A Compact Digital Signature Scheme Based on the Module-LWR problem*

Hiroki Okada[1], Atsushi Takayasu[2†], Kazuhide Fukushima[1], Shinsaku Kiyomoto[1], and Tsuyoshi Takagi[2]

[1] KDDI Research, Inc., Japan
ir-okada@kddi-research.jp
[2] The University of Tokyo, Japan

December 28, 2021

Abstract. We propose a lattice-based digital signature scheme MLWRSign by modifying Dilithium, which is one of the third-Round finalists of NIST's call for post-quantum cryptographic standards. To the best of our knowledge, our scheme MLWRSign is the first signature scheme whose security is based on the (module) learning with rounding (LWR) problem. Due to the simplicity of the LWR, the secret key size is reduced by approximately 30% in our scheme compared to Dilithium, while achieving the same level of security. Moreover, we implemented MLWRSign and observed that the running time of MLWRSign is comparable to that of Dilithium.

Keywords: Lattice Cryptography, Digital Signatures, Learning with Rounding

1 Introduction

Lattice-based cryptography is believed to be a promising candidate for the NIST's call for post-quantum cryptographic (PQC) standards [Nat19]. In the second round of the NIST PQC [Nat20], for key encapsulation mechanisms (KEM), the lattice-based schemes proposed are the schemes based on the learning with errors (LWE) problem [Reg05], e.g. FrodeKEM [BCD+16], NewHope [ADPS16], CRYSTALS-Kyber [BDK+18], the learning with rounding (LWR)-based schemes Round5 [BBF+19] and SABER [DKRV18], and NTRU-based schemes [BCLv18, HRSS17]. For digital signatures, LWE-based schemes qTESLA [ABB+19], CRYSTALS-Dilithium [DKL+18, DLL+19, KLS18], and the NTRU-based scheme FALCON [FHK+19] are the only lattice-based schemes. In

*This paper is a revised version of [OTF+21]. We have corrected an error in Eq. (3), and slightly modified the Sign procedure of MLWRSign. See Appendix A for the details.

†During a part of this work, the author was affiliated with National Institute of Information and Communications Technology, Japan.

July 2020, the third-round finalists of the NIST PQC were announced [Nat21]. The finalists for KEM were CRYSTALS-Kyber, SABER, NTRU [HRSS17], and Classic McEliece [BCL+19]. The finalists for digital signatures are CRYSTALS-Dilithium, FALCON, and Rainbow [DCP+19]. NIST mentions in [Nat21] that NIST should standardize Classic McEliece and Rainbow for special-purpose use, since the schemes offers very small ciphertexts or signature at the expense of very large public keys. The rest of the finalists, CRYSTALS-Kyber, SABER, NTRU, CRYSTALS-Dilithium, and FALCON are all lattice-based schemes. It is also mentioned that NIST intends to select (at most) one lattice-based schemes for the standard for each of general-purpose KEM and digital signature. Here, note that no LWR-based scheme is proposed for signature in the NIST PQC, and moreover, no LWR-based signature scheme has been proposed to date.

Banerjee *et al.* [BPR12] introduced the LWR problem, which is a variant of LWE where the random errors are replaced by a deterministic rounding function. Bogdanov *et al.* [BGM+16] showed that there exists a reduction from search Ring-LWE (RLWE) to search Ring-LWR (RLWR). Following the work, Chen *et al.* [CZZ18] introduced a computational RLWR (CRLWR) problem, which is a counterpart of the computational Diffie-Hellman problem, and showed a reduction from decisional RLWE to CRLWR. This paper also showed that the KEM scheme based on Module-LWR (MLWR), to which RLWR can be viewed as a special case, Saber and the RLWR-based scheme Round5 are secure under the CRLWR assumption.

