{n}{4}} + 1)$ 中进行.

综上, 对于任意 $k \mid n$, 均可将分圆 Ring-LWR 样本视为秩为 k 的结构化 Module-LWR 样本, 其中

$$a_{i,j}(y) = \begin{cases} \text{均匀随机}, & j=0, \\ y \cdot a_{k+i-j,0}(y), & 0 \leq i < j < k, \\ a_{i-j,0}(y), & 0 < j \leq i < k. \end{cases} \quad (4)$$

上述重新解释纯属语法层面, 不影响安全性. 在式 (2) 的环同构下, 分圆 Ring-LWR 样本与式 (4) 所述秩为 k 的结构化 Module-LWR 实例构成双射. 将结构化 Module-LWR 的任意区分器或秘密恢复算法与逆同构复合, 便立即得到 Ring-LWR 的相应算法, 其优势完全相同、运行时间也基本相同. 因此, 将分圆 Ring-LWR 样本视为秩为 k 的结构化 Module-LWR 样本不会改变底层困难性假设或具体估计; 这只是同一分布的不同表示 (相同论证也适用于 LWE).

3.4 根式环舍入学习 (Radical Ring-LWR)

第 3.3 节的观察是以不同方式解释并定义 Ring-LWR 与 Module-LWR 类方案的新环和新参数的关键。

我们不使用二的幂分圆多项式 $f(x) = x^n + 1$ ，而是对二的幂 n 和正整数 k 选择多项式 $f(x) = (x^k - 2)^n + 1$ 。可以证明 f 不可约 [BRVW26, 引理 1]，因此 $\mathbf{Q}[x]/(f(x))$ 是数域，从而可针对该 $f(x)$ 定义 Ring-LWR 型问题。在归藏实例中，舍入在迁移后的单项式基中逐系数进行；该基即根式环单项式基 $\{y^i x^j\}$ 在数域同构下的像。安全性章节通过界定相应的基失真，处理它与通常典范嵌入视角之间的联系。乍看之下，改变多项式似乎会丧失二的幂分圆多项式所带来的优势；然而，存在如下简单环同构：

$$S_{q,n,k} := \mathbf{Z}_q[x]/((x^k - 2)^n + 1) \cong (\mathbf{Z}_q[y]/(y^n + 1)[x])/(x^k - y - 2), \quad (5)$$

$$x^k \mapsto y + 2.$$

虽然 $S_{q,n,k}$ 本身不是分圆环，但它包含一个大的二的幂分圆子环，仍可在其中采用 NTT 等算术优化。因此， $f(x)$ 所定义的域是第 $2n$ 个分圆域的 k 次扩张；扩张次数 k 的作用类似于模秩，使底层格问题的困难性比必须取二的幂次数 n 的分圆环更易调节。由于多项式 $x^k - y - 2$ 定义了第 $2n$ 个分圆域的 k 次根式扩张，在舍入学习场景中将此类环称为根式 Ring-LWR (RR-LWR)。

注意，即便不局限于二的幂 n ，而考虑更一般的第 m 个分圆域，也仍无法构造既含大型二的幂分圆子域、又具有给定次数的分圆域。事实上，第 m 个分圆域的次数为 Euler 函数 $\varphi(m)$ ，因此可能的扩张次数受限于此类整数；若出于效率需要还要求大型二的幂子域，限制会更严格。例如，不存在满足 $\varphi(m) = 3$ 或 $\varphi(m) = 5$ 的整数 m ，因而不存在对应于二的幂分圆域 3 次或 5 次扩张的分圆域。采用上述 $f(x)$ 后，只需令 $k = 3$ 或 $k = 5$ 即可直接得到此类域。

与通用模结构相比，上述构造通过施加环结构降低了相对于 Module-LWR 的计算复杂度和随机性需求。也就是说，矩阵 $\mathbf{A}$ 由分圆环中的 k 个而非 k^2 个均匀随机元素组成。更具体地，对元素 $\mathbf{A}, \mathbf{s} \in S_{q,n,k}$ ，写作 $\mathbf{A} = \sum_{i=0}^{k-1} a_i(y)x^i$ 且 $\mathbf{s} = \sum_{i=0}^{k-1} s_i(y)x^i$ ，则多项式乘法 $\mathbf{A} \cdot \mathbf{s}$ 对应于矩阵乘法

$$\begin{bmatrix} a_0(y) & (y+2) \cdot a_{k-1}(y) & (y+2) \cdot a_{k-2}(y) & \cdots & (y+2) \cdot a_1(y) \\ a_1(y) & a_0(y) & (y+2) \cdot a_{k-1}(y) & \cdots & (y+2) \cdot a_2(y) \\ \vdots & \vdots & \vdots & & \vdots \\ a_{k-1}(y) & a_{k-2}(y) & a_{k-3}(y) & \cdots & a_0(y) \end{bmatrix} \begin{bmatrix} s_0(y) \\ s_1(y) \\ s_2(y) \\ \vdots \\ s_{k-1}(y) \end{bmatrix}. \quad (6)$$

该矩阵乘法的全部运算均在分圆环 $\mathbf{Z}_q[y]/(y^n + 1)$ 中进行。与式 (3) 的 Ring-LWR 矩阵结构相比，可以看出它实质上只是将乘法因子 y 替换为 $y + 2$ 。

在不同于分圆环的数环上实例化 Ring-LWR 时必须谨慎。安全性章节将本实例形式化为根式数域上的迁移基 Ring-LWR 变体，并在具体估计中计入所用的基变换。

4 算法描述

本节描述构造 玄色 (Octarine) 数字签名方案所需的全部组件。完整算法见算法 3。

参数集与安全级别。 按照提交要求, 算法采用参数化设计以支持多个安全级别. 规定三个参数集, 分别以 128、256 和 512 比特经典安全强度为目标; 估计相应抗量子安全强度分别不低于 80、128 和 256 比特.

各参数集实例化相同算法结构, 仅在决定底层格维数、模数大小及若干签名算法常数的参数选择上有所不同. 该统一设计使所有安全级别可使用一致的实现框架, 同时通过参数选择细粒度调节安全性. 完整参数列表见表 2.

环的选择。 本方案在如下形式的根式环上运算:

$$\mathbf{S}_{q,n,k} = (\mathbf{Z}_q[y]/(y^n + 1))[x]/(x^k - y - 2), \quad (7)$$

其中 q 和 n 为二的幂, k 为任意正整数; 软猬 (Mithril) 使用相同的环. 它与数域表示 $\mathbf{S}_{q,n,k} \cong \mathbf{Z}_q[x]/((x^k - 2)^n + 1)$ 存在直接同构 [BRVW26, 第 4 节], 由此得到迁移基 Ring-LWR 变体所用的根式环表示. 环 $\mathbf{S}_{q,n,k}$ 是第 $2n$ 个分圆域的 k 次根式扩张, 因而可按需使用 NTT 或其他多项式乘法例程等标准实现技术; 与模结构相比, 根式扩张还可更高效地生成矩阵.

所有参数集统一取 $q = 2^{24}$ 、 $n = 1024$, 改变 k 以得到适当矩阵维数. 选择比特长度为 8 的倍数的 q 以简化实现, 并尽量增大 n , 以便按需使用 NTT. 此外, q 的大小恰为 软猬 (Mithril) 所用素数的两倍, 更易复用其构件. 该环虽适合 NTT, 方案仍将 NTT 设计为完全可选. Ring-LWR 问题由舍入误差 q/p 定义. 选择 p , 使 $\mathbf{s}_1$ 从 $[-\eta, \eta - 1]$ 均匀采样, 且导出舍入误差 $\mathbf{s}_2 = \mathbf{A}\mathbf{s}_1 - \frac{q}{p}\mathbf{b}$ 在相同系数区间内有界, 其中 $\eta = q/(2p)$. 启发式地, 若将 $\mathbf{A}\mathbf{s}_1$ 视为模 q 均匀, 则导出误差在该区间上接近均匀; 这是不同于对舍入值 $\mathbf{b}$ 的 RR-LWR 假设的更强均匀乘积启发式. $\mathbf{s}_1$ 与 $\mathbf{s}_2$ 每个系数均恰可用 $\log_2(2\eta)$ 比特表示, 在各参数集中为 2 或 3 比特.

哈希。 玄色 (Octarine) 需要哈希函数, 更准确地说是 XOF, 用于实例化随机预言机、生成消息摘要, 并为密钥材料和临时秘密生成伪随机比特. 所有哈希和 XOF 输入均进行域分离和长度定界.

首先, 令 $P = \texttt{``ARCANE-Octarine-v1''}$ 为固定协议前缀. 对于标签 L 、输出字节长度 o 和 t 个字节串字段 $X_1, \dots, X_t$, 头部 $\text{Head}(L, o, t)$ 为下列字段的串接, 全部整数均采用小端编码.

字段	值	字节数
协议前缀	P	18
标签长度	$ L $	1
标签	L	$ L $
安全级别	λ	2
模秩	k	1
输出长度	o	4
字段数量	t	1

每个变长字段均以自身长度为前缀, 带标签输入由头部及随后各编码字段组成:

$$\text{Field}(X) = \text{IntegerToBytes}(|X|, 8) \parallel X,$$

$$\text{Encode}(L, o; X_1, \dots, X_t) = \text{Head}(L, o, t) \parallel \text{Field}(X_1) \parallel \dots \parallel \text{Field}(X_t).$$

	玄色 (Octarine) -128	玄色 (Octarine) -256	玄色 (Octarine) -512
n	1024	1024	1024
k	1	2	5
$\epsilon_q = \log_2 q$	24	24	24
$\epsilon_p = \log_2 p$	21	22	22
η	4	2	2
d	11	13	14
τ	16	36	87
λ	128	256	512
γ_1	2^{18}	2^{19}	2^{21}
γ_2	2^{17}	2^{18}	2^{20}
$\beta = \tau\eta$	64	72	174
ω	79	210	399
挑战熵	131	257	512
重复次数	2.1	2.3	3.6
$D_{\mathbf{A}}$	$\mathcal{U}_{\epsilon_q}$	$\mathcal{U}_{\epsilon_q}$	$\mathcal{U}_{\epsilon_q}$
$D_{\mathbf{s}_1}$	$\mathcal{U}_4$	$\mathcal{U}_2$	$\mathcal{U}_2$
私钥 (字节)	2 432	4 608	11 776
公钥 (字节)	1 344	2 368	5 184
签名 (字节)	2 564	5 449	14 713
	<u>LWR 困难度 (uSVP)</u>		
BKZ 块大小 (b)	434	935	2788
经典 Core-SVP (0.292 b)	127	273	814
量子 Core-SVP (0.265 b)	115	248	739
精化估计 (门数)	153	294	817
	<u>LWR 困难度 (BDD)</u>		
BKZ 块大小 (b)	424	924	2181
经典 Core-SVP (0.292 b)	124	270	637
量子 Core-SVP (0.265 b)	112	245	578
精化估计 (门数)	150	291	686
	<u>SIS 困难度 (EUF-CMA; 括号内为 SUF-CMA)</u>		
BKZ 块大小 (b)	439 (419)	910 (916)	2340 (2340)
经典 Core-SVP (0.292 b)	128 (122)	266 (267)	683 (683)
量子 Core-SVP (0.265 b)	116 (111)	241 (242)	620 (620)
精化估计 (门数)	156 (150)	288 (290)	691 (692)

Table 2: 参数集、编码大小与困难度估计. SIS 困难度各行的主条目是表 1 使用的 EUF-CMA 估计, 括号内为相应 SUF-CMA 估计.

记

$$H_L(X_1, \dots, X_t; o) = H(\text{Encode}(L, o; X_1, \dots, X_t), o).$$

签名方案使用的随机预言机调用如下：

- 密钥派生： $(\rho, \rho', K) \leftarrow H_{\text{kdf}}(\xi; |\rho| + |\rho'| + |K|)$ 将初始随机性扩展为公开种子、秘密种子和签名密钥。参数集的 λ 与 k 包含在公共编码头中。
- 公钥哈希： $tr \leftarrow H_{\text{tr}}(pk; |tr|)$ 将公钥压缩为签名和验证所用的短摘要。
- 消息代表元： $\mu \leftarrow H_{\text{mu}}(tr, M; |\mu|)$ 将消息与 tr 一同预哈希，使签名算法处理定长输入；变长消息 M 以自身长度编码。
- 签名随机性： $\rho'' \leftarrow H_{\text{rhop}}(K, rnd, \mu; |\rho''|)$ 导出每次尝试采样 y 的随机性。加入 K 可对冲低质量随机性，加入 rnd 可提供 nonce 新鲜性。
- 挑战哈希： $\tilde{c} \leftarrow H_{\text{ch}}(\mu, w_1; |\tilde{c}|)$ 从消息代表元和承诺高位导出 Fiat-Shamir 挑战种子。

每次调用分别规定 H 的输出长度。安全证明要求 H 表现为随机预言机，具有 $\lambda/2$ 比特碰撞抗性和 λ 比特原像抗性，其中 λ 为目标安全级别。只要满足安全要求，方案安全性不依赖具体哈希函数。参考与优化实现族以基于 SM3 的 `pseudoXOF` 实例化 H ；附加 SHA3/SHAKE 实现族使用 SHAKE256。由于 SHAKE256 容量为 512 比特，最多达到 256 比特安全性，规范的 512 比特实现使用基于 SM3 的 `pseudoXOF` 而非 SHAKE256。

其次，还需要 XOF 生成伪随机比特，以从种子扩展公私钥材料；该过程用于生成 s_1 、 $\mathbf{A}$ 、 y 和 c （用于 `SampleInBall`）。公开数据 $\mathbf{A}$ 使用 XOF_A ，秘密采样使用 XOF_S ，挑战扩展使用 XOF_C 。挑战扩展 XOF 使用相同 Encode 格式，标签为 `XOF_C`。大容量种子扩展 XOF XOF_A 和 XOF_S 使用紧凑长度定界编码，以避免实现中不必要的额外压缩块：

$$\begin{aligned} & \text{AOX1} \parallel \text{id} \parallel \text{IntegerToBytes}(\lambda, 2) \parallel \text{IntegerToBytes}(k, 1) \parallel \text{IntegerToBytes}(o, 2) \\ & \parallel \text{IntegerToBytes}(|s|, 1) \parallel s \parallel \text{IntegerToBytes}(i, 1) \parallel \text{IntegerToBytes}(\ell, 1). \end{aligned}$$

其中 XOF_A 取 $\text{id} = 1$ ， XOF_S 取 $\text{id} = 2$ ； s 为传入 XOF 的种子， i 为多项式或流索引， ℓ 为通道索引。方案使用的种子短于 256 字节，请求的 XOF 输出短于 2^{16} 字节，故每次已实现调用均保持单射和长度定界。 XOF_A 底层具体原语记为 G ， XOF_S 与 XOF_C 的底层原语记为 F 。参考与优化实现族均以基于 SM3 的 `pseudoXOF` 实例化二者；附加 SHA3/SHAKE 实现族分别使用 SHAKE128 与 SHAKE256。

采样分布。 秘密值和公开值从不同均匀分布中采样。除非另有说明，比特和字节寻址采用小端序。采样一个 $n = 1024$ 的多项式可能需要顺序执行多轮置换，形成潜在瓶颈；对需要较多随机字节的 $\mathbf{A}$ 和 y 尤其如此。为提高并行度，利用域分离独立生成每块 256 个多项式系数。 s 每个系数只需 2 或 3 比特，性能影响很小，故不采用该方式。

- $D_{\mathbf{A}}(\mathbf{S}_{q,n,k}, \rho)$ ：使用可扩展输出函数，将种子 ρ 确定性地扩展为 $\mathbf{S}_{q,n,k}$ 中的均匀随机元素。生成 $\mathbf{A}$ 共需 k 个各含 n 个系数的多项式，每个系数由 $\epsilon_q = 24$ 比特组成。记 $\mathbf{A} = \sum_{i=0}^{k-1} \mathbf{A}_i(y)$ ，且 $\mathbf{A}_i(y) = \sum_{j=0}^{n-1} \alpha_{i,j} y^j$ 。采样分两步进行。首先由 ρ 生成字节流。对每个多项式，将索引 i 和四个通道索引作为独立编码字段。具体而言， $\mathbf{A}_i(y)$ 的字节流生成为 $B_{i,0} = XOF_A(\rho, \text{IntegerToBytes}(i, 1), \text{IntegerToBytes}(0, 1); 768), \dots, B_{i,3} = XOF_A(\rho, \text{IntegerToBytes}(i, 1), \text{IntegerToBytes}(3, 1); 768)$ ，其中每个 $B_{i,j}$ 输出 $24 \cdot 256/8 = 768$ 字节。第二步，令 $\alpha_{i,j} = B_{i,j'}[3j''] + B_{i,j'}[3j'' + 1] \cdot 2^8 + B_{i,j'}[3j'' + 2] \cdot 2^{16}$ ，其中 $j' = \lfloor j/256 \rfloor$ 、 $j'' = j \bmod 256$ 。

- $D_y(\mathbf{S}_{q,n,k}, \rho'', \kappa)$: 采样一个环元素, 其系数从 $[-\gamma_1, \gamma_1 - 1]$ 中独立均匀选取. 生成 $\mathbf{y}$ 共需 k 个各含 n 个系数的多项式, 每个系数由 $\log_2(2\gamma_1)$ 比特组成. 计数器 κ 保证每次签名尝试均使用全新且相互独立的样本. 记 $\mathbf{y} = \sum_{i=0}^{k-1} \mathbf{y}_i(y)$, 且 $\mathbf{y}_i(y) = \sum_{j=0}^{n-1} \xi_{i,j} y^j$. 采样分两步进行. 首先构造本次尝试的种子 $\rho''_\kappa = \rho'' \parallel \text{IntegerToBytes}(\kappa, 4)$. 对每个多项式, 将索引 i 和四个通道索引作为独立编码字段. 即 $\mathbf{y}_i(y)$ 的字节流生成为 $B_{i,0} = \text{XOF}_5(\rho''_\kappa, \text{IntegerToBytes}(i, 1), \text{IntegerToBytes}(0, 1); \log_2(2\gamma_1) \cdot 256/8), \dots, B_{i,3} = \text{XOF}_5(\rho''_\kappa, \text{IntegerToBytes}(i, 1), \text{IntegerToBytes}(3, 1); \log_2(2\gamma_1) \cdot 256/8)$, 其中每个 $B_{i,j}$ 输出 $\log_2(2\gamma_1) \cdot 256$ 比特. 第二步, 令 $\xi_{i,j} = B_{i,j'}[\log_2(2\gamma_1)j'' : \log_2(2\gamma_1)(j'' + 1)]$, 其中 $B_{i,j'}[a : b]$ 表示从比特索引 a (含) 到 b (不含) 的片段, $j' = \lfloor j/256 \rfloor$, $j'' = j \bmod 256$.
- $D_s(\mathbf{S}_{q,n,k}, \rho')$: 采样一个环元素, 其 nk 个整数系数从 $[-\eta, \eta - 1]$ 中独立均匀选取. 生成 $\mathbf{s}$ 共需 k 个各含 n 个系数的多项式, 每个系数由 $\theta = \log_2(2\eta)$ 比特组成. 记 $\mathbf{s} = \sum_{i=0}^{k-1} \mathbf{s}_i(y)$, 且 $\mathbf{s}_i(y) = \sum_{j=0}^{n-1} \sigma_{i,j} y^j$. 首先, 以种子 ρ' 调用 XOF_5 , 输出 k 个字节流 $B_i = \text{XOF}_5(\rho', \text{IntegerToBytes}(i, 1), \text{IntegerToBytes}(0, 1); \theta n/8)$. 第二步, 令 $\sigma_{i,j} = B_i[\theta j : \theta(j + 1)]$, 其中 $B_i[a : b]$ 表示从比特索引 a (含) 到 b (不含) 的片段.