The RLWR-based KEM scheme, namely, the third-round finalist Saber, is among the most promising candidates for the NIST PQC standards due to the efficiency resulting from the simplicity of the RLWR problem. The RLWE-based KEM schemes require sampling noise from discrete Gaussian distributions, resulting in higher bandwidth. In contrast, RLWR-based KEM schemes naturally reduce bandwidth, avoiding additional randomness for the noise, since the (R)LWR problem generates noise through rounding of some least significant bits. RLWR schemes are usually designed with power-of-two moduli, and due to this, the rounding operation can be simply performed with a bit-shift operation. Furthermore, Beirendonck *et al.* [BDK+21] presented an efficient side-channel resistant masked implementation of Saber by leveraging the characteristic of the (R)LWR: power-of-two moduli, and limited noise sampling. While Oder *et al.* [OSPG18] presented a masked implementation of a complete chosen ciphertext attack (CCA) secure RLWE decapsulation similar to NewHope KEM [ADPS16] with a factor 5.7x overhead over an unmasked implementation, Saber's CCA-secure decapsulation algorithm [BDK+21] has an overhead factor of only 2.5x over the unmasked implementation.

The Module-LWE (RLWE)-based signature scheme CRYSTALS-Dilithium [DKL+18, DLL+19, KLS18] (hereinafter, referred to as Dilithium) is also among the most promising candidates due to its efficiency, especially on its public key size. Dilithium decreases the size of the public key by separating the high/low order bits of the element of the LWE sample. The high part is included in the public key and the low part is included in the secret key. This technique is

conceptually similar to the construction of the LWR-based KEM schemes. In the LWR, the low order bits are rounded off to be the deterministic noise (corresponds to a part of the secret key), and the high order bits are the LWR sample, which corresponds to the public key.

Our contributions. In this paper, we propose an MLWR-based digital signature scheme MLWRSign by modifying Dilithium. To the best of our knowledge, our scheme is the first digital signature scheme based on the (ring variants of) LWR problem. We modify Dilithium to be a MLWR-based scheme, aiming to obtain the best of both worlds of the LWR-based KEM schemes and Dilithium. As a result, the size of the secret key in our scheme is reduced by approximately 30%, compared to Dilithium. We present detailed analytical results on the probability of the rejection sampling during the signing procedure of our scheme, and show that the expected number of rejections is at the same level as Dilithium. This analysis is applicable to Dilithium, and it would be helpful for optimizing parameters of the scheme.

We efficiently implement MLWRSign and the results show that the running time of our scheme is comparable to Dilithium. Following the LWR-based KEM schemes such as Round5 and Saber, we also use all moduli of the powers of 2 in our scheme. Due to this setting, the bit decomposing technique in our scheme becomes simpler and more efficient. As discussed in [DKRV18], when the moduli are powers of 2, (negligibly small) exceptional biased sets exist for the secret key: If all coefficients of the polynomials in a secret vector are divisible by a high power of 2, then the same property will hold for the linear combination of them. However, since all the coefficients of a secret vector are small enough ($\leq 2^3$) in our parameters, our scheme can disregard the case. Although the number theoretic transform (NTT) cannot be used to speed up polynomial multiplication in our setting of the moduli, this disadvantage can be mitigated with Toom-Cook and Karatsuba polynomial multiplication. We implement our scheme using the Toom-Cook and Karatsuba, and the results show that the running time of our scheme is comparable to that of the reference implementation of Dilithium that uses NTT for polynomial multiplication.

This paper is the full version of the paper [OTF+20]. We have three main additional technical contributions over the preliminary version. First, we provide a full proof for the tight security reduction for MLWRSign in the Quantum random-oracle model (QROM) from the MLWR problem and another non-interactive assumption, based on the framework give in [KLS18]. Second, we present the additional parameter sets that 192- and 256-bits security, while neither the preliminary version [OTF+20] nor the Dilithium [DKL+18, DLL+19, KLS18] provides the corresponding parameter sets. Third, we give an optimized implementation of MLWRSign for CPUs that supports the AVX2 instruction set, and the results show that CPU cycles of AVX2 optimized versions of MLWRSign achieves 1.53x-2.06x speed-ups.

Organizations. We refer to the definition of QROM, canonical identification scheme, signature scheme, and the Fiat-Shamir transformation in Sect. 2. In

Sect. 3, we propose our identification scheme ID, and we construct the our signature scheme MLWRSign in Sect. 4 by using Fiat-Shamir transformation on ID. In Sect. 5 we provide a full proof for the tight security reduction for MLWRSign in the QROM from the MLWR problem and another non-interactive assumption. We implement MLWRSign and provide a comparison with other signature schemes proposed to NIST PQC in Sect. 6.

2 Preliminary

2.1 Notations

We write the rings $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = \mathbb{Z}_q[X]/(X^n + 1)$, where q and n are integers, and the value of n is always 256 throughout this paper. We denote elements in R or R_q (which includes elements in $\mathbb{Z}$ and $\mathbb{Z}_q$) in regular font letters, and bold lower-case letters represent column vectors whose elements are in R or R_q. All vectors will be column vectors by default. Bold upper-case letters are matrices. For a vector $\mathbf{v}$, we denote its transpose by $\mathbf{v}^\intercal$.

For an even (resp. odd) positive integer α, we define $r' = r \bmod^{\pm} \alpha$ to be the unique element r' in the range $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (resp. $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) such that $r' \equiv r \bmod \alpha$. For an element $u \in \mathbb{Z}_q$, let $\|u\|_\infty := \left|u \bmod^{\pm} q\right|$. We define the ℓ_∞ and ℓ_2 norms for a polynomial $w = \sum_{i=0}^{n-1} w_i X^i \in R$ as

$$\|w\|_\infty := \max_i \left|w_i \bmod^{\pm} q\right| \text{ and } \|w\| := \sqrt{\textstyle\sum_{j=0}^{n-1} \|w_0\|_\infty^2},$$

respectively. Similarly, for a vector $\mathbf{v} = (v_0, \ldots, v_{k-1}) \in R^k$, we define

$$\|\mathbf{v}\|_\infty := \max_i \|v_i\|_\infty \text{ and } \|\mathbf{v}\| := \sqrt{\textstyle\sum_{j=0}^{k-1} \|v_j\|}.$$

We define $S_\eta := \{w \in R \mid \|w\|_\infty \leq \eta\}$. Let B_h be the set of elements of R whose h coefficients are either -1 or 1 and the rest are 0. By $\mathrm{Hw}(\mathbf{w})$ we denote the # of non-zero coefficients in $\mathbf{w} \in R^k$ for $k > 0$.

We define rounding to the nearest integer as $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$, and we extend it to polynomials and matrices coefficient-wise. The Boolean operator $[\![\texttt{statement}]\!]$ outputs 1 if the `statement` is `true`, and 0 otherwise. We denote by $a \xleftarrow{\$} A$ the process of drawing an element a from a set A uniformly at random.

Let A be an algorithm. Unless otherwise stated, we assume all algorithms to be probabilistic. We denote by $y \leftarrow \mathsf{A}(x)$ probabilistic computation of the algorithm A on input x, where the output is stored as y. $\mathsf{A}(x) \Rightarrow y$ denotes the event that A on input x returns y. With fixed randomness, we can run any probabilistic A deterministically. We write $y := \mathsf{A}(x; r)$ to indicate that A is run on input x with a fixed randomness r.

We follow [BR06] to use code-based games. We implicitly assume that values of boolean flags, numerical types, sets, and strings are initialized to be false, 0, $\varnothing$, and the empty string ϵ, respectively. We make the convention that a procedure terminates once it returned an output. We write the event that an algorithm A return 1 for a game GAME by $\mathsf{GAME}^{\mathsf{A}} \Rightarrow 1$.

2.2 Quantum Computation

Quantum states. The state of a qubit $|\phi\rangle$ is described by a two-dimensional complex vector $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $\{|0\rangle, |1\rangle\}$ form an orthonormal basis of $\mathbb{C}^2$ and $\alpha, \beta \in \mathbb{C}$ with $|\alpha|^2 + |\beta|^2 = 1$ are the complex amplitudes of $|\phi\rangle$. The qubit $|\phi\rangle$ is called *in superposition* if $0 < |\alpha| < 1$. A classical bit $b \in \{0,1\}$ is naturally encoded as state $|b\rangle$ of a qubit.