舍入与除法. 按惯例, 模二的幂系数模数 Q 的整数在区间 $[-Q/2, Q/2 - 1]$ 内表示. 对于任意二的幂因子 $m \mid Q$ 和 $x \in [-Q/2, Q/2 - 1]$, 可唯一写成 $x = x_0 + x_1 m \bmod Q$, 其中 $x_0 \in [-m/2, m/2 - 1]$, $x_1 \in [-Q/(2m), Q/(2m) - 1]$. 可按 $x_0 = x \bmod m$, $x_1 = (x - x_0)/m \bmod Q/m$ 计算. 舍入例程以下标表示环境模数. 在此基础上定义:

- $\text{HighBits}_q(\mathbf{w}, 2\gamma_2)$ 返回环元素 $\mathbf{w}_1 = ((\mathbf{w} - (\mathbf{w} \bmod 2\gamma_2))/(2\gamma_2)) \bmod q/(2\gamma_2)$, 其中所有操作均逐系数应用.
- $\text{Power2Round}_p(\mathbf{b}, 2^d)$ 返回环元素 $\mathbf{b}_0 = \text{LowBits}_p(\mathbf{b}, 2^d)$, 其中所有操作均逐系数应用, 同时返回 $\mathbf{b}_1 = \text{HighBits}_p(\mathbf{b}, 2^d)$. 指数 d 及模数 2^d 属于参数集定义.
- $\mathbf{b} = \lfloor_q \mathbf{t} \rfloor$ 返回舍入后的环元素. 实现层面, 令 $\langle x \rangle_Q \in \{0, \dots, Q - 1\}$ 表示 $x \bmod Q$ 的无符号代表元, $\text{ctr}_Q(u) = ((u + Q/2) \bmod Q) - Q/2$ 表示中心代表元. 逐系数计算舍入余数

$$u = (\langle \mathbf{t} \rangle_q + q/(2p)) \bmod q, \quad v = u \gg (\log_2 q - \log_2 p) \in \{0, \dots, p - 1\}, \quad \mathbf{b} = \text{ctr}_p(v).$$

其中 $\gg$ 表示无符号右移; 实现可将 v 保留为 $\mathbf{b} \bmod p$ 的无符号代表元. 若目标环境保证中心有符号代表元的算术 (符号传播) 右移语义, 也可先加 $q/(2p)$, 右移后模 p 约减. 可移植 C 实现应在右移前构造无符号代表元, 因为 C 中负有符号整数右移属于实现定义行为.

算法描述见算法 1.

公钥压缩. 公钥仅存储通过 Power2Round_p 得到的 LWR 公钥 $\mathbf{b}$ 的高位 $\mathbf{b}_1$. 验证时, 验证者必须恢复承诺 $\mathbf{w} = \mathbf{A}\mathbf{y}$ 的高位 $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$. 但验证者不知道 $\mathbf{b}_0$, 只能计算近似值 $\mathbf{w}'_{\text{Approx}} = \mathbf{w} + c\mathbf{s}_2 + c_p^q \mathbf{b}_0$; 它与 $\mathbf{w} + c\mathbf{s}_2$ 相差小量 $c_p^q \mathbf{b}_0$. 若扰动导致高位进位, 验证者会得到错误的 $\mathbf{w}_1$. 为校正该问题, 签名者加入紧凑提示向量 $\mathbf{h}$, 编码发生进位的系数位置, 使验证者可准确恢复 $\mathbf{w}_1$. 即提示准确编码如下二者不同的系数位置: $\text{HighBits}_q(\mathbf{w} + c\mathbf{s}_2, 2\gamma_2)$ 与 $\text{HighBits}_q(\mathbf{w} + c\mathbf{s}_2 + c_p^q \mathbf{b}_0, 2\gamma_2)$. 为此, 若提示系数置位, 验证者便校正 $\mathbf{w}'_{\text{Approx}}$: 低位为正则加 1, 为负则减 1. MakeHint_q 和 UseHint_q 的算法描述见算法 1, 二者逐系数扩展到完整环元素.

由于 q 与 $2\gamma_2$ 均为二的幂且 $2\gamma_2 \mid q$, 高低位分解是直接的比特提取. 与 Dilithium [DKL⁺18] (后成为 ML-DSA 标准) 等素数模数情形不同, $q - 1$ 处不存在边界异常, 也无需专用 Decompose

Algorithm 1 签名方案的辅助算法.

HighBits_Q(r, m) 1: $r_0 \leftarrow r \bmod m$ 2: return $((r - r_0) / m) \bmod Q/m$ LowBits_Q(r, m) 1: return $r \bmod m$ Power2Round_Q(r, m) 1: $r_0 \leftarrow \text{LowBits}_Q(r, m)$ 2: $r_1 \leftarrow \text{HighBits}_Q(r, m)$ 3: return (r_1, r_0)	MakeHint_Q(z, r, m) 1: $r_1 \leftarrow \text{HighBits}_Q(r, m)$ 2: $v_1 \leftarrow \text{HighBits}_Q(r + z, m)$ 3: return $\llbracket r_1 \neq v_1 \rrbracket$ UseHint_Q(h, r, m) 1: $r_1, r_0 \leftarrow \text{Power2Round}_Q(r, m)$ 2: if $h = 1$ and $r_0 > 0$ then 3: return $(r_1 + 1) \bmod Q/m$ 4: else if $h = 1$ and $r_0 \leq 0$ then 5: return $(r_1 - 1) \bmod Q/m$ 6: end if 7: return r_1
--	---

* 下标 Q 表示环境系数模数. 本方案中, Power2Round_p 对 $\mathbf{b} \in S_p$ 以 $m = 2^d$ 调用 LowBits_p 和 HighBits_p ; 其余例程使用 $Q = q$ 和 $m = 2\gamma_2$. 所有操作均逐系数扩展到根式环元素, 并逐分量扩展到向量.

子例程, 提示机制因而简化为一次进位方向检查.

随着 d 增大, 公钥压缩程度提高, 但签名过程的失败率也随之上升. 我们以启发式方法选择 d : 取实现测量中使 $c_p^q \mathbf{b}_0$ 界检查及随后的提示重量检查仅偶尔造成拒绝的最大值. 这一选择不影响安全性, 并可用参考实现复现实验.

密钥与签名编码. 我们将密钥和签名按小端序序列化为紧凑字节数组. 前文描述以有符号整数表示元素, 但序列化时选择打包和解包无符号整数. 字节串存储无符号代表元, 解包时再映射回中心化有符号代表元, 从而避免符号扩展带来的实现问题. 对于一列 b 比特系数 $(w_i)_{0 \leq i < N}$, 定义

$$\text{Enc}_b(a) = (2^{b-1} - 1 - a) \bmod 2^b \in \{0, \dots, 2^b - 1\}$$

以及

$$\text{Pack}((w_i)_{0 \leq i < N}, b) = \text{Enc}_b(w_0) \parallel \dots \parallel \text{Enc}_b(w_{N-1})$$

作为其序列化, 其中拼接方式使 w_0 存于最低有效地址, 且 $\text{Enc}_b(w_0) \bmod 2$ 对应第 0 比特. 解包应用逆映射 $\text{Dec}_b(u) = 2^{b-1} - 1 - u$, 返回区间 $[-2^{b-1}, 2^{b-1} - 1]$ 中的系数. 对于系数宽度为 b 比特的环元素 $\mathbf{w} = \sum_{i=0}^{k-1} (\sum_{j=0}^{n-1} w_{i,j} y^j) x^i \in \mathbf{S}_{q,n,k}$, 定义其序列化为

$$\text{Pack}(\mathbf{w}, b) = \text{Pack}((w_{0,j})_{0 \leq j < n}, b) \parallel \dots \parallel \text{Pack}((w_{k-1,j})_{0 \leq j < n}, b),$$

其中 $\text{Pack}((w_{0,j})_{0 \leq j < n}, b)$ 同样存于最低有效地址.

- 定义公钥的序列化为

$$\text{pkEncode}(\rho, \mathbf{b}_1) = \rho \parallel \text{Pack}(\mathbf{b}_1, \log_2(p) - d).$$

相应的反序列化函数为 $\text{pkDecode}(pk)$. 公钥总大小为 $64 + (\log_2(p) - d)kn/8$ 字节.

- 定义私钥的序列化为

$$\begin{aligned} \text{skEncode}(\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{b}_0) = & \rho \parallel K \parallel tr \\ & \parallel \text{Pack}(\mathbf{s}_1, \log_2(2\eta)) \\ & \parallel \text{Pack}(\mathbf{s}_2, \log_2(2\eta)) \\ & \parallel \text{Pack}(\mathbf{b}_0, d). \end{aligned}$$

相应的反序列化函数为 $\text{skDecode}(sk)$. 私钥总大小为 $64 + 64 + 128 + (2\log_2(2\eta) + d)kn/8$ 字节.

- 签名的序列化需要编码 MakeHint_q 函数输出的提示向量; 该向量由 kn 个二元系数位置组成, 排列为 k 个各含 n 个系数的多项式. 我们沿用 Dilithium/ML-DSA [DKL⁺18] 的方法, 将提示编码为总计 ω 个系数索引, 并另用 k 个索引编码切换至下一个多项式的位置. 因为 $n = 1024$, 每个系数索引需用 $\log_2(1024) = 10$ 比特表示. 因此提示总大小为 $10(\omega + k)/8$ 字节; 选择 ω 使 $10(\omega + k) \equiv 0 \pmod{8}$, 即 $(\omega + k) \equiv 0 \pmod{4}$. 该函数称为 hintEncode . 编码是规范的: 每个多项式内编码的提示索引严格递增, k 个分隔索引单调不减且位于 0 与 ω 之间, 最后一个分隔符之后所有未使用的索引槽均为零. 解码例程 sigDecode 拒绝提示编码不规范的任何签名. 我们以启发式方法选择 ω , 使生成超过 ω 个提示系数的概率很小; 参考实现中包含用于验证这些选择的宏. ω 的取值不影响方案安全性, 但能以适度降低签名速度为代价缩短签名.

最后定义

$$\text{sigEncode}(\tilde{c}, \mathbf{z}, \mathbf{h}) = \tilde{c} \parallel \text{Pack}(\mathbf{z}, \log_2(2\gamma_1)) \parallel \text{hintEncode}(\mathbf{h}).$$

相应的反序列化函数为 $\text{sigDecode}(\sigma)$. 签名总大小为 $\lambda/4 + \log_2(2\gamma_1)kn/8 + 10(\omega + k)/8$ 字节.

带对冲签名. 与现代签名方案的通常做法一致, 我们将字节数组 rnd 纳入临时随机性的派生函数, 从而同时支持随机化签名和确定性签名. 将 rnd 设为随机值会得到随机化签名; 将其固定为已知常量则得到确定性变体. 在需要防范物理侧信道攻击 (physical side-channel attack) 的环境中, 建议使用随机化版本.

带中止 Fiat-Shamir 循环. 带中止 Fiat-Shamir (Fiat-Shamir with aborts) 循环在以下任一条件成立时拒绝: $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$, $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$, $\|c_p^g \mathbf{b}_0\|_\infty \geq \gamma_2$, or $\text{HW}(\mathbf{h}) > \omega$. 前两个条件主导拒绝概率, 可按如下方式解析估计. 容易证明 (例如见 [LDK⁺22, Section 3.4]), $\mathbf{z}$ 的每个系数 z_i 满足 $|z_i| < \gamma_1 - \beta$, 当且仅当 $-\gamma_1 + \beta - \sigma_i < y_i < \gamma_1 - \beta - \sigma_i$, 其中 y_i 和 σ_i 分别是 $\mathbf{y}$ 与 $\mathbf{cs}_1$ 的对应系数. 由于 $\mathbf{y}$ 的 kn 个系数均匀分布于 $[-\gamma_1, \gamma_1 - 1]$, $\mathbf{z}$ 通过 z 界检查的概率为

$$\Pr[\|\mathbf{z}\|_\infty < \gamma_1 - \beta] = \left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_1} \right)^{kn} = \begin{cases} 0.777 & \text{对应玄色 (Octarine) -128,} \\ 0.753 & \text{对应玄色 (Octarine) -256,} \\ 0.653 & \text{对应玄色 (Octarine) -512.} \end{cases}$$

该概率不依赖任何秘密数据. 类似地, 假设中心化低位 $\mathbf{r}_0 = \text{LowBits}_q(\mathbf{w} + c\mathbf{s}_2, 2\gamma_2)$ 均匀分布于 $[-\gamma_2, \gamma_2 - 1]$, 则通过低位检查的概率为

$$\Pr[\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta] = \left(\frac{2(\gamma_2 - \beta) - 1}{2\gamma_2} \right)^{kn} = \begin{cases} 0.604 & \text{对应玄色 (Octarine) -128,} \\ 0.568 & \text{对应玄色 (Octarine) -256,} \\ 0.427 & \text{对应玄色 (Octarine) -512.} \end{cases}$$

由于 $\mathbf{r}_0$ 的通过率低于 $\mathbf{z}$, 实现中先依据 $\mathbf{r}_0$ 执行拒绝更为高效, 可减少发生拒绝前已经完成的计算.

此外, c 恰含 τ 个取值为 1 或 -1 的非零元素, 且 $\mathbf{b}_0$ 的每个系数均位于 $[-2^{d-1}, 2^{d-1} - 1]$, 故 $c_p^q \mathbf{b}_0$ 每个系数的绝对值可由 $\frac{q}{p}\tau 2^{d-1}$ 上界. 这是正确性与安全性估计采用的保守逐系数界; 对表 2 中的参数集, 该界在 玄色 (Octarine) -128 中等于 γ_2 , 在 玄色 (Octarine) -256 和 玄色 (Octarine) -512 中则超过 γ_2 . 然而这一最坏情况界较为宽松: 它对应全部 τ 个带符号贡献同向叠加, 而典型密钥和挑战会使稀疏乘积产生大量抵消. 因此, 我们通过实验检查 $c_p^q \mathbf{b}_0$ 界检查和提示重量检查的影响, 而不只根据该确定性界推断.

参考实现对 玄色 (Octarine) -128、玄色 (Octarine) -256 和 玄色 (Octarine) -512 各生成 5000 个签名的一次运行中, $c_p^q \mathbf{b}_0$ 界检查导致的拒绝轮数分别为 0、30 和 1, 提示重量检查导致的相应轮数分别为 36、32 和 8. 计入所有拒绝检查后, 平均循环轮数分别为 2.10、2.29 和 3.54. 玄色 (Octarine) -512 的拒绝数低于 玄色 (Octarine) -256, 说明这些经验分布尾部由典型抵消尺度以及选定的 γ_2 和 ω 决定, 而非仅由线性最坏情况界决定. 表 2 采用乘积

$$\frac{1}{\Pr[\|\mathbf{z}\|_\infty < \gamma_1 - \beta] \cdot \Pr[\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta]}$$

的倒数作为完成一次签名所需平均轮数的解析近似. 上述经验均值与该双检查估计接近, 并在本次运行中略低; 少量额外拒绝处于启发式均匀性假设和有限测量样本所引入的波动范围内.

挑战生成. 函数 $\text{SampleInBall}(\tilde{c})$ 将哈希摘要 $\tilde{c}$ 确定性地映射为挑战多项式 $c \in \mathbf{R}_{q,n}$; 该多项式恰有 τ 个非零系数, 且每个非零系数均属于 $\{-1, +1\}$. 种子 $\tilde{c}$ 由 XOF_C 扩展. 符号比特流使用流标识符 0 和计数器 0, 索引流使用流标识符 1 和 32 比特计数器. 该算法对 $\mathbf{R}_{q,n}$ 中多项式的 n 个整数系数执行由内向外的 Fisher-Yates 洗牌 (inside-out Fisher-Yates shuffle). 由于 $n = 1024$, 每个候选洗牌索引均从 XOF 流读取为 10 比特字; 若其不在 $\{0, \dots, i\}$ 中则予以拒绝. 完整描述见算法 2.

挑战熵为 $\log_2 \binom{n}{\tau} + \tau$ 比特, 即所有带符号、汉明重量 (Hamming weight) 为 τ 的挑战上的均匀分布熵. 由于选择 $n = 1024$, 与 $n = 256$ 的模 LWE (Module-LWE) 方案相比可采用更小的 τ , 从而减小 β 并减少签名循环的中止次数. 我们取使精确挑战熵至少为 λ 的最小 τ ; 表 2 给出向下取整到整数比特的相应熵值.

在签名算法中, 将 c 作为常数分量嵌入 $\mathbf{S}_{q,n,k}$, 即 $c := c(y) + \sum_{i=1}^{k-1} 0 \cdot x^i$. 这模拟了 M-LWE 中 c 是单个环元素而非完整模元素的情形. 因此, 对 $\mathbf{s} = \sum_{i=0}^{k-1} s_i x^i$, 乘以 c 按分量作用为 $c\mathbf{s} = \sum_{i=0}^{k-1} (cs_i) x^i$.

验证条件. 下面概述诚实生成的签名为何总能通过验证. 回顾 KeyGen (见算法 3): $\mathbf{b} = \lfloor \frac{p}{q} \mathbf{A} \mathbf{s}_1 \rfloor \in \mathbf{S}_{p,n,k}$ 是 LWR 公钥, $\mathbf{s}_2 = \mathbf{A} \mathbf{s}_1 - \frac{q}{p} \mathbf{b}$ 是舍入误差 (故 $\mathbf{A} \mathbf{s}_1 - \mathbf{s}_2 = \frac{q}{p} \mathbf{b}$), 且 $(\mathbf{b}_1, \mathbf{b}_0) = \text{Power2Round}_p(\mathbf{b}, 2^d)$.

Algorithm 2 挑战多项式采样.

SampleInBall($\tilde{c}$)

- 1: $c = (c_0, c_1, \dots, c_{n-1}) \leftarrow \mathbf{0}$
- 2: $\text{idx_ctr} \leftarrow 0$
- 3: $(s_0, \dots, s_{\tau-1}) \leftarrow XOF_{\mathbb{C}}(\tilde{c}, \text{IntegerToBytes}(0, 1), \text{IntegerToBytes}(0, 4); \lceil \tau/8 \rceil) \triangleright$ Sample the sign bits
- 4: $(B_0, \dots, B_{79}) \leftarrow XOF_{\mathbb{C}}(\tilde{c}, \text{IntegerToBytes}(1, 1), \text{IntegerToBytes}(\text{idx_ctr}, 4); 80) \triangleright$ Sample byte buffer for 64 indices
- 5: $\text{bitpos} \leftarrow 0$
- 6: **for** $i = n - \tau$ **to** $n - 1$ **do**
- 7: **do**
- 8: **if** $\text{bitpos} + 10 > 640$ **then**
- 9: $\text{idx_ctr} \leftarrow \text{idx_ctr} + 1$
- 10: $(B_0, \dots, B_{79}) \leftarrow XOF_{\mathbb{C}}(\tilde{c}, \text{IntegerToBytes}(1, 1), \text{IntegerToBytes}(\text{idx_ctr}, 4); 80)$
- 11: $\text{bitpos} \leftarrow 0$
- 12: **end if**
- 13: $j \leftarrow 0$
- 14: **for** $\ell = 0$ **to** 9 **do**
- 15: $j \leftarrow j + 2^\ell \cdot \text{bit}(B_0, \dots, B_{79}; \text{bitpos})$
- 16: $\text{bitpos} \leftarrow \text{bitpos} + 1$
- 17: **end for**
- 18: **while** $(j > i)$
- 19: $c_i \leftarrow c_j$
- 20: $c_j \leftarrow (-1)^{s_{i-(n-\tau)}}$
- 21: **end for**
- 22: **return** c $\triangleright$ Interpret as an element of $\mathbf{R}_{q,n}$

* 输出恰含 τ 个非零系数, 每个均属于 $\{-1, +1\}$. 该算法对 $\mathbf{R}_{q,n}$ 中多项式的 n 个整数系数执行由内向外的 Fisher-Yates 洗牌. 由于 $n = 1024$, 洗牌索引以 10 比特字采样.

签名时, 签名者计算 $\mathbf{w} = \mathbf{A}\mathbf{y}$ 和 $\mathbf{r}_0 = \text{LowBits}_q(\mathbf{w} + \mathbf{cs}_2, 2\gamma_2)$. 拒绝条件保证 $\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta$. 依 β 的定义有 $\|\mathbf{cs}_2\|_\infty \leq \beta$, 故加入 $\mathbf{cs}_2$ 不会使 $\mathbf{w}$ 的高位产生进位. 因此, $\text{HighBits}_q(\mathbf{w} + \mathbf{cs}_2, 2\gamma_2) = \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$.

验证时, 验证者计算

$$\begin{aligned}
 \mathbf{w}'_{\text{Approx}} &= \mathbf{A}\mathbf{z} - c_p^q \mathbf{b}_1 \cdot 2^d \\
 &= \mathbf{A}(\mathbf{y} + \mathbf{cs}_1) - c_p^q (\mathbf{b} - \mathbf{b}_0) \\
 &= \mathbf{A}\mathbf{y} + \mathbf{cs}_2 + c_p^q \mathbf{b}_0 \\
 &= \mathbf{w} + \mathbf{cs}_2 + c_p^q \mathbf{b}_0,
 \end{aligned}$$

其中使用了 $\mathbf{A}\mathbf{s}_1 - \mathbf{s}_2 = \frac{q}{p}\mathbf{b}$ 以及 Power2Round_p 分解 $\mathbf{b} = \mathbf{b}_1 \cdot 2^d + \mathbf{b}_0$. 提示 $\mathbf{h} = \text{MakeHint}_q(-c_p^q \mathbf{b}_0, \mathbf{w} + \mathbf{cs}_2 + c_p^q \mathbf{b}_0, 2\gamma_2)$ 编码了 $c_p^q \mathbf{b}_0$ 影响高位时的进位位置. 因此, $\text{UseHint}_q(\mathbf{h}, \mathbf{w}'_{\text{Approx}}, 2\gamma_2) = \text{HighBits}_q(\mathbf{w} + \mathbf{cs}_2, 2\gamma_2) = \mathbf{w}_1$, 验证者可正确恢复 $\mathbf{w}_1$, 并将其摘要与签名中收到的摘要比较.

为确保不接受恶意签名, 验证者必须检查 $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$. 允许更大的 $\mathbf{z}$ 会降低伪造难度, 从而影响目标安全级别. 若 sigDecode 强制提示采用至多含 ω 个索引的规范编码, 则显式检查 $\text{HW}(\mathbf{h}) > \omega$ 是冗余的; 为表述清晰, 仍在 Verify 中保留该检查.

Algorithm 3 玄色 (Octarine) 的 KeyGen、Sign 和 Verify 算法.

$(pk, sk) \leftarrow \text{玄色 (Octarine) .KeyGen}()$

- 1: $\xi \leftarrow U(\mathbb{B}^{64})$ ▷ Sample initial seed
- 2: $(\rho, \rho', K) \in \mathbb{B}^{64} \times \mathbb{B}^{128} \times \mathbb{B}^{64} \leftarrow H_{\text{kdf}}(\xi; |\rho| + |\rho'| + |K|)$
- 3: $\mathbf{A} \leftarrow D_{\mathbf{A}}(\mathbf{S}_{q,n,k}; \rho)$
- 4: $\mathbf{s}_1 \leftarrow D_{\mathbf{s}}(\mathbf{S}_{q,n,k}; \rho')$
- 5: $\mathbf{t} = \mathbf{A}\mathbf{s}_1 \in \mathbf{S}_{q,n,k}$
- 6: $\mathbf{b} = \lfloor \frac{p}{q} \mathbf{t} \rfloor \in \mathbf{S}_{p,n,k}$ ▷ Rounded public value in $\mathbf{S}_{p,n,k}$
- 7: $\mathbf{s}_2 = \mathbf{t} - \frac{q}{p} \mathbf{b} \in \mathbf{S}_{q/p,n,k}$ ▷ $\frac{q}{p} \mathbf{b} = \mathbf{A}\mathbf{s}_1 - \mathbf{s}_2$, with derived $\mathbf{s}_2$ bounded in $[-\frac{q}{2p}, \frac{q}{2p} - 1]$
- 8: $(\mathbf{b}_1, \mathbf{b}_0) \leftarrow \text{Power2Round}_p(\mathbf{b}, 2^d)$ ▷ Compress $\mathbf{b} \in \mathbf{S}_p$
- 9: $pk \leftarrow \text{pkEncode}(\rho, \mathbf{b}_1)$
- 10: $tr \leftarrow H_{\text{tr}}(pk; |tr|)$
- 11: $sk \leftarrow \text{skEncode}(\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{b}_0)$ ▷ K and tr are for use in signing
- 12: **return** (pk, sk)

$\sigma \leftarrow \text{玄色 (Octarine) .Sign}(sk, M, rnd)$

- 1: $(\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{b}_0) \leftarrow \text{skDecode}(sk)$
- 2: $\mathbf{A} \leftarrow D_{\mathbf{A}}(\mathbf{S}_{q,n,k}; \rho)$
- 3: $\mu \leftarrow H_{\text{mu}}(tr, M; |\mu|)$
- 4: $\rho'' \leftarrow H_{\text{rhopp}}(K, rnd, \mu; |\rho''|)$
- 5: $\kappa \leftarrow 0$
- 6: $(z, h) \leftarrow \perp$
- 7: **while** $(z, h) = \perp$ **do**
- 8: $\mathbf{y} \leftarrow D_{\mathbf{y}}(\mathbf{S}_{q,n,k}; \rho'', \kappa)$
- 9: $\mathbf{w} = \mathbf{A} \cdot \mathbf{y}$
- 10: $\mathbf{w}_1 = \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$
- 11: $\tilde{c} = H_{\text{ch}}(\mu, \mathbf{w}_1; |\tilde{c}|)$
- 12: $c = \text{SampleInBall}(\tilde{c})$
- 13: $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$
- 14: $\mathbf{r}_0 = \text{LowBits}_q(\mathbf{w} + c\mathbf{s}_2, 2\gamma_2)$
- 15: **if** $\|\mathbf{r}_0\|_{\infty} \geq \gamma_2 - \beta$ **or** $\|\mathbf{z}\|_{\infty} \geq \gamma_1 - \beta$ **then** $(\mathbf{z}, \mathbf{h}) \leftarrow \perp$
- 16: **else**
- 17: $\mathbf{h} = \text{MakeHint}_q(-c\frac{q}{p}\mathbf{b}_0, \mathbf{w} + c\mathbf{s}_2 + c\frac{q}{p}\mathbf{b}_0, 2\gamma_2)$
- 18: **if** $\|c\frac{q}{p}\mathbf{b}_0\|_{\infty} \geq \gamma_2$ **or** $\text{HW}(\mathbf{h}) > \omega$ **then** $(\mathbf{z}, \mathbf{h}) \leftarrow \perp$
- 19: **end if**
- 20: **end if**
- 21: $\kappa = \kappa + k$
- 22: **end while**
- 23: **return** $\sigma = \text{sigEncode}(\tilde{c}, \mathbf{z}, \mathbf{h})$

$\text{bool} \leftarrow \text{玄色 (Octarine) .Verify}(pk, M, \sigma)$

- 1: $(\rho, \mathbf{b}_1) \leftarrow \text{pkDecode}(pk)$
 - 2: $(\tilde{c}, \mathbf{z}, \mathbf{h}) \leftarrow \text{sigDecode}(\sigma)$
 - 3: $\mathbf{A} \leftarrow D_{\mathbf{A}}(\mathbf{S}_{q,n,k}; \rho)$
 - 4: $tr \leftarrow H_{\text{tr}}(pk; |tr|)$
 - 5: $\mu \leftarrow H_{\text{mu}}(tr, M; |\mu|)$
 - 6: $c \leftarrow \text{SampleInBall}(\tilde{c})$
 - 7: $\mathbf{w}'_{\text{Approx}} \leftarrow \mathbf{A}\mathbf{z} - c\frac{q}{p}\mathbf{b}_1 2^d$
 - 8: $\mathbf{w}'_1 \leftarrow \text{UseHint}_q(\mathbf{h}, \mathbf{w}'_{\text{Approx}}, 2\gamma_2)$
 - 9: $\tilde{c}' \leftarrow H_{\text{ch}}(\mu, \mathbf{w}'_1; |\tilde{c}'|)$
 - 10: **return** $[[\|\mathbf{z}\|_{\infty} < \gamma_1 - \beta]]$ **and** $[[\tilde{c} = \tilde{c}']]$ **and** $[[\text{HW}(\mathbf{h}) \leq \omega]]$
-

5 设计原理

SIS 型假设. 方案的不可伪造性还要求 $\mathbf{S}_{q,n,k}$ 上的模 SIS 型 (Module-SIS-type) 假设. 具体而言, 需要假设寻找短非零向量 $\mathbf{z} \in \mathbf{S}_{q,n,k}^\ell$, 使其在 $\mathbf{S}_{q,n,k}$ 中满足 $\mathbf{A}\mathbf{z} = \mathbf{0}$, 是困难的. RR-LWR 与 SIS 型假设均可通过标准格规约攻击分析 (见第 6 节).

5.1 参数选择准则

方案参数为 $q, p, n, k, \gamma_1, \gamma_2, \beta$ 和 ω , 其作用如下:

- q : 主模数 (二的幂), 控制环元素大小及与安全性相关的格维数.
- p : LWR 舍入模数 (二的幂且 $p < q$), 决定 $\mathbf{b}$ 的公钥压缩.
- n, k : n 是基环次数, k 是根式扩张次数. 二者共同确定总代数次数 nk , 也就是通用攻击估计采用的有效格维数. 增大 k 会提高格维数和安全级别, 但也会增大密钥和签名.
- γ_1 : 掩蔽范围; 既要足够大, 使拒绝采样不泄露 $\mathbf{s}_1$ 的信息, 又要足够小, 使签名大小可控.
- γ_2 : 低位舍入范围 (二的幂), 决定 HighBits 分解粒度及提示比特数. 较小的 γ_2 可缩短提示, 但会提高拒绝概率.
- β : 拒绝界, 设为 $\beta = \tau \cdot \eta$, 其中 τ 是挑战多项式的汉明重量, η 是 $\mathbf{s}_1$ 的系数界.
- ω : 提示向量 $\mathbf{h}$ 的最大汉明重量.

参数选择需同时满足:

1. 正确性: 拒绝条件 $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ 和 $\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta$ 保证所有有效签名均能通过验证.
2. 安全性: 对选定的格维数和模数, RR-LWR 与 SIS 型问题在目标安全级别下必须保持困难. 具体估计见第 6 节.
3. 效率: 签名的预期重复次数 (拒绝率) 应较小, 所得密钥和签名也应紧凑.

5.2 正确性与拒绝概率分析

带中止 Fiat-Shamir 范式保证完美正确性 (*perfect correctness*): Sign 算法中通过所有拒绝条件的每个签名, 均以概率 1 被 Verify 算法接受. 换言之, 不存在类似“解密失败”的事件.