The state of n qubits can be expressed by the linear combination $|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x |x\rangle \in \mathbb{C}^{2^n}$ where $\{\alpha_x\}_{x\in\{0,1\}^n}$ is a set of 2^n complex amplitudes such that $\sum_{x\in\{0,1\}^n} |\alpha_x|^2 = 1$. The standard orthonormal or *computational basis* is given by $\{|x\rangle\}_{x\in\{0,1\}^n}$. When the quantum state $|\psi\rangle$ is *measured* on a computational basis, the outcome is the classical string $x \in \{0,1\}^n$ with probability $|\alpha_x|^2$ and the quantum state collapses to the observed $|x\rangle$.

The evolution of a quantum system in state $|\psi\rangle$ can be described by a linear transformation $U : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$. The transformations correspond to unitary matrices $U \in \mathbb{C}^{2^n \times 2^n}$ and U has the property that $UU^\dagger = 1$, where $U^\dagger$ is the complex-conjugate transpose of U.

Quantum oracles and quantum adversaries. We follow the standard approach of [BBC+01, BDF+11] to execute the classical oracle function $\mathrm{O} : \{0,1\}^n \to \{0,1\}^m$ with a reversible unitary transformation. Let $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, and we model the quantum access to O by $U_\mathrm{O} : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus \mathrm{O}(x)\rangle$. Note that U_O is its own inverse, and also we obtain $U_\mathrm{O}^\dagger = U_\mathrm{O}$, and hence, $U_\mathrm{O}U_\mathrm{O}^\dagger = U_\mathrm{O}^2 = 1$. Quantum oracle adversaries $\mathsf{A}^{|\mathrm{O}\rangle}$ can access O in superposition by applying U_O. The quantum time that takes for applying U_O is linear in the time that takes to evaluate O classically. We write $\mathsf{A}^{|\mathrm{O}\rangle}$ to indicate that an oracle is quantum-accessible, contrary to oracles that can only be accessed classically denoted by A^O.

Quantum random-oracle model. We consider security games in the quantum random-oracle model (QROM) [BDF+11] as their counterparts in the classical random-oracle model [BR93], with the difference that we consider quantum adversaries that are given quantum access to the random oracles, and classical access to all other oracles such as the signing oracle. Zhandry [Zha12] proved that no quantum algorithm $\mathsf{A}^{|\mathsf{H}\rangle}$, issuing at most Q quantum queries to $|\mathsf{H}\rangle$, can distinguish between a random function $H : \{0,1\}^m \to \{0,1\}^n$ and a $2Q$-wise independent function f_{2Q}. Concretely, we regard $f_{2Q} : \{0,1\}^m \to \{0,1\}^n$ as a random polynomial of degree $2Q$ over the finite field $\mathbb{F}_{2^n}$. The running time to evaluate f_{2Q} is linear in Q. Let an adversary B simulates quantum adversary $\mathsf{A}^{|\mathsf{H}\rangle}$ which makes at most Q queries to $|\mathsf{H}\rangle$, then the running time of B is $\mathrm{Time}(\mathsf{B}) = \mathrm{Time}(\mathsf{A}) + q \cdot \mathrm{Time}(\mathsf{H})$, where $\mathrm{Time}(\mathsf{H})$ is the running time to simulate $|\mathsf{H}\rangle$. From this observation, B can use a $2Q$-wise independent function to simulate $|\mathsf{H}\rangle$ and we obtain that the running time of B is $\mathrm{Time}(\mathsf{B}) = \mathrm{Time}(\mathsf{A}) + Q \cdot \mathrm{Time}(f_{2Q})$, and the time $\mathrm{Time}(f_{2Q})$ to evaluate f_{2Q} is linear in Q. The second term of this running time that is quadratic in Q can be reduced to linear in Q in the QROM where B can simply use another random oracle to simulate $|\mathsf{H}\rangle$.

Assuming that the random oracle can be evaluated by one time unit, we write $\mathrm{Time}(\mathsf{B}) = \mathrm{Time}(\mathsf{A}) + Q \simeq \mathrm{Time}(\mathsf{A})$.

2.3 Problems

We define the MLWR problem, the Module-SIS (MSIS) problem, and the SelfTargetMSIS problem, on which the hardness and the security of our scheme MLWRSign is based.