引理 1 (高位稳定性). 设 $m = 2\gamma_2$ 整除 q , 且 HighBits_q 和 LowBits_q 按系数定义, 低位表示于 $[-\gamma_2, \gamma_2 - 1]$. 对环元素 $a, e \in \mathbf{S}_{q,n,k}$, 若

$$r_0 = \text{LowBits}_q(a + e, m), \quad \|r_0\|_\infty < \gamma_2 - \beta, \quad \|e\|_\infty \leq \beta.$$

则

$$\text{HighBits}_q(a + e, m) = \text{HighBits}_q(a, m).$$

Proof. 只需逐系数论证. 写作

$$a + e = hm + r_0 \pmod{q}$$

其中 $r_0 \in [-\gamma_2, \gamma_2 - 1]$. 由假设可知 $r_0 - e$ 仍处于同一低位区间. 因此

$$a = hm + (r_0 - e) \pmod{q}$$

与 $a + e$ 具有相同的高位分量 h . □

引理 2 (提示正确性). 设 $m = 2\gamma_2$ 整除 q , MakeHint_q 和 UseHint_q 是算法 1 中的逐系数算法. 对环元素 $r, e \in \mathbf{S}_{q,n,k}$, 假设 $\|e\|_\infty < \gamma_2$. 若

$$\mathbf{h} = \text{MakeHint}_q(e, r, m),$$

则

$$\text{UseHint}_q(\mathbf{h}, r, m) = \text{HighBits}_q(r + e, m).$$

Proof. 仍逐系数论证. 令 $r = r_1m + r_0 \pmod{q}$, 其中 $r_0 \in [-\gamma_2, \gamma_2 - 1]$. 若 $\text{HighBits}_q(r, m) = \text{HighBits}_q(r + e, m)$, 则 MakeHint_q 返回 0, UseHint_q 返回 r_1 . 否则, 因 $|e| < \gamma_2 = m/2$, 扰动至多跨越一个分解边界. 若 $r_0 > 0$, 只能向上跨越, UseHint_q 返回 $r_1 + 1$; 若 $r_0 \leq 0$, 只能向下跨越, 返回 $r_1 - 1$. 模 q/m 意义下, 这些值恰为 $r + e$ 的高位. $\square$

签名循环中的四个拒绝条件保证了这一点:

1. 条件 $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ 保证响应向量 $\mathbf{z}$ 保持在验证者接受的范围.
2. 条件 $\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta$, 其中 $\mathbf{r}_0 = \text{LowBits}_q(\mathbf{w} + c\mathbf{s}_2, 2\gamma_2)$, 保证低位足够小, 从而由引理 1 有 $\text{HighBits}_q(\mathbf{w} + c\mathbf{s}_2, 2\gamma_2) = \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$.
3. 条件 $\|c_p^q \mathbf{b}_0\|_\infty < \gamma_2$ 保证公钥压缩扰动足够小, 使逐系数的 $\text{MakeHint}_q / \text{UseHint}_q$ 进位校正有效. 在诚实签名等式中, 应用引理 2, 代入 $r = \mathbf{w} + c\mathbf{s}_2 + c_p^q \mathbf{b}_0$ 且 $e = -c_p^q \mathbf{b}_0$.
4. 提示重量条件 $\text{HW}(\mathbf{h}) \leq \omega$ 限制进位校正次数, 保证提示可在签名为其分配的空间内表示.

四个条件全部满足时, 第 4 节的正确性论证表明验证者可精确恢复 $\mathbf{w}_1$.

这种方法的代价是, 产生有效签名前可能需要多轮循环. 预期重复次数取决于掩蔽范围 γ_1 与拒绝界 β 的比值以及舍入参数 γ_2 . 参数经过选择, 使所有安全级别的预期重复次数均较小 (低于 4); 具体数值见表 2.

5.3 其他考虑

二的幂模数 q . 选择二的幂模数 q (而非 Dilithium 中的素数模数) 具有以下优点:

- 舍入更简单: 所有舍入操作 (Power2Round、HighBits、LowBits) 均化为移位操作, 无需为这些步骤执行模约减例程.
- 与 KEM 兼容: KEM 使用相同的二的幂模数 q , 故两个方案可以共享 NTT 例程、CRT 表示和采样基础设施.
- 侧信道与故障防护: 涉及二的幂的操作天然无分支且时序一致, 有利于常数时间实现; 二的幂结构也使某些故障注入攻击场景更易分析.
- 安全性论证更简单: 当 q 为二的幂时, 舍入就是直接截断比特, 故 RR-LWR 假设无需 q 为素数时所需的“模数切换 (modulus switching)”技术论证 [BPR12].

挑战空间与熵. 挑战被采样为稀疏多项式 $c \in \mathbf{R}_{q,n}$, 恰含 τ 个属于 $\{-1, +1\}$ 的非零系数, 并作为常数 x^0 分量嵌入 $\mathbf{S}_{q,n,k}$. 因此, 对 $\mathbf{s} = \sum_{i=0}^{k-1} s_i x^i$, 乘以 c 按分量进行:

$$c\mathbf{s} = \sum_{i=0}^{k-1} (cs_i) x^i.$$

在所有此类带符号、汉明重量为 τ 的挑战上的均匀分布，其挑战熵为 $\log_2 \binom{n}{\tau} + \tau$ 比特. 算法 2 的 SampleInBall 过程通过对 $n = 1024$ 的 10 比特洗牌索引执行拒绝采样来实现该分布.

6 安全性声明与分析

6.1 理论安全性

6.1.1 RR-LWR 困难问题

设 n 为二的幂, k 为正整数. 可以证明多项式 $f(x) = (x^k - 2)^n + 1$ 不可约 [BRVW26, Lemma 1], 因此 $\mathbf{Q}[x]/(f(x))$ 是数域, 可在该数域上定义 Ring-LWR 问题. 这是标准 LWR [BPR12] 的数域对应形式, 并以 玄色 (Octarine) 使用的根式环表示实例化.

由多项式 $f(x) = (x^k - 2)^n + 1$, 在数域 $\mathbf{K} = \mathbf{Q}[x]/(f(x)) = \mathbf{Q}[\alpha]$ 的整数环 $\mathcal{O}_{\mathbf{K}}$ 上定义 RR-LWR 问题. 与 Ring-LWE 不同, LWR 样本不含独立采样的高斯误差项; 其噪声是下述逐系数映射引入的确定性量化误差:

$$z \in \mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}} \mapsto \left\lfloor \frac{p}{q} z \right\rfloor \in \mathcal{O}_{\mathbf{K}}/p\mathcal{O}_{\mathbf{K}}.$$

通过显式计算 (例如使用 Dedekind 判别准则) 可验证 $\mathcal{O}_{\mathbf{K}} = \mathbf{Z}[\alpha]$. 因此, 实现环

$$S_{q,n,k} = (\mathbf{Z}_q[y]/(y^n + 1)[x]) / (x^k - y - 2)$$

只是 $\mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}}$ 的一种具体系数表示.

定义 2 (搜索型 RR-LWR). 设 $q > p \geq 2$, 令

$$S_q = S_{q,n,k} = (\mathbf{Z}_q[y]/(y^n + 1)[x]) / (x^k - y - 2), \quad S_p = S_{p,n,k},$$

并令 D_s 为 S_q 上的秘密分布. 对固定秘密 $\mathbf{s} \in S_q$, 定义 $S_q \times S_p$ 上的 RR-LWR 样本分布 $\mathcal{L}_{s,q,p}^{\text{rr-lwr}}$ 如下: 均匀采样 $\mathbf{A} \leftarrow S_q$, 并输出

$$\left(\mathbf{A}, \left\lfloor \frac{p}{q} \mathbf{A} \mathbf{s} \right\rfloor \right),$$

其中舍入在 $y^i x^j$ 基下逐系数执行. 搜索型 RR-LWR 问题为: 对 $\mathbf{s} \leftarrow D_s$, 给定来自 $\mathcal{L}_{s,q,p}^{\text{rr-lwr}}$ 的 N_{samp} 个独立样本, 恢复 $\mathbf{s}$. 算法 $\mathcal{A}$ 的优势为

$$\text{Adv}_{q,p,D_s,N_{\text{samp}}}^{\text{s-rr-lwr}}(\mathcal{A}) = \Pr[\mathcal{A}((\mathbf{A}_i, \mathbf{b}_i)_{i=1}^{N_{\text{samp}}}) = \mathbf{s}].$$

定义 3 (判定型 RR-LWR). 沿用定义 2 的记号, 判定型 RR-LWR 问题要求区分以下两个实验:

Real: $\mathbf{s} \leftarrow D_s$, $(\mathbf{A}_i, \mathbf{b}_i) \leftarrow \mathcal{L}_{s,q,p}^{\text{rr-lwr}}$ 其中 $i = 1, \dots, N_{\text{samp}}$,

Rand: $\mathbf{A}_i \leftarrow S_q$, $\mathbf{b}_i \leftarrow S_p$ 对 $i = 1, \dots, N_{\text{samp}}$ 独立均匀采样.

区分器 $\mathcal{A}$ 的优势为

$$\text{Adv}_{q,p,D_s,N_{\text{samp}}}^{\text{d-rr-lwr}}(\mathcal{A}) = |\Pr[\mathcal{A}(\text{Real}) = 1] - \Pr[\mathcal{A}(\text{Rand}) = 1]|.$$

判定型 RR-LWR 假设断言, 对所有高效区分器 $\mathcal{A}$, 该优势均可忽略.

定义 4 (判定型密钥生成 RR-LWR). 沿用定义 2 的记号, 密钥生成判定型 RR-LWR 问题是算法 3 中签名密钥生成算法扩展公钥视图的单样本判定问题, 其中带种子的公开矩阵已替换为显式均匀元素. 挑战者输出 $S_q \times S_p$ 中的一对元素. 真实实验中的元素对为

$$\text{KGReal}_{q,p,D_s} = \left(\mathbf{A}, \left\lfloor \frac{p}{q} \mathbf{A} \mathbf{s} \right\rfloor \right), \quad \mathbf{A} \leftarrow S_q, \quad \mathbf{s} \leftarrow D_s,$$

随机分布则为

$$\text{KGRand}_{q,p} = (\mathbf{A}, \mathbf{u}), \quad \mathbf{A} \leftarrow S_q, \quad \mathbf{u} \leftarrow S_p,$$

其中 $\mathbf{A}$ 和 $\mathbf{u}$ 独立均匀采样. 区分器只接收所示元素对, 尤其不会获得秘密 $\mathbf{s}$. 区分器 $\mathcal{A}$ 的优势为

$$\text{Adv}_{\text{kg},q,p,D_s}^{\text{d-rr-lwr}}(\mathcal{A}) = |\Pr[\mathcal{A}(\text{KGReal}_{q,p,D_s}) = 1] - \Pr[\mathcal{A}(\text{KGRand}_{q,p}) = 1]|.$$

下述定理形式化陈述表示规约.

定理 3 (根式数域上作为基变换 Ring-LWR 的 RR-LWR). 设 $q > p \geq 2$ 且 $p \mid q$, $f(x) = (x^k - 2)^n + 1$ 不可约, 并令 $\mathbf{K} = \mathbf{Q}[x]/(f(x)) = \mathbf{Q}[\alpha]$, $\mathcal{O}_{\mathbf{K}} = \mathbf{Z}[\alpha]$. 令

$$\varphi_q : S_{q,n,k} \longrightarrow \mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}}$$

为由 $x \mapsto \alpha$, $y \mapsto \alpha^k - 2$ 诱导的环同构, 并在模 p 下类似定义 φ_p . 令 $\mathcal{B} = \{(\alpha^k - 2)^i \alpha^j : 0 \leq i < n, 0 \leq j < k\}$ 为 $\mathcal{O}_{\mathbf{K}}$ 的 $\mathbf{Z}$ -基, 即单项式基 $\{y^i x^j\}$ 在 φ_q 下的像; 令 $[\cdot]_{\mathcal{B}}$ 表示在基 $\mathcal{B}$ 下逐系数将 $\mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}}$ 舍入至 $\mathcal{O}_{\mathbf{K}}/p\mathcal{O}_{\mathbf{K}}$. 对于 $S_{q,n,k}$ 上任意秘密分布 D_s , 任何区分下述 RR-LWR 样本

$$\left(\mathbf{A}, \left\lfloor \frac{p}{q} \mathbf{A} \mathbf{s} \right\rfloor \right) \in S_{q,n,k} \times S_{p,n,k}, \quad \mathbf{A} \leftarrow S_{q,n,k}, \quad \mathbf{s} \leftarrow D_s,$$

与均匀样本的区分器 $\mathcal{A}$, 均可导出 $\mathcal{O}_{\mathbf{K}}$ 上相应判定型 Ring-LWR 问题的区分器 $\mathcal{B}'$; 该问题采用舍入 $[\cdot]_{\mathcal{B}}$ 和秘密分布 $\varphi_q(D_s)$, 样本数相同、运行时间基本相同且优势相同.

Proof. 由于 $S_{q,n,k} \cong \mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}}$ 且 $S_{p,n,k} \cong \mathcal{O}_{\mathbf{K}}/p\mathcal{O}_{\mathbf{K}}$ (后者使用 $\mathcal{O}_{\mathbf{K}} = \mathbf{Z}[\alpha]$), 映射 φ_q 和 φ_p 是可高效计算的双射. RR-LWR 的舍入在基 $\{y^i x^j\}$ 下逐系数执行, 而 φ_q 将该基映至 $\mathcal{B}$; 故根据 $[\cdot]_{\mathcal{B}}$ 的定义, 对每个 $z \in S_{q,n,k}$, 有

$$\varphi_p \left(\left\lfloor \frac{p}{q} z \right\rfloor \right) = \left\lfloor \frac{p}{q} \varphi_q(z) \right\rfloor_{\mathcal{B}}.$$

规约 $\mathcal{B}'$ 接收 Ring-LWR 挑战样本 $(\tilde{\mathbf{A}}_i, \tilde{\mathbf{b}}_i) \in (\mathcal{O}_{\mathbf{K}}/q\mathcal{O}_{\mathbf{K}}) \times (\mathcal{O}_{\mathbf{K}}/p\mathcal{O}_{\mathbf{K}})$, 并将其映射为

$$(\mathbf{A}_i, \mathbf{b}_i) = (\varphi_q^{-1}(\tilde{\mathbf{A}}_i), \varphi_p^{-1}(\tilde{\mathbf{b}}_i)).$$

若挑战样本是秘密为 $\tilde{\mathbf{s}} = \varphi_q(\mathbf{s})$ 的 Ring-LWR 样本, 则映射后的元素对恰为秘密为 $\mathbf{s}$ 的 RR-LWR 样本; 若挑战均匀, 则因 φ_q, φ_p 均为双射, 映射后的元素对也均匀. 因此 $\mathcal{B}'$ 与 $\mathcal{A}$ 具有相同优势. $\square$

需要强调, 定理 3 将 RR-LWR 规约到一种 Ring-LWR 变体, 其舍入在变换后的单项式基 $\mathcal{B}$ 下逐系数执行, 而非在典范嵌入中执行. 在具体估计中, 我们通过下述线性换基 L 计入相应的基失真: 其奇异值 $\sigma_{\min}(L) = \sqrt{nk}$ 和 $\sigma_{\max}(L) \leq C_{\text{emb}} \sqrt{nk}$ (其中较小常数 (下表列出较小常数 C_{emb}) 表明, 在相关秘密/误差范数中, 典范嵌入范数估计至多应计入因子 C_{emb} 的失真).

由 f 可容易地将 nk 个根显式写为

$$r_{i,j} = (\xi_{2n}^{(2j+1)} + 2)^{(1/k)} \xi_k^i$$

其中 $i = 0, \dots, k-1, j = 0, \dots, n-1, \xi_m$ 是本原 m 次单位根. 对这些根排序, 使 $m = 1, \dots, nk/2$ 时 $r_{m+nk/2} = \bar{r}_m$. 于是元素 $a(x) \in \mathbf{K}$ 的典范嵌入由 $i = 1, \dots, nk$ 的 nk 个取值 $a(r_i)$ 构成.

为比较 RR-LWR 实例与通常的典范嵌入观点, 还需计入如下差异: 在环表示 $(\mathbf{Z}_q[y]/(y^n + 1)[x])/(x^k - y - 2)$ 中, 按基 $y^i x^j$ ($i = 0, \dots, n-1, j = 0, \dots, k-1$) 逐系数采样, 与在 $\mathbf{K}$ 的典范嵌入中采样并不相同. 该差异对应线性变换 L , 它由两个线性变换复合而成:

- 第一个变换将基 $y^i x^j$ 映射为相应元素 $b(x) = (x^k - 2)^i x^j \bmod f(x)$ (回顾本环中确有 $y = x^k - 2$);
- 第二个变换是标准典范嵌入, 将 $b(x)$ 映射为 $[b(r_i)]_{i=1, \dots, nk}$. 该变换即 Vandermonde 矩阵, 其第 i 行由 $j = 0, \dots, nk-1$ 时 r_i^j 的各次幂组成.

为限制在基 $y^i x^j$ 而非典范基中采样所引入的最大失真, 只需考察上述变换 L 的最小与最大奇异值. 在二次幂分圆多项式 $x^n + 1$ 的“理想情形”中, 二者相同, 均为 $\sqrt{n}$. 我们计算了 $k = 2, \dots, 15$ 及不同 n 下变换 L 的奇异值, 发现最小奇异值为 $\sqrt{nk}$ (nk 是扩张次数, 故为最优), 最大奇异值为依赖于 k 的较小倍数 $C_{\text{emb}} \sqrt{nk}$: 对 $k = 2$, $C_{\text{emb}} < 1.74$; 对 $k = 3$, $C_{\text{emb}} < 2.08$; 对 $k = 5$, $C_{\text{emb}} < 2.41$, 对 $k = 7$, $C_{\text{emb}} < 2.57$; 对 $k = 15$, $C_{\text{emb}} < 2.79$. 这表明, 与在典范基中采样相比, 在系数基中采样只引入很小失真. 因此, 玄色 (Octarine) 所用变换单项式基下表示的秘密分布, 可在仅计入这一有界基失真的情况下, 与次数为 kn 的数域上通常的典范嵌入 Ring-LWR 观点比较. 由此可使用维数 kn 上的通用格攻击估计所得 RR-LWR 实例的困难性.

最后注意, 由于 $f(x)$ 不可约, 该环结构不同于 m-RLWE 文献 [PUTPPG15] 及一系列后续工作 [PUTPPG16, APUPG17, PUTPG+21] 所研究的多元环. 这些工作考察形如 $R_q[x, y] = \mathbf{Z}_q[x, y]/(g(x), h(y))$ 的环, 其中相应多项式是在 $\mathbf{Z}$ 上定义的一元多项式. Bootland、Castricky 和 Vercauteren [BCV20] 破解了某些 g, h 选择下的 m-RLWE 问题, 特别是当二者的根同处于扩张次数小于 $\deg(g)\deg(h)$ 的数域时. 由于式 (5) 的环模数 $f(x) = (x^k - 2)^n + 1$ 确实定义次数为 kn 的数域, 这些攻击并不适用. 虽然我们利用式 (5) 的同构高效运用环结构, 但该格问题是一元环 $S_{q,n,k}$ 上的变换基 Ring-LWR 问题.

上述讨论给出了下文签名证明采用的 RR-LWR 困难性假设. 签名方案只需要密钥生成分布: EUF-CMA 或 SUF-CMA 实验中不存在加密挑战. 证明在扩展公钥视图 $(\mathbf{A}, \mathbf{b}_1)$ 中进行, 其中 $\mathbf{A}$ 显式可得且均匀. 底层 RR-LWR 挑战仍是完整元素对 $(\mathbf{A}, \mathbf{b})$; 规约计算 $(\mathbf{b}_1, \mathbf{b}_0) = \text{Power2Round}_p(\mathbf{b}, 2^d)$, 仅将 $\mathbf{b}_1$ 交给攻击者, 并保留 $\mathbf{b}_0$ 用于提取. 实际公钥只存储种子 ρ , 并重新生成 $\mathbf{A} = D_{\mathbf{A}}(\mathbf{S}_{q,n,k}; \rho)$; 以扩展均匀视图替换该带种子采样器的代价至多为采样器区分项 $\Delta_{\mathbf{A}}$. 当 $D_{\mathbf{A}}$ 被建模为理想随机预言机时, 该项为零.

6.1.2 玄色 (Octarine) 的不可伪造性

我们在随机预言机模型中分析该签名方案. 随机预言机的所有输入均使用固定域分离标签和长度定界字符串编码, 使密钥派生、消息代表元、签名随机性及 Fiat-Shamir 挑战的调用处于互不相交的随机预言机域. 攻击者获得公钥以及对签名算法的预言机访问权. 对每次签名查询, 随机化签名预言机在内部采样新的 rnd , 并返回 玄色 (Octarine) .Sign(sk, M, rnd). 固定 rnd 的确定性变体具有不同的重复查询记录分布, 并非下述证明采用的预言机接口. 在 EUF-CMA 实

验中, 若攻击者针对从未提交给签名预言机的消息 M^* 输出有效元素对 (M^*, σ^*) , 则攻击者获胜. 在 SUF-CMA 实验中, 若攻击者输出不同于签名预言机曾返回的每个元素对的有效元素对 (M^*, σ^*) , 则攻击者获胜. 因此 SUF-CMA 还排除了为已签名消息生成第二个有效签名.

令 $S_q = \mathbf{S}_{q,n,k}$, 所有范数均指有符号系数表示下的逐系数无穷范数. 令

$$\mathcal{C}_\tau = \{c \in \mathbf{R}_{q,n} : c \text{ 恰有 } \tau \text{ 个系数属于 } \{-1, +1\}\}$$

按第 4 节所述, 通过常数 x^0 分量将其嵌入 S_q . 对一个密钥对, 记

$$\mathbf{b} = \left\lfloor \frac{p}{q} \mathbf{A} \mathbf{s}_1 \right\rfloor, \quad \mathbf{b} = \mathbf{b}_1 2^d + \mathbf{b}_0, \quad \mathbf{A} \mathbf{s}_1 - \mathbf{s}_2 = \frac{q}{p} \mathbf{b}.$$

RR-MSIS 关系. 不可伪造性所用的 SIS 型假设是模短整数解问题 (module short integer solution problem) 的根式环对应形式, 记为 RR-MSIS. 其齐次形式中, S_q 上的 RR-MSIS 实例给出公开行向量 $\mathbf{a} = (a_1, \dots, a_m) \in S_q^m$, 要求寻找非零短向量 $\mathbf{x} = (x_1, \dots, x_m) \in S_q^m$, 使

$$\sum_{i=1}^m a_i x_i = 0 \pmod{q}.$$

短性在 S_q 的有符号代表元中逐系数度量. 从 Fiat-Shamir 伪造中提取的关系具有相同结构, 但公开行向量和目标被具体化为 玄色 (Octarine) 的验证等式. 在 EUF-CMA 证明中, 目标是自目标式的 (self-targeted): 攻击者的 $\mathbf{w}_1$ 确定等式右侧, 并且必须通过 $\tilde{c} = \mathbf{H}(\mu \parallel \mathbf{w}_1)$ 和 $c = \text{SampleInBall}(\tilde{c})$ 与随机预言机保持一致. 在 SUF-CMA 证明中, 将恢复出相同 $\mathbf{w}_1$ 的两个可接受签名相减会消去该目标, 留下齐次 RR-MSIS 关系.

表 2 的不可伪造性估计采用以下两个界:

$$B_{\text{EUF}} = \max \left\{ \gamma_1 - \beta, 2\gamma_2 + 1 + \frac{q}{p} \tau 2^{d-1} \right\},$$

以及

$$B_{\text{SUF}} = \max \{2(\gamma_1 - \beta), 4\gamma_2 + 2\}.$$

EUF-CMA 提取得到非齐次自目标 RR-MSIS 关系: 给定 $(\mathbf{A}, \mathbf{b})$, 寻找 $(\mu, \mathbf{z}, \tilde{c}, c, \mathbf{r}, \mathbf{w}_1)$, 其中 $\mathbf{z}, \mathbf{r} \in S_q$, $\tilde{c} \in \{0, 1\}^{|\tilde{c}|}$, 且 $c \in \mathcal{C}_\tau$, 使得

$$c = \text{SampleInBall}(\tilde{c}), \quad \tilde{c} = \mathbf{H}(\mu \parallel \mathbf{w}_1),$$

并满足 $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ 、 $\|\mathbf{r}\|_\infty \leq B_{\text{EUF}}$ 以及

$$\mathbf{A} \mathbf{z} - c \frac{q}{p} \mathbf{b} - \mathbf{r} = 2\gamma_2 \mathbf{w}_1 \pmod{q}.$$

短未知量共有 $2kn + n$ 个系数位置: $\mathbf{z}$ 和 $\mathbf{r}$ 各有 kn 个, 常数分量挑战有 n 个. 表 2 保守估计将挑战嵌入完整 S_q 元素后的代价, 得到较大的维数 $3kn$. 按此约定, SUF-CMA 提取得到维数为 $2kn$ 的齐次 RR-MSIS 关系: 给定 $\mathbf{A}$, 寻找非零元素对 $(\Delta \mathbf{z}, \Delta \mathbf{r}) \in S_q^2$, 满足

$$\|\Delta \mathbf{z}\|_\infty < 2(\gamma_1 - \beta), \quad \|\Delta \mathbf{r}\|_\infty \leq 4\gamma_2 + 2, \quad \mathbf{A} \Delta \mathbf{z} - \Delta \mathbf{r} = 0 \pmod{q}.$$

两者均为带中止 Fiat-Shamir 签名安全性分析所用 MSIS 问题的根式环对应形式. 下述规约在

经典随机预言机模型 (random oracle model) 中陈述. QROM 证明还需要 QROM 下带中止 Fiat-Shamir 签名所采用的额外重编程分析; 本文不声称给出此类证明.

规约还将纯计算损失与签名模拟器的统计损失分离. 令 Δ_{sim} 表示在 RR-LWR 混合实验之后的所有公钥及签名查询消息上, 可接受的真实签名记录分布与模拟器输出记录分布之间的最大统计距离. 该项涵盖拒绝采样条件、有界 $\mathbf{y}$ 下 $\mathbf{A}\mathbf{y}$ 高低位的分布, 以及规范提示分布. 令 $2^{-\alpha}$ 表示模拟承诺高位 $\mathbf{w}_1$ 的最大点概率上界. 这些量不能仅由 RR-LWR 或 RR-MSIS 假设消除; 对具体参数集, 必须单独给出界, 或在具体安全性估计中明确作为统计假设处理.

定理 4 (EUF-CMA 安全性). 假设定义 4 的密钥生成 RR-LWR 问题对秘密分布 D_s 是困难的, 上述自目标 RR-MSIS 关系在界 B_{EUF} 下是困难的, H 被建模为随机预言机, 且签名模拟器满足上述统计界 Δ_{sim} 和 $2^{-\alpha}$. 则任何至多进行 Q_H 次随机预言机查询和 Q_S 次签名查询的攻击者, 其优势上界为

$$\begin{aligned} & \text{Adv}_{\text{kg}, q, p, D_s}^{\text{d-rr-lwr}}(\mathcal{B}_1) + \Delta_A + \text{Adv}_{q, B_{\text{EUF}}}^{\text{st-rr-msis}}(\mathcal{B}_2) + (Q_H + 1)2^{-|\tilde{c}|} + (Q_H + Q_S + 1)2^{-|\mu|} \\ & + Q_S(Q_H + Q_S)2^{-\alpha} + Q_S\Delta_{\text{sim}}, \end{aligned}$$

其中 Δ_A 是以显式均匀矩阵替换带种子公开矩阵采样器时的区分优势; 点概率界表示, 对每个固定摘要 μ 和固定值 $\mathbf{u}$, 采样的 $\mathbf{w}_1$ 满足 $\Pr[\mathbf{w}_1 = \mathbf{u}] \leq 2^{-\alpha}$. 项 $(Q_H + Q_S + 1)2^{-|\mu|}$ 仅计入第一阶段消息哈希 $M \mapsto H(\text{tr} \parallel M)$ 的不同输入之间发生的碰撞.

Proof. 公钥包含 $(\rho, \mathbf{b}_1)$, 其中 ρ 扩展为 $\mathbf{A}$, $\mathbf{b}_1$ 由 $\mathbf{b} = \lfloor (p/q)\mathbf{A}\mathbf{s}_1 \rfloor$ 得到. 首先以显式均匀的 $\mathbf{A} \leftarrow S_q$ 替换带种子矩阵 $D_{\mathbf{A}}(\mathbf{S}_{q,n,k}; \rho)$, 代价为 Δ_A . 密钥生成 RR-LWR 跳步作用于完整挑战 $(\mathbf{A}, \mathbf{b}) \in S_q \times S_p$: 规约计算 $(\mathbf{b}_1, \mathbf{b}_0) = \text{Power2Round}_p(\mathbf{b}, 2^d)$, 仅将扩展公钥 $(\mathbf{A}, \mathbf{b}_1)$ 交给攻击者, 并保留 $\mathbf{b}_0$ 用于之后的提取步骤. 根据密钥生成 RR-LWR 假设, 以 S_p 中的均匀元素替换 $\mathbf{b}$, 对攻击者优势的改变量至多为 $\text{Adv}_{\text{kg}, q, p, D_s}^{\text{d-rr-lwr}}$. 完成该跳步后, 规约无需知道私钥. 它使用标准带中止 Fiat-Shamir 记录模拟器回答签名查询. 对查询 M_i , 计算 $\mu_i = H(\text{tr} \parallel M_i)$. 若再次查询同一消息, 模拟器像真实随机预言机一样复用相同 μ_i . 消息摘要坏事件仅指两个不同输入 $\text{tr} \parallel M \neq \text{tr} \parallel M'$ 得到相同摘要. 对当前查询, 模拟器依公开记录分布采样一个可接受验证记录 $(\tilde{c}_i, \mathbf{z}_i, \mathbf{h}_i, \mathbf{w}_{1,i})$, 并将 $H(\mu_i \parallel \mathbf{w}_{1,i})$ 编程为 $\tilde{c}_i$. 若点 $(\mu_i, \mathbf{w}_{1,i})$ 已被随机预言机查询或此前的模拟签名确定 (包括同一消息上更早的签名), 则声明坏事件并重新采样记录. 这些坏事件分别贡献定理中的 $(Q_H + Q_S + 1)2^{-|\mu|}$ 和 $Q_S(Q_H + Q_S)2^{-\alpha}$ 项. 以这些坏事件均未发生为条件, 每次签名查询的记录分布与真实签名之间至多相差 Δ_{sim} . 特别地, 可接受响应通过通常的拒绝采样论证隐藏 $\mathbf{s}_1$: 对每个系数, 条件 $\|\mathbf{z}\|_{\infty} < \gamma_1 - \beta$ 为 $\mathbf{c}\mathbf{s}_1$ 的每个可能系数选择长度相同的区间. 类似的低位和提示分布取决于如下情形中 $\mathbf{A}\mathbf{y}$ 的分布: $\mathbf{y}$ 从有界签名分布采样, 而非模 q 均匀采样. 真实与模拟高位、低位及提示分布之间的任何偏差均计入 Δ_{sim} .

令 (M^*, σ^*) 为模拟实验中的有效伪造, 并记

$$\sigma^* = \text{sigEncode}(\tilde{c}^*, \mathbf{z}^*, \mathbf{h}^*), \quad c^* = \text{SampleInBall}(\tilde{c}^*), \quad \mu^* = H(\text{tr} \parallel M^*).$$

验证计算

$$\mathbf{v}^* = \mathbf{A}\mathbf{z}^* - c^* \frac{q}{p} \mathbf{b}_1 2^d$$

并由 $\text{UseHint}_q(\mathbf{h}^*, \mathbf{v}^*, 2\gamma_2)$ 恢复值 $\mathbf{w}_1^*$. 由于签名通过验证, 点 $(\mu^*, \mathbf{w}_1^*)$ 处的随机预言机值为 $\tilde{c}^*$. 若输出伪造前从未查询该预言机输入, 则攻击者猜中了一个独立的 $|\tilde{c}|$ 比特随机预言机值. 定理对这一最终挑战猜测事件采用保守界 $(Q_H + 1)2^{-|\tilde{c}|}$.

还需处理查询 $(\mu^*, \mathbf{w}_1^*)$ 出现在随机预言机记录中的情形. 根据 UseHint_q 的定义, 对每个可接受签名, 均存在元素 $\mathbf{e}_h^* \in S_q$, 满足 $\|\mathbf{e}_h^*\|_\infty \leq 2\gamma_2 + 1$, 且

$$\mathbf{A}\mathbf{z}^* - c^* \frac{q}{p} \mathbf{b}_1 2^d = 2\gamma_2 \mathbf{w}_1^* + \mathbf{e}_h^* \pmod{q}.$$

利用 $\mathbf{b} = \mathbf{b}_1 2^d + \mathbf{b}_0$, 上式变为

$$\mathbf{A}\mathbf{z}^* - c^* \frac{q}{p} \mathbf{b} = 2\gamma_2 \mathbf{w}_1^* + \mathbf{r}^* \pmod{q},$$

其中

$$\mathbf{r}^* = \mathbf{e}_h^* - c^* \frac{q}{p} \mathbf{b}_0.$$

由于 c^* 的汉明重量为 τ , 且 $\mathbf{b}_0$ 的每个系数均位于 $[-2^{d-1}, 2^{d-1} - 1]$, 有

$$\left\| c^* \frac{q}{p} \mathbf{b}_0 \right\|_\infty \leq \frac{q}{p} \tau 2^{d-1}.$$

因此

$$\|\mathbf{r}^*\|_\infty \leq 2\gamma_2 + 1 + \frac{q}{p} \tau 2^{d-1}, \quad \|\mathbf{z}^*\|_\infty < \gamma_1 - \beta.$$

因此, 元素组 $(\mu^*, \mathbf{z}^*, \tilde{c}^*, c^*, \mathbf{r}^*, \mathbf{w}_1^*)$ 是自目标 RR-MSIS 关系的有效解, 因为 $\tilde{c}^* = \mathbf{H}(\mu^* \parallel \mathbf{w}_1^*)$ 且 $c^* = \text{SampleInBall}(\tilde{c}^*)$. 由于 $c^* \in \mathcal{C}_\tau$ 非零, 所提取关系是非平凡的. 因此, 除去前述随机预言机猜测、消息摘要碰撞、编程和模拟损失, 任何成功概率不可忽略的 EUF-CMA 攻击者均可导出该自目标 RR-MSIS 问题的求解器. $\square$

定理 5 (SUF-CMA 安全性). 在定理 4 的假设下, 进一步假设上述齐次 RR-MSIS 关系在界 B_{SUF} 下是困难的, 则 玄色 (Octarine) 具有 SUF-CMA 安全性. 更准确地说, 定理 4 的界需增加

$$\text{Adv}_{q, B_{\text{SUF}}}^{\text{rr-msis}}(\mathcal{B}_3),$$

, 对应界为 B_{SUF} 的齐次 RR-MSIS 问题.

Proof. 对从未签名的消息所作的每个 SUF-CMA 伪造本身就是 EUF-CMA 伪造, 故定理 4 适用. 只需考虑消息 M 上一个新的有效签名, 而签名预言机此前已对该消息返回一个或多个有效签名 $\sigma^{(j)} = \text{sigEncode}(\tilde{c}^{(j)}, \mathbf{z}^{(j)}, \mathbf{h}^{(j)})$. 令伪造签名为

$$\sigma^* = \text{sigEncode}(\tilde{c}^*, \mathbf{z}^*, \mathbf{h}^*).$$

令 $\mu = \mathbf{H}(\text{tr} \parallel M)$, 并令 $\mathbf{w}_1^*$ 为验证 σ^* 时恢复的高位. 若 $\mathbf{w}_1^*$ 不同于此前为 M 返回的所有签名所恢复的高位值 $\mathbf{w}_1^{(j)}$, 则相对于已编程签名点, $(\mu, \mathbf{w}_1^*)$ 是新的. 因此定理 4 的证明会给出自目标 RR-MSIS 解或随机预言机猜测事件.

因此只需考虑 $\mathbf{w}_1^* = \mathbf{w}_1^{(j)}$ 的情形, 其中 $\sigma^{(j)}$ 是此前返回的某个签名. 由于随机预言机是一致的,

$$\tilde{c}^* = \mathbf{H}(\mu \parallel \mathbf{w}_1^*) = \mathbf{H}(\mu \parallel \mathbf{w}_1^{(j)}) = \tilde{c}^{(j)}.$$

故还有 $c^* = c^{(j)} = \text{SampleInBall}(\tilde{c}^*)$.