Definition 2.1 ($\mathsf{MLWR}_{p,k,l,D}$ distribution). *Let q, p, k, l be positive integers such that $q > p \geq 2$. For a probability distribution $D : R_q \to \{0,1\}$, choose a random matrix $\mathbf{A} \xleftarrow{\$} R_q^{k\times l}$, and a vector $\mathbf{s} \leftarrow D^l$, and output $(\mathbf{A}, \lfloor \frac{p}{q}\mathbf{A}\mathbf{s} \rceil)$.*

Definition 2.2 (decision $\mathsf{MLWR}_{p,k,l,D}$ problem). *Given a pair $(\mathbf{A}, \mathbf{t})$ decide, with non-negligible advantage, whether it came from the $\mathsf{MLWR}_{p,k,l,D}$ distribution or it was generated uniformly at random from $R_q^{k\times l} \times R_p^k$. The advantage of an algorithm A in solving the decision $\mathsf{MLWR}_{p,k,l,D}$ problem is*

$$\begin{aligned}\mathrm{Adv}^{\mathsf{MLWR}}_{p,k,l,D}(\mathsf{A}) := \Big|&\Pr\left[b = 1 \;\middle|\; \mathbf{A} \leftarrow R_q^{k\times l}; \mathbf{t} \leftarrow R^k; b \leftarrow \mathsf{A}(\mathbf{A}, \mathbf{t})\right] \\ &- \Pr\left[b = 1 \;\middle|\; \mathbf{A} \leftarrow R^{k\times l}; \mathbf{s} \leftarrow D^l; b \leftarrow \mathsf{A}(\mathbf{A}, \lfloor \tfrac{p}{q}\mathbf{A}\mathbf{s} \rceil)\right]\Big|.\end{aligned}$$

We say MLWR is hard when the above advantage is negligible for all (quantum) probabilistic polynomial-time algorithms A.

Definition 2.3 ($\mathsf{MSIS}_{k,l,\zeta}$ problem). *Given $\mathbf{A} \xleftarrow{\$} R_q^{k\times l}$, find a vector $\mathbf{y} = \left[\mathbf{z}^\intercal \mid \mathbf{u}^\intercal\right]^\intercal \in R_q^{l+k}$ such that $\|\mathbf{y}\|_\infty \leq \zeta$ and $\left[\mathbf{A} \mid \mathbf{I}_k\right] \cdot \mathbf{y} = \mathbf{0}$. The advantage of an algorithm A in solving the $\mathsf{MSIS}_{k,l,\zeta}$ problem is*

$$\mathrm{Adv}^{\mathsf{MSIS}}_{k,l,\zeta}(\mathsf{A}) := \Pr\left[\begin{array}{l}\|\mathbf{y}\|_\infty \leq \zeta \wedge \\ \left[\mathbf{A} \mid \mathbf{I}_k\right] \cdot \mathbf{y} = \mathbf{0}\end{array} \;\middle|\; \begin{array}{l}\mathbf{A} \xleftarrow{\$} R_q^{k\times l}; \\ \mathbf{y} \leftarrow \mathsf{A}(\mathbf{A})\end{array}\right].$$

Definition 2.4 ($\mathsf{SelfTargetMSIS}_{\mathsf{H},k,l+1,\zeta}$ problem). *Let H $\{0,1\}^* \to B_{60}$ be a cryptographic hash function. Given a random matrix $\left[\mathbf{A} \mid \mathbf{t}\right] \xleftarrow{\$} R^{k\times(l+1)}$, find a message μ and a vector $\mathbf{y} = \left[\mathbf{z}^\intercal \mid c \mid \mathbf{u}^\intercal\right]^\intercal \in R^{l+1+k}$ such that $\|\mathbf{y}\|_\infty \leq \zeta$ and $\mathsf{H}\left(\mu \,\|\, \left[\mathbf{A} \mid \mathbf{t} \mid \mathbf{I}_k\right] \cdot \mathbf{y}\right) = c$. The advantage of an algorithm A in solving the $\mathsf{SelfTargetMSIS