对 $\sigma^{(j)}$ 与伪造签名, 验证等式给出满足下式的元素 $\mathbf{e}_h^{(j)}, \mathbf{e}_h^* \in S_q$:

$$\mathbf{A}\mathbf{z}^{(j)} - c^* \frac{q}{p} \mathbf{b}_1 2^d = 2\gamma_2 \mathbf{w}_1^* + \mathbf{e}_h^{(j)} \pmod{q}$$

以及

$$\mathbf{A}\mathbf{z}^* - c^* \frac{q}{p} \mathbf{b}_1 2^d = 2\gamma_2 \mathbf{w}_1^* + \mathbf{e}_h^* \pmod{q},$$

其中 $\|\mathbf{e}_h^{(j)}\|_\infty, \|\mathbf{e}_h^*\|_\infty \leq 2\gamma_2 + 1$. 两式相减会消去公钥项和目标:

$$\mathbf{A}(\mathbf{z}^* - \mathbf{z}^{(j)}) - (\mathbf{e}_h^* - \mathbf{e}_h^{(j)}) = 0 \pmod{q}.$$

此外,

$$\|\mathbf{z}^* - \mathbf{z}^{(j)}\|_\infty < 2(\gamma_1 - \beta), \quad \|\mathbf{e}_h^* - \mathbf{e}_h^{(j)}\|_\infty \leq 4\gamma_2 + 2.$$

因此

$$(\Delta \mathbf{z}, \Delta \mathbf{r}) = (\mathbf{z}^* - \mathbf{z}^{(j)}, \mathbf{e}_h^* - \mathbf{e}_h^{(j)})$$

是界为 B_{SUF} 的齐次 RR-MSIS 关系的解.

最后只需证明该解非零. 若 $\Delta \mathbf{z} = 0$, 则两次验证使用相同的值 $\mathbf{v} = \mathbf{A}\mathbf{z}^{(j)} - c^*(q/p)\mathbf{b}_1 2^d$. 对固定的 $\mathbf{v}$ 和所恢复的固定高位 $\mathbf{w}_1^*$, UseHint_q 接受的提示向量按系数唯一确定. 签名编码是规范的: 未使用的提示索引必须为零, 每个多项式内的索引严格递增, 解码器拒绝非规范编码. 因此, 相同的 $\tilde{c}^*$ 、 $\mathbf{z}^* = \mathbf{z}^{(j)}$ 和 $\mathbf{w}_1^*$ 强制得到相同的 $\mathbf{h}$, 进而得到相同签名字节. 这与 SUF-CMA 实验中的新鲜性矛盾. 故提取出的 RR-MSIS 解非零. $\square$

6.1.3 超越不可伪造性的性质

玄色 (Octarine) 签名携带的挑战种子也提供了 Fiat-Shamir 签名 BUFF 分析所用的绑定机制 [CDF⁺20]. BUFF 安全性指超越普通不可伪造性的性质: 恶意强通用排他所有权 (malicious-strong universal exclusive ownership, M-S-UEO), 即同一签名不应在攻击者选择的两个公钥下通过验证 (消息也可能不同); 消息绑定签名 (message-bound signatures, MBS), 即固定公钥下的同一签名不应应对两条不同消息通过验证; 以及不可重签名性 (non-resignability), 即未知消息上的签名不应允许攻击者在另一公钥下为同一未知消息产生相关签名.

后续研究表明, 原始强 NR 表述及 BUFF 变换的相应证明策略通常过强, 并提出了可实现的弱化表述 [DFHS24]. 目前关于 BUFF 结构的肯定结果, 是在理想化模型中对高最小熵消息分布证明弱 NR [DFH⁺24]. 因此, 下述结论在随机预言机模型中声称普通 BUFF 所有权性质, 而只在该理想化高最小熵环境中声称相应弱 NR 性质. 显式规范解码条件在恶意密钥环境中尤为重要: BUFF 所有权性质恰可能在弱公钥或畸形公钥处失效 [DS24].

令

$$H_{\text{BUFF}}(pk, M, \mathbf{w}_1) = H_{\text{ch}}(H_{\text{mu}}(H_{\text{tr}}(pk; |tr|), M; |\mu|), \mathbf{w}_1; |\tilde{c}|)$$

表示复合挑战哈希, 采用 $\tilde{c}$ 所用的固定输出长度. H_{tr} 、 H_{mu} 和 H_{ch} 均使用第 4 节规定的域分离、长度定界编码. 实际算法计算 $tr = H_{\text{tr}}(pk; |tr|)$, $\mu = H_{\text{mu}}(tr, M; |\mu|)$, 以及 $\tilde{c} = H_{\text{BUFF}}(pk, M, \mathbf{w}_1)$; 值 $\tilde{c}$ 包含在签名中, 验证会显式检查由恢复的 $\mathbf{w}_1$ 得到相同摘要. 因此 BUFF 摘要已存在于 Fiat-Shamir 挑战种子中, 无需通用变换或额外签名分量.

定理 6 (BUFF 性质). 假设公钥解码仅接受规范字节编码, 且对每个被接受的公钥字节串, $(pk, M, \mathbf{w}_1)$ 的域分离、长度定界编码均为单射. 在随机预言机模型中, 任何至多进行 Q_H 次

随机预言机查询的攻击者，其 M - S - UEO 和 MBS 优势上界为

$$(Q_H + 6)^2 \left(2^{-|tr|} + 2^{-|\mu|} + 2^{-|\tilde{c}|} \right),$$

当中间哈希输出至少为 $|\tilde{c}|$ 比特时，该值至多为 $3(Q_H + 6)^2 2^{-|\tilde{c}|}$ 。特别地， M - S - UEO 覆盖的排他所有权性质族也覆盖 [\[CDF⁺20\]](#) 的 S - CEO 和 S - DEO 概念。在 [\[DFH⁺24\]](#) 的理想化弱 NR 模型中，对高最小熵消息分布，同一挑战绑定赋予 玄色 (Octarine) 已为 $BUFF$ 结构建立的弱 NR 性质。本文不声称 [\[CDF⁺20\]](#) 的原始强 NR 概念。

Proof. 我们将 H_{BUFF} 中的哈希调用视为随机预言机调用。因此，复合挑战哈希的抗碰撞性对应于随机预言机碰撞事件；当两个复合输出相等时，可沿 H_{tr} 、 H_{mu} 和最后一次 H_{ch} 调用构成的链提取碰撞。

首先考虑攻击者输出单个签名 $\sigma = \text{sigEncode}(\tilde{c}, \mathbf{z}, \mathbf{h})$ ，该签名对两个元素组 (pk_1, M_1) 和 (pk_2, M_2) 均通过验证，其中公钥可由攻击者恶意选择。验证首先解码公钥字节；依假设，非规范编码会被拒绝，每个被接受的公钥均有唯一字节表示。随后解析同一 $\tilde{c}$ ，重新计算挑战 $c = \text{SampleInBall}(\tilde{c})$ ，从两个公钥视图恢复高位 $\mathbf{w}_{1,1}$ 和 $\mathbf{w}_{1,2}$ ，并且仅在下式成立时接受：

$$\tilde{c} = H_{BUFF}(pk_1, M_1, \mathbf{w}_{1,1}) = H_{BUFF}(pk_2, M_2, \mathbf{w}_{1,2}).$$

M - S - UEO 实验要求 $pk_1 \neq pk_2$ ，并允许 $M_1 \neq M_2$ ； MBS 实验则要求 $pk_1 = pk_2$ 且 $M_1 \neq M_2$ 。由于所有被接受的三元组 $(pk, M, \mathbf{w}_1)$ 均采用单射编码，除可忽略概率外，任一获胜事件都会在复合哈希链某处给出底层随机预言机碰撞。同一 M - S - UEO 结论蕴含较弱的排他所有权变体 S - CEO 和 S - DEO 。定理中的 $Q_H + 6$ 项计入攻击者的预言机查询，以及在获胜检查中重新计算两条三步挑战哈希链所需的至多六次随机预言机求值。

对于不可重签名性，上述碰撞论证本身不能证明原始强 NR 概念：已知该概念在通用 $BUFF$ 环境中不可实现，且原证明路线存在缺口 [\[DFHS24\]](#)。我们改用在理想化捉迷藏模型中为 $BUFF$ 结构证明的较弱表述 [\[DFH⁺24\]](#)。玄色 (Octarine) 签名已包含 $BUFF$ 挑战摘要

$$\tilde{c} = H_{BUFF}(pk, M, \mathbf{w}_1),$$

，验证会根据公钥、消息和恢复出的高位重新计算该摘要。在捉迷藏视图中，基础签名分量是原生 Fiat-Shamir 响应与提示数据 $(\mathbf{z}, \mathbf{h})$ ，连同验证者对 $\mathbf{w}_1$ 的确定性重构；附加 $BUFF$ 摘要即挑战种子 $\tilde{c}$ ，玄色 (Octarine) 已存储该值，而非将其作为第二签名分量另行添加。因此，暴露给理想化弱 NR 游戏的输入是带摘要 $\tilde{c}$ 的 $BUFF$ 元素组 $(pk, M, \mathbf{w}_1)$ 。对于从所需高最小熵分布抽取的消息，成功的重签名攻击将导致消息预测、底层随机预言机碰撞，或攻破 [\[DFH⁺24\]](#) 所分析的弱 NR 游戏。□

6.2 实践安全性

6.2.1 物理安全性

签名算法使用长期签名密钥，因而是侧信道攻击的首要目标。鉴于其高层带中止 Fiat-Shamir 结构与 ML - DSA 相似，预计此类攻击也与针对 ML - DSA 的研究相近 [\[RCDB24\]](#)。同时，玄色 (Octarine) 的若干设计选择旨在简化加固实现。尤其是二的幂模数使采样、舍入、分解和比特选择操作天然适合以移位、掩码和加法构成的固定时间实现。这些操作也很适合基于掩蔽的对策，因为避免了奇素数模算术通常引入的额外复杂性。

因此，预计 玄色 (Octarine) 采用掩蔽等对策的开销低于 ML-DSA。这与先前关于掩蔽 Dilithium 的研究一致；该研究发现，以二的幂模数替代素数模数可显著降低掩蔽开销 [MGTF19]。玄色 (Octarine) 的详细侧信道评估留待未来工作，但其二的幂算术和规则采样过程为常数时间及掩蔽实现提供了良好起点。

7 性能评估

7.1 算术实现

归藏 (ARCANE) 套件有意不为 KEM (软猬 (Mithril)) 或签名方案 (玄色 (Octarine)) 规定唯一的多项式乘法策略。因此，多项式乘积可采用学校书式乘法 (schoolbook multiplication)、Karatsuba、Toom-Cook、基于 NTT 的方法、Kronecker 型技术，或任何适合目标平台的方法实现。这属于实现选择而非规范组成部分，使实现者可灵活选择最符合可用硬件、性能目标、内存约束及加固要求的算术算法。

两个方案规定的算术均使用二的幂模数，使模方案模数的约减格外简单，并使舍入、打包和分解操作接近比特提取。此外，玄色 (Octarine) 模数的比特长度约为软猬 (Mithril) 模数的两倍。这一关系有利于共享软硬件实现：按较大玄色系数设计的数据通路也可自然容纳软猬算术，而较小的子例程通常可在次数更大的玄色计算中复用。

尽管规定模数为二的幂，实现内部仍可使用辅助 NTT 素数。例如，可将输入提升为有符号整数并映射到辅助素域 $\mathbb{Z}_\pi$ ，在模 π 下执行负循环 NTT 乘法，再将恢复的整数结果模 q_K 约减，从而计算 $\mathbb{Z}_{q_K}[x]/(x^n + 1)$ 中的乘积。辅助素数 π 应支持所需负循环变换长度；通常这意味着 π 含本原 $2n$ 次单位根，等价于 $\pi \equiv 1 \pmod{2n}$ 。它还必须足够大，使相关整数卷积系数不会模 π 回绕。对方案模数处于 12–16 比特范围的软猬，一个 20–31 比特 NTT 素数通常足够。

同一思想适用于玄色中的若干小乘积。挑战多项式 c 是基环的稀疏元素，并嵌入根式环的常数分量，故乘以 c 会逐分量作用于 k 个基环分量。由于 c 的汉明重量为 τ ，且 s_1 、 s_2 和 b_0 的系数有界， cs_1 、 cs_2 和 cb_0 等乘积的系数增长较小。因此，只要所选素数支持玄色的基次数，这些乘积通常可使用与 KEM 算术相同的辅助素数 NTT 基础设施计算。若 KEM 素数仅针对较短的软猬变换长度选择，实现可改用兼容最大所需变换长度的共享素数，或为这些挑战乘积另用较小的辅助素数。

较大的玄色乘积，特别是 As_1 、 Ay 和 Az ，需要更加谨慎。它们是根式环

$$S_{q_D, n, k} = (\mathbb{Z}_{q_D}[y]/(y^n + 1))[x]/(x^k - y - 2),$$

中的乘积，实现为负循环基环乘法结构化组合。由于 $q_D = 2^{24}$ 、 y 和 z 的系数约为 20 比特且 $n = 1024$ ，单个 32 比特辅助素数通常不能提供足够动态范围来无回绕地恢复整数乘积系数。因此自然的方案是使用两个辅助 NTT 素数 π_1 和 π_2 。将输入提升为有符号整数，分别模每个 π_i 约减，并在

$$\mathbb{Z}_{\pi_i}[y]/(y^n + 1),$$

中独立相乘，再通过 CRT 模 $M = \pi_1\pi_2$ 重组结果。若 M 大于相关有符号系数增长界的两倍，CRT 结果可唯一恢复整数乘积系数，随后再模 q_D 约减。

对于玄色参数集，预期系数增长界表明，两个小于 32 比特的 NTT 素数之积已经足够：约 56–60 比特的模数乘积可覆盖占主导的 Ay 和 Az 乘积，包括根式环分量间的累加。这也允许两个素数之一与软猬使用的辅助素数相同，前提是该素数支持共享实现所需的最大变换长度。这样，KEM 算术可直接复用为玄色双素数 CRT 计算的一个分支。

具体而言, 对每个辅助素数 π_i , 实现在模 π_i 的 NTT 域中计算所需基环乘积. 加法、减法、分量乘积以及关系 $x^k = y + 2$ 诱导的结构化累加, 均分别在模 π_1 和模 π_2 下独立执行. 无需在每次单独基乘法后进行逆 CRT 重组; 实现可将所有中间累加值保持为双素数表示, 仅在完整根式环乘积甚至整个矩阵乘积完成后重组. 这样可摊销 CRT 成本; 随着根式扩张次数 k 增大, 与公开对象 A 的乘积主导算术成本, 因此这一做法对玄色尤其有利.

最后, 两个辅助素数分支基本可共享同一软件结构. 在 Montgomery 实现中, 算术层可由素数 π_i 、相应 Montgomery 常数 $-\pi_i^{-1} \bmod R$ 和预计算 NTT 单位根参数化; 控制流、变换结构、逐点乘法例程和累加逻辑均无需改变. 因此, 同一实现框架只需少量参数变更即可支持两个辅助素数, 进一步实现 归藏 (ARCANE) 系列内算术复用的总体目标.

7.2 性能分析

玄色 (Octarine) 的成本特征由固定基环、根式扩张次数以及签名拒绝循环决定. 三个参数集均使用 $n = 1024$ 和 $q = 2^{24}$, 而 玄色 (Octarine) -128、玄色 (Octarine) -256 和 玄色 (Octarine) -512 的根式扩张次数分别为 $k = 1, 2, 5$. 带中止 Fiat-Shamir 的预期迭代次数约为 2.1、2.3 和 3.6. 因此, 相关性能量包括扩展为环系数的伪随机材料数量、与公开对象 A 相乘的成本, 以及后续挑战乘积的成本.

采样与扩展. 主要的非算术成本来自将伪随机输出扩展为系数表示. 公开对象 A 由种子 ρ 确定性生成, 在根式环表示中由 k 个模 q 、各含 n 个系数的多项式组成. 由于 $q = 2^{24}$, 公开系数通过固定宽度解码输出流获得, 而非通过拒绝采样. 掩蔽元素 y 在每轮拒绝循环中从 $[-\gamma_1, \gamma_1 - 1]$ 采样, 故其扩展成本需乘以预期签名尝试次数.

对 A 和 y , 采样规范将每个多项式分为四条域分离流, 每条对应 $n/4$ 个系数的一块. 因此每个多项式需四次独立可扩展输出函数求值和四条独立系数解码流; 这是规则的域分离构造, 而非依赖数据的采样过程. 相比之下, s_1 只在密钥生成时采样一次, 每个系数仅使用 $\log_2(2\eta)$ 比特. 存储的舍入误差分量 s_2 不独立采样, 而是在密钥生成时由 As_1 和舍入后的公开值派生.

SampleInBall 生成的挑战多项式 c 的汉明重量为 τ . 但受测实现对后续与 c 的乘积仍采用规则 NTT 域乘法, 而未采用内存访问或控制流依赖 c 稀疏支撑集的乘法规则.

与公开对象的乘积. 三个算法的主要算术操作为

$$As_1, \quad Ay, \quad Az.$$

这些乘积在有符号整数提升和两个辅助 NTT 素域上计算. 对 $u \in \{s_1, y, z\}$, 可将计算抽象表示为

$$Au = \text{red}_q \left(\text{CRT}_{\pi_1, \pi_2} \left(\text{INTT}_{\pi_1} (\hat{A}^{(1)} \odot \hat{u}^{(1)}), \right. \right. \\ \left. \left. \text{INTT}_{\pi_2} (\hat{A}^{(2)} \odot \hat{u}^{(2)}) \right) \right), \quad (8)$$

其中上标表示约减至相应辅助素域, 尖帽表示 NTT 域表示, red_q 表示约减回模 q 的有符号代表元. 当 $\pi_1 \pi_2$ 覆盖相关整数系数增长界时, 该表示有效. 此处 $\odot$ 表示根式环乘法诱导的结构化 NTT 域乘积与累加, 而非单次逐分量乘积.

直接来看, 根式扩张中的乘法产生 k 个输出分量, 涉及基多项式乘积的结构化组合. 优化实现可利用约减关系 $x^k = y + 2$ 并复用预处理表示, 故下式应理解为解释性模型, 而非精确周

期公式：

$$C_A(k) \approx 2(kC_{\text{NTT}} + M(k)C_{\text{pt}} + kC_{\text{INTT}}) + kC_{\text{CRT}}, \quad (9)$$

在直接根式环视图中， $M(k) = O(k^2)$ 。该模型分别表示两个辅助素域、 k 个分量的变换、分量乘积及累加，以及最终 CRT 恢复。它解释了为何随 k 增大，与 $\mathbf{A}$ 的乘积预计会成为主要算术操作；精确成本则由实现测量结果给出。

挑战乘积。 与挑战多项式的乘积具有不同结构。挑战 c 是嵌入根式环常数分量的单个基环多项式，故乘以 c 会逐分量作用于

$$\mathbf{s}_1, \quad \mathbf{s}_2, \quad \mathbf{b}_0, \quad \mathbf{b}_1.$$

在受测实现中， c 只变换一次，随后用于规则的逐分量 NTT 乘积。完成该变换后，一组挑战乘积的成本因而大致随 k 线性增长，并避免了与 $\mathbf{A}$ 相乘时存在的跨分量公开乘积结构。

算法级分解。 在三个主要算法的层面，可得到如下粗略分解：

$$C_{\text{KeyGen}} \approx C_{\text{seed}} + C_{D_A} + C_{D_{s_1}} + C_A(k) + O(kn), \quad (10)$$

$$C_{\text{Sign}} \approx C_{\text{pre}} + r_{\text{FS}} \left(C_{D_y} + C_A(k) + C_{w_1} + C_{\text{SampleInBall}} + C_{c\text{-prod}} + C_{\text{MakeHint}} \right) + C_{\text{pack}}, \quad (11)$$

$$C_{\text{Verify}} \approx C_{\text{unpack}} + C_{D_A} + C_A(k) + C_{\text{SampleInBall}} + C_{\text{cb}_1, \text{hint}} + C_{\text{hash}}. \quad (12)$$

其中 r_{FS} 表示拒绝循环的预期迭代次数。乘以 r_{FS} 的各项是拒绝循环中每次尝试的操作，包括仅在先前拒绝检查通过后才执行的操作；最终签名编码成本 C_{pack} 在某轮被接受后只产生一次。项 $C_{c\text{-prod}}$ 汇总签名循环中挑战与 $\mathbf{s}_2$ 、 $\mathbf{s}_1$ 和 $\mathbf{b}_0$ 的乘积。项 $C_{\text{cb}_1, \text{hint}}$ 包含挑战与 $\mathbf{b}_1$ 的乘积、相应高位重构以及验证中的提示应用。项 C_{D_A} 和 C_{D_y} 包含将伪随机输出扩展为环系数的成本。下一小节的实现结果通过分别给出整函数计时与插桩阶段计时来检验该分解。

7.3 性能测试

实验设置。 本节全部软件测量均在 Intel Xeon E5-2686 v4 处理器的单个核心上完成，`/proc/cpuinfo` 报告频率为 2.30 GHz。机器配有 3.8 GiB 内存，在 x86_64 上运行 Ubuntu GNU/Linux，Linux 内核为 7.0.0-1006-aws。编译器为 gcc 15.2.0。基准测试串行运行，每个进程固定到单个 CPU；每次运行报告 1000 次测量执行的中位数，测试框架报告单位为周期/计时刻度。

构建配置如下。使用所提供 pseudoXOF 的参考实现在 `-O3` 下编译，并采用参考 Makefile 的警告选项，见下表。SHA3/SHAKE AVX2 实现采用与所提供 pseudoXOF AVX2 实现相同的 AVX2 算术选项，并使用 SHA3/SHAKE 后端和四路 Keccak 支持。

实现	编译器选项
参考 pseudoXOF	<code>-O3</code> ，以及参考 Makefile 警告选项
AVX2 pseudoXOF	<code>-O3, -mavx2, -mbmi2, -flto</code> <code>-fno-semantic-interposition, -funroll-loops</code> <code>-fno-inline-functions-called-once</code> ，以及警告选项
ML-DSA AVX2	<code>-O3, -mavx2, -mpopcnt</code> <code>-march=native, -mtune=native</code>

用于比较的 ML-DSA 实现是官方 CRYSTALS-Dilithium AVX2 实现，检出于 Git 提交

6e00625c5b29f516c6de973fe2ee2fbb150973f9.

采用 pseudoXOF 的参考实现与 AVX2 实现性能. 表 3 首先给出所提交实现的整函数成本. 两个实现使用相同的 pseudoXOF 接口，因此该比较单独反映所提交哈希接口下 AVX2 算术层和打包层的贡献.

Table 3: 采用 pseudoXOF 的参考实现和 AVX2 实现的整函数性能. 数值为周期/计时刻度中位数；加速比为参考实现/AVX2.

参数集	操作	参考实现	AVX2	加速比
玄色 (Octarine) -128	KeyGen	896 744	496 887	1.80×
	Sign	2 369 061	1 131 795	2.09×
	Verify	1 062 976	493 150	2.16×
玄色 (Octarine) -256	KeyGen	1 757 081	922 466	1.90×
	Sign	4 754 636	2 337 069	2.03×
	Verify	2 086 376	964 220	2.16×
玄色 (Octarine) -512	KeyGen	4 711 052	2 216 628	2.13×
	Sign	17 112 796	8 337 111	2.05×
	Verify	5 511 306	2 336 848	2.36×

表 4–6 给出相应的插桩分解结果. 阶段测量值是同一串行基准运行的中位数，应理解为阶段级测量，而非整函数中位数的精确加法重构.

Table 4: 玄色 (Octarine) -128 使用 pseudoXOF 的分解结果. 数值为周期/计时刻度中位数；加速比为参考实现/AVX2.

阶段	参考实现	AVX2	加速比
KeyGen 总计	900 817	498 804	1.81×
KeyGen 随机数/哈希	29 735	27 545	1.08×
KeyGen D_A 及准备	283 242	272 794	1.04×
KeyGen D_{s_1}	53 941	48 889	1.10×
KeyGen A_{s_1}	389 988	29 000	13.45×
KeyGen 哈希/存储	125 275	115 794	1.08×
Sign 总计	2 367 737	1 135 682	2.08×
Sign 准备 A	397 040	272 558	1.46×
Sign 解包/变换秘密	170 301	17 947	9.49×
Sign 哈希 μ, ρ''	64 508	59 844	1.08×
Sign D_y	659 362	613 577	1.07×
Sign 变换 y	225 517	24 109	9.35×
Sign Ay	332 398	33 370	9.96×
Sign w_1 /挑战哈希	46 401	40 955	1.13×
Sign SampleInBall 及变换	134 865	34 276	3.93×
Sign cs_2 及检查	147 580	15 512	9.51×
Sign cs_1 及检查	73 656	7 707	9.56×
Sign cb_0 及提示	74 574	8 538	8.73×
Sign 打包签名	7 314	3 754	1.95×
Verify 总计	1 064 732	495 839	2.15×
Verify 解包/检查	4 837	1 491	3.24×
Verify 准备 A	283 158	273 427	1.04×
Verify Az	388 774	28 892	13.46×
Verify SampleInBall 及变换	67 795	17 318	3.91×
Verify 解包提示	844	635	1.33×
Verify cb_1 及高位	131 036	14 665	8.94×
Verify 哈希比较	170 207	157 216	1.08×

分解结果支持上一小节的成本分解. 与 A 的大乘积，即 A_{s_1} 、 Ay 和 Az ，在优化算术中获
得约一个数量级的加速；规则 NTT 域挑战乘积同样如此. 相比之下，由 pseudoXOF 接口和最

Table 5: 玄色 (Octarine) -256 使用 pseudoXOF 的分解结果. 数值为周期/计时刻度中位数; 加速比为参考实现/AVX2.

阶段	参考实现	AVX2	加速比
KeyGen 总计	1 798 146	920 670	1.95×
KeyGen 随机数/哈希	29 799	27 754	1.07×
KeyGen D_A 及准备	566 443	548 733	1.03×
KeyGen D_{s_1}	71 356	65 731	1.09×
KeyGen A_{s_1}	876 998	61 067	14.36×
KeyGen 哈希/存储	216 116	199 375	1.08×
Sign 总计	4 875 423	2 340 256	2.08×
Sign 准备 A	812 532	549 769	1.48×
Sign 解包/变换秘密	356 823	36 009	9.91×
Sign 哈希 μ, ρ''	64 937	60 251	1.08×
Sign D_y	1 413 432	1 301 403	1.09×
Sign 变换 y	473 736	50 484	9.38×
Sign A_y	816 980	71 473	11.43×
Sign w_1 /挑战哈希	140 254	125 440	1.12×
Sign SampleInBall 及变换	154 201	45 286	3.41×
Sign cs_2 及检查	230 142	23 994	9.59×
Sign cs_1 及检查	152 426	15 309	9.96×
Sign cb_0 及提示	154 134	16 892	9.12×
Sign 打包签名	12 727	8 809	1.44×
Verify 总计	2 147 015	959 349	2.24×
Verify 解包/检查	12 142	3 015	4.03×
Verify 准备 A	566 007	549 382	1.03×
Verify A_z	877 855	60 561	14.50×
Verify SampleInBall 及变换	77 550	22 792	3.40×
Verify 解包提示	2 049	1 531	1.34×
Verify cb_1 及高位	276 043	30 060	9.18×
Verify 哈希比较	305 710	282 061	1.08×

终哈希重构主导的阶段仅略有改善. 这解释了为何表 3 的整函数加速比约为 $2\times$, 尽管占主导的算术阶段加速幅度大得多.

采用 SHA3/SHAKE 的性能比较. 表 7 比较基于 SHA3/SHAKE 的 玄色 (Octarine) AVX2 实现与 ML-DSA AVX2 实现. 比较对象是安全级别相近而非完全相同的参数集: 玄色 (Octarine) -128 与 ML-DSA-44 比较, 玄色 (Octarine) -256 与 ML-DSA-87 比较. 玄色 (Octarine) -512 的目标安全级别显著高于所有 ML-DSA 参数集, 故不列入该表. 数值越低表示性能越好.

Table 7: 采用 SHA3/SHAKE AVX2 实现的 玄色 (Octarine) 与 ML-DSA 比较. 数值为周期/计时刻度中位数; 粗体表示较快的中位数.

比较	操作	玄色 (Octarine)	AVX2	ML-DSA	AVX2	中位数较快者
玄色 (Octarine) -128 与 ML-DSA-44	密钥生成	79 994		87 164		玄色 (Octarine) , 1.09×
	签名	204 968		203 781		ML-DSA 0.994×
	验证	102 704		91 568		ML-DSA 0.89×
玄色 (Octarine) -256 与 ML-DSA-87	密钥生成	153 377		236 020		玄色 (Octarine) , 1.54×
	签名	383 428		433 228		玄色 (Octarine) , 1.13×
	验证	190 885		234 677		玄色 (Octarine) , 1.23×

Table 6: 玄色 (Octarine) -512 使用 pseudoXOF 的分解结果. 数值为周期/计时刻度中位数; 加速比为参考实现/AVX2.

阶段	参考实现	AVX2	加速比
KeyGen 总计	4 697 799	2 223 798	2.11×
KeyGen 随机数/哈希	30 080	28 054	1.07×
KeyGen D_A 及准备	1 441 244	1 401 922	1.03×
KeyGen D_{s_1}	178 394	163 880	1.09×
KeyGen A_{s_1}	2 541 761	169 657	14.98×
KeyGen 哈希/存储	465 667	430 248	1.08×
Sign 总计	17 134 788	8 319 675	2.06×
Sign 准备 A	2 051 675	1 400 079	1.47×
Sign 解包/变换秘密	898 403	91 148	9.86×
Sign 哈希 μ, ρ''	64 779	60 492	1.07×
Sign D_y	5 773 898	5 361 874	1.08×
Sign 变换 y	1 794 654	194 953	9.21×
Sign A_y	3 991 001	318 462	12.53×
Sign w_1 /挑战哈希	589 284	532 365	1.11×
Sign SampleInBall 及变换	302 456	134 714	2.25×
Sign cs_2 及检查	790 328	79 662	9.92×
Sign cs_1 及检查	399 837	38 966	10.26×
Sign cb_0 及提示	396 684	42 337	9.37×
Sign 打包签名	30 870	18 845	1.64×
Verify 总计	5 496 793	2 343 033	2.35×
Verify 解包/检查	14 960	8 124	1.84×
Verify 准备 A	1 441 628	1 403 678	1.03×
Verify A_z	2 531 417	169 927	14.90×
Verify SampleInBall 及变换	101 761	45 093	2.26×
Verify 解包提示	4 393	3 162	1.39×
Verify cb_1 及高位	701 161	75 916	9.24×
Verify 哈希比较	682 770	625 908	1.09×

致谢

我们谨感谢三位博士研究生 Shicong Zhang、Chenxin Zhong 和 Haochen Huang 参与算法设计的技术讨论, 并为提交材料的准备作出贡献. 其中, Shicong Zhang 参与算法描述, Chenxin Zhong 参与安全性证明, Haochen Huang 参与软件实现.

A SageMath 脚本

A.1 安全性估计

我们使用 Albrecht 等人的 lattice-estimator 框架 [APS15] 估计 RR-LWR 实例的具体困难度. 具体采用以下格估计器实现: [malb/lattice-estimator](#), 提交版本为 [be81b18](#).

表 2 所报告的经典与量子比特安全性估计及精化攻击结果, 可在 lattice-estimator 仓库中运行以下脚本复现. 为获得 512 比特安全级别的估计, 需要在 `estimator/conf.py` 中将块大小上界调整为

```

1 max_beta = 3500

1 from estimator import *
2 from sage.all import binomial, log
3
4 q = 2**24
5 n = 1024

```

```

6 eta128 = 4
7 eta = 2
8
9 sig128 = {
10     "q": q,
11     "p": q / (2*eta128),
12     "n": n,
13     "k": 1,
14     "eta": eta128,
15     "gamma1": 2**18,
16     "gamma2": 2**17,
17     "tau": 16,
18     "d": 11,
19     "omega": 79,
20     "Xs": ND.UniformMod(2*eta128)
21 }
22
23 sig256 = {
24     "q": q,
25     "p": q/(2*eta),
26     "n": n,
27     "k": 2,
28     "eta": eta,
29     "gamma1": 2**19,
30     "gamma2": 2**18,
31     "tau": 36,
32     "d": 13,
33     "omega": 210,
34     "Xs": ND.UniformMod(2*eta)
35 }
36
37 eta = 2
38 sig512 = {
39     "q": q,
40     "p": q/(2*eta),
41     "n": n,
42     "k": 5,
43     "eta": eta,
44     "gamma1": 2**21,
45     "gamma2": 2**20,
46     "tau": 87,
47     "d": 14,
48     "omega": 399,
49     "Xs": ND.UniformMod(2*eta)
50 }
51
52 def evaluate_params(params):
53
54     q      = params["q"]
55     p      = params["p"]

```

```

56     n      = params["n"]
57     k      = params["k"]
58     gamma1 = params["gamma1"]
59     gamma2 = params["gamma2"]
60     tau    = params["tau"]
61     eta    = params["eta"]
62     d      = params["d"]
63     omega  = params["omega"]
64     beta   = tau*eta # Max size of challenge (in absolute value)
65     Xs     = params["Xs"]
66     Xe     = ND.UniformMod(2*q/(2*p))
67
68     assert(q/(2*p) == eta) # If this is not true, then c*s2 needs to be
        handled separately
69
70     print('beta = ' + str(beta))
71
72     lwe = LWE.Parameters(
73         n=k*n,
74         q=q,
75         Xs=Xs,
76         Xe=Xe,
77         m=k*n,
78         tag="LWE"
79     )
80
81     LWE.estimate(lwe)
82
83     zeta = max((gamma1-beta), 2*gamma2 + 1 + q/p*2**(d-1)*tau)
84     print(zeta, gamma1-beta, 2*gamma2 + 1 + q/p*2**(d-1)*tau)
85     selftargetmsis = SIS.Parameters(
86         n=k*n,
87         q=q,
88         length_bound=zeta,
89         m=3*k*n,
90         norm=+Infinity,
91         tag="SIS"
92     )
93
94     print('selftargetmsis (forgery)')
95     print(selftargetmsis)
96     SIS.estimate(selftargetmsis)
97
98     zeta_ = max(2*(gamma1-beta), 4*gamma2 + 2)
99     print(zeta_, 2*(gamma1-beta), 4*gamma2 + 2)
100    msis = SIS.Parameters(
101        n=k*n,
102        q=q,
103        length_bound=zeta_,
104        m=2*k*n,

```

```

105     norm=+Infinity,
106     tag="SIS"
107 )
108
109 print('msis (strong unforgeability)')
110 print(msis)
111 SIS.estimate(msis)
112
113 # Compute (approximate) rejection probabilities and expected number of
    tries
114 prob_z = ((2*(gamma1 - beta) - 1) / (2*gamma1)) ** (k*n)
115 prob_r0 = ((2*(gamma2 - beta) - 1) / (2*gamma2)) ** (k*n)
116 prob_acc = prob_z * prob_r0
117 E_tries = float(1/prob_acc)
118
119 print("E_tries = " + str(E_tries))
120
121 # Only second-preimage resistance is required
122 challenge_entropy = log(binomial(n, tau) * 2**tau, 2)
123 print("Challenge entropy = " + str(int(challenge_entropy)))
124
125 # Figure out the size of the prime for NTT operations
126 coefficient_size = 1
127 coefficient_size *= 2          # taking into account sign for y and z
128 if k > 1:
129     coefficient_size *= 3      # for contribution of (y+2)
130     coefficient_size *= q-1    # coefficient size of A
131     coefficient_size *= gamma1-1 # coefficient size of y and z
132     coefficient_size *= n      # summing across polynomial
133     coefficient_size *= k      # summing across matrix
134     print("max coefficient bit length = " + str(float(log(coefficient_size
        ,2))))
135
136 # Compare the conservative bound for q/p*c*b0 against gamma2.
137 print("q/p*c*b0: " + str(q/p*tau*2**(d-1)))
138 print("gamma2: " + str(gamma2))
139 print("conservative ||q/p*c*b0||_inf bound < gamma2: " + str(q/p*tau
    *2**(d-1) < gamma2))
140
141 # Sizes
142 print("sk (bytes): " + str(int(64 + 64 + 128 + 2*k*n*ceil(log(2*eta, 2)
    )/8 + k*n*d/8)))
143 print("pk (bytes): " + str(int(64 + k*n*(ceil(log(p, 2)) - d)/8)))
144 print("sig (bytes, without h): " + str(int(64 + k*n*log(2*gamma1,2)/8))
    )
145
146 evaluate_params(sig128)
147 evaluate_params(sig256)
148 evaluate_params(sig512)

```

B 检查清单

以下检查清单依据提交要求文件编制，便于定位相关内容。

- 1. 算法描述
 - 完整算法运行流程：第 4 节，尤其是算法 3.
 - 数字签名 KeyGen、Sign 和 Verify 算法：第 4 节，尤其是算法 3.
 - 密钥生成、签名和验证的高层描述：第 2.1 节.
 - 数学记号与基本定义：第 3.1 节.
 - LWE/LWR 背景与公式：第 3.2 节.
 - 根式环结构与公式：第 3.3 和 3.4 节.
 - 根式环上的结构化乘法：第 3.4 节，尤其是式 (6).
 - 所有安全强度级别的算法实例参数：第 4 节，表 2.
 - 玄色-128、玄色-256 和玄色-512 的参数：第 4 节，表 2.
 - 私钥、公钥和签名大小：第 4 节，表 2.
 - 参数表中的重复次数与安全性估计：第 4 节，表 2.
 - 采样、随机性、种子、哈希/XOF 使用及输入输出处理：第 3.1 和 4 节.
 - 支持算法描述的表格：第 2.3 节表 1；第 4 节表 2；以及第 7 节表 3-7.
 - 算法规范：第 4 节，尤其是算法 3.
- 2. 设计原理
 - 主要设计思想与原理：第 1、2.1 和 5 节.
 - 数学困难问题：第 2.3、3.4 和 6.1.1 节.
 - 以 RR-LWR 为底层假设：第 1、2.1、3.4 和 6.1.1 节.
 - 不可伪造性的 SIS 型假设：第 2.3 和 6.1.2 节.
 - 参数选择准则：第 5.1 节及表 2.
 - 根式环结构的选择：第 1、2.1、3.4 和 5.1 节.
 - n 、 k 、 p 、 q 、 η 、 d 、 τ 、 λ 、 γ_1 、 γ_2 、 β 和 ω 的选择：表 2 及第 5.1 节.
 - 二的幂模数与均匀采样的使用：第 1、2.1、5.1 和 5.3 节.
 - 正确性与拒绝概率分析：第 5.2 节.
 - 其他设计考虑：第 5.3 节.
 - 与现有工作的关系及比较依据：第 2.2 节.
- 3. 安全性声明与分析
 - 算法安全性声明：第 2.3 和 6 节.
 - 理论安全性：第 6.1 节.
 - RR-LWR 困难问题：第 6.1.1 节.
 - 玄色的不可伪造性：第 6.1.2 节.
 - 安全模型与安全性证明：第 6.1.2 节，定理 4 和 5.
 - 随机预言机模型安全性分析：第 2.3 和 6.1.2 节.
 - 实践安全性：第 6 节.
 - 各实例已知经典与量子攻击的复杂度：第 2.3 节表 1；第 4 节表 2；以及附录 A.1.

- 密钥恢复与不可伪造性安全估计：第 2.3 节表 1；第 4 节表 2；以及附录 A.1.
- BUFF 性质：第 6.1.3 节.
- 可复现安全估计脚本：附录 A.1.
- 物理安全性：第 6 节.
- 4. 性能评估
 - 性能分析：第 7.1 和 7.2 节.
 - 算术实现分析：第 7.1 节.
 - 参考实现算术：第 7.1 节.
 - AVX2 实现算术：第 7.1 节.
 - 采样与扩展成本分析：第 7.2 节.
 - 根式环乘积成本分析：第 7.2 节.
 - 操作级性能分析：第 7.2 节.
 - 通信与存储成本分析：表 2 及第 7.2 节.
 - 所有算法实例在主流 64 位 PC 处理器上的性能测试结果：第 7.3 节.
 - 实现平台的详细配置：第 7.3 节“实验设置”. 文档给出 CPU、时钟频率、指令集、操作系统、编译器、实现语言和基准方法.
 - KeyGen、Sign 和 Verify 的整函数性能结果：第 7.3 节，表 3 和 7.
 - 参考实现与 AVX2 性能结果：第 7.3 节，表 3.
 - 算术分解结果：第 7.3 节，表 4–6.
 - SHA3/SHAKE AVX2 整函数性能结果：第 7.3 节，表 7.
 - 与 Dilithium AVX2 实现的比较：第 7.3 节，表 7.
- 5. 特性声明
 - 特性声明，包括创新性、简洁性、灵活性、兼容性、可扩展性及不同平台上的性能优势：第 1 节.

References

- [ADR02] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer, Berlin, Heidelberg, April / May 2002. [doi:10.1007/3-540-46035-7_6](https://doi.org/10.1007/3-540-46035-7_6).
- [AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 57–74. Springer, Berlin, Heidelberg, August 2013. [doi:10.1007/978-3-642-40041-4_4](https://doi.org/10.1007/978-3-642-40041-4_4).
- [APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015. [doi:doi:10.1515/jmc-2015-0016](https://doi.org/10.1515/jmc-2015-0016).

- [APUPG17] Troncoso-Pastoriza Alberto Pedrouzo-Ulloa, Juan Ramón and Fernando Pérez-González. Multivariate cryptosystems for secure processing of multidimensional signals. *ArXiv*, abs/1712.00848, 2017.
- [BBE⁺18] Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Benjamin Grégoire, Mélissa Rossi, and Mehdi Tibouchi. Masking the GLP lattice-based signature scheme at any order. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 354–384. Springer, Cham, April / May 2018. doi:[10.1007/978-3-319-78375-8_12](https://doi.org/10.1007/978-3-319-78375-8_12).
- [BCV20] Carl Bootland, Wouter Castryck, and Frederik Vercauteren. On the security of the multivariate ring learning with errors problem. In *ANTS-XIV*, volume 4, pages 57–71, 2020. doi:[10.2140/obs.2020.4.57](https://doi.org/10.2140/obs.2020.4.57).
- [BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO’98*, volume 1462 of *LNCS*, pages 26–45. Springer, Berlin, Heidelberg, August 1998. doi:[10.1007/BFb0055718](https://doi.org/10.1007/BFb0055718).
- [BGM⁺16] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the hardness of learning with rounding over small modulus. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 209–224. Springer, Berlin, Heidelberg, January 2016. doi:[10.1007/978-3-662-49096-9_9](https://doi.org/10.1007/978-3-662-49096-9_9).
- [BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012. doi:[10.1145/2090236.2090262](https://doi.org/10.1145/2090236.2090262).
- [BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Berlin, Heidelberg, April 2012. doi:[10.1007/978-3-642-29011-4_42](https://doi.org/10.1007/978-3-642-29011-4_42).
- [BRvV22] Joppe W. Bos, Joost Renes, and Christine van Vredendaal. Post-quantum cryptography with contemporary co-processors: Beyond Kronecker, Schönhage-Strassen & Nussbaumer. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3683–3697, Boston, MA, 2022. USENIX Association. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/bos>.
- [BRVW26] Joppe W. Bos, Joost Renes, Frederik Vercauteren, and Peng Wang. Structured module lattice-based cryptography: Fine-grained parameter selection without compromising performance. *IACR Transactions on Cryptographic Hardware and Embedded Systems (to appear)*, 2026(3), 2026. URL: <https://eprint.iacr.org/2026/098>.

- [CCD⁺24] Jung Hee Cheon, Hyeongmin Choe, Julien Devevey, Tim Güneysu, Dongyeon Hong, Markus Krausz, Georg Land, Marc Möller, Damien Stehlé, and MinJune Yi. HAETAE: Shorter lattice-based Fiat-Shamir signatures. *IACR TCHES*, 2024(3):25–75, 2024. doi:[10.46586/tches.v2024.i3.25-75](https://doi.org/10.46586/tches.v2024.i3.25-75).
- [CDF⁺20] Cas Cremers, Samed Düzlül, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. Cryptology ePrint Archive, Report 2020/1525, 2020. URL: <https://eprint.iacr.org/2020/1525>.
- [CGV14] Jean-Sébastien Coron, Johann Großschädl, and Praveen Kumar Vadnala. Secure conversion between Boolean and arithmetic masking of any order. In Lejla Batina and Matthew Robshaw, editors, *CHES 2014*, volume 8731 of *LNCS*, pages 188–205. Springer, Berlin, Heidelberg, September 2014. doi:[10.1007/978-3-662-44709-3_11](https://doi.org/10.1007/978-3-662-44709-3_11).
- [CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *CRYPTO’99*, volume 1666 of *LNCS*, pages 398–412. Springer, Berlin, Heidelberg, August 1999. doi:[10.1007/3-540-48405-1_26](https://doi.org/10.1007/3-540-48405-1_26).
- [Coo66] S. Cook. *On the minimum computation time of functions*. PhD thesis, Harvard University, 1966.
- [CT65] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.
- [DFH⁺24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, Jyun-Jie Liao, and Patrick Struck. Hide-and-seek and the non-resignability of the BUFF transform. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part III*, volume 15366 of *LNCS*, pages 347–370. Springer, Cham, December 2024. doi:[10.1007/978-3-031-78020-2_12](https://doi.org/10.1007/978-3-031-78020-2_12).
- [DFHS24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, and Patrick Struck. On the (in)security of the BUFF transform. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 246–275. Springer, Cham, August 2024. doi:[10.1007/978-3-031-68376-3_8](https://doi.org/10.1007/978-3-031-68376-3_8).
- [DKL⁺18] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018. URL: <https://tches.iacr.org/index.php/TCHES/article/view/839>, doi:[10.13154/tches.v2018.i1.238-268](https://doi.org/10.13154/tches.v2018.i1.238-268).
- [DKR⁺20] Jan-Pieter D’Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, Jose Maria Bermudo Mera, Michiel Van Beirendonck, and Andrea Basso. SABER. Technical report, National Institute of Standards and Technology, 2020. available at

<https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>.

- [DS24] Samed Düzl  and Patrick Struck. The role of message-bound signatures for the beyond UnForgeability features and weak keys. Cryptology ePrint Archive, Report 2024/1669, 2024. URL: <https://eprint.iacr.org/2024/1669>.
- [KO62] Anatolii Alekseevich Karatsuba and Yu P Ofman. Multiplication of many-digital numbers by automatic computers. *Doklady Akademii Nauk SSSR*, 145(2):293–294, 1962. URL: <https://www.mathnet.ru/eng/dan26729>.
- [Kro82] L. Kronecker. Grundz ge einer arithmetischen Theorie der algebraischen Gr ssen. *Journal f r die reine und angewandte Mathematik*, 92:1–122, 1882.
- [LDK⁺22] Vadim Lyubashevsky, L o Ducas, Eike Kiltz, Tancre de Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehl , and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Berlin, Heidelberg, May / June 2010. doi:10.1007/978-3-642-13190-5_1.
- [LS15] Adeline Langlois and Damien Stehl . Worst-case to average-case reductions for module lattices. *DCC*, 75(3):565–599, 2015. doi:10.1007/s10623-014-9938-4.
- [Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Berlin, Heidelberg, December 2009. doi:10.1007/978-3-642-10366-7_35.
- [Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Berlin, Heidelberg, April 2012. doi:10.1007/978-3-642-29011-4_43.
- [MGTF19] Vincent Migliore, Beno t G rard, Mehdi Tibouchi, and Pierre-Alain Fouque. Masking dilithium - efficient implementation and side-channel evaluation. In Robert H. Deng, Val rie Gauthier-Uma a, Mart n Ochoa, and Moti Yung, editors, *Applied Cryptography and Network Security - 17th International Conference, ACNS 2019, Bogota, Colombia, June 5-7, 2019, Proceedings*, Lecture Notes in Computer Science, pages 344–362. Springer, 2019. doi:10.1007/978-3-030-21568-2_17.
- [NR23] Parker Newton and Silas Richelson. A lower bound for proving hardness of learning with rounding with polynomial modulus. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*,

pages 805–835. Springer, Cham, August 2023.

[doi:10.1007/978-3-031-38554-4_26](https://doi.org/10.1007/978-3-031-38554-4_26).

- [OTF⁺20] Hiroki Okada, Atsushi Takayasu, Kazuhide Fukushima, Shinsaku Kiyomoto, and Tsuyoshi Takagi. A compact digital signature scheme based on the module-LWR problem. In Weizhi Meng, Dieter Gollmann, Christian Damsgaard Jensen, and Jianying Zhou, editors, *ICICS 20*, volume 11999 of *LNCS*, pages 73–90. Springer, Cham, August 2020. [doi:10.1007/978-3-030-61078-4_5](https://doi.org/10.1007/978-3-030-61078-4_5).
- [Pei16] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends® in Theoretical Computer Science*, 10(4):283–424, 2016. [doi:10.1561/04000000074](https://doi.org/10.1561/04000000074).
- [PFH⁺22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [PN25] Rafaël Del Pino and Guilhem Niot. Finally! A compact lattice-based threshold signature. In Tibor Jager and Jiaxin Pan, editors, *PKC 2025, Part III*, volume 15676 of *LNCS*, pages 169–199. Springer, Cham, May 2025. [doi:10.1007/978-3-031-91826-1_6](https://doi.org/10.1007/978-3-031-91826-1_6).
- [Pol71] John M. Pollard. The fast Fourier transform in a finite field. *Mathematics of Computation*, 25(114):365–374, 1971.
- [PR13] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 142–159. Springer, Berlin, Heidelberg, May 2013. [doi:10.1007/978-3-642-38348-9_9](https://doi.org/10.1007/978-3-642-38348-9_9).
- [PUTPG⁺21] Alberto Pedrouzo-Ulloa, Juan Ramón Troncoso-Pastoriza, Nicolas Gama, Mariya Georgieva, and Fernando Pérez-González. Revisiting multivariate ring learning with errors and its applications on lattice-based cryptography. *Mathematics*, 9(8), 2021. URL: <https://www.mdpi.com/2227-7390/9/8/858>, [doi:10.3390/math9080858](https://doi.org/10.3390/math9080858).
- [PUTPPG15] Alberto Pedrouzo-Ulloa, Juan Ramón Troncoso-Pastoriza, and Fernando Pérez-González. Multivariate lattices for encrypted image processing. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1707–1711, 2015. [doi:10.1109/ICASSP.2015.7178262](https://doi.org/10.1109/ICASSP.2015.7178262).
- [PUTPPG16] Alberto Pedrouzo-Ulloa, Troncoso-Pastoriza, and Fernando Pérez-González. On ring learning with errors over the tensor product of number fields, 2016. URL: <https://arxiv.org/abs/1607.05244>.
- [RCDB24] Prasanna Ravi, Anupam Chattopadhyay, Jan Pieter D’ Anvers, and Anubhab Baksi. Side-channel and fault-injection attacks over lattice-based post-quantum schemes (Kyber, Dilithium): Survey and new results. *ACM Trans. Embed. Comput. Syst.*, 23(2), March 2024. [doi:10.1145/3603170](https://doi.org/10.1145/3603170).

- [Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. doi:[10.1145/1060590.1060603](https://doi.org/10.1145/1060590.1060603).
- [Reg10] Oded Regev. The learning with errors problem (invited survey). In *IEEE Conference on Computational Complexity*, pages 191–204, 2010. doi:[10.1109/CCC.2010.26](https://doi.org/10.1109/CCC.2010.26).
- [Too63] Andrei L Toom. The complexity of a scheme of functional elements realizing the multiplication of integers, published in soviet math (translations of dokl. adad. nauk. sssr), 4, 1963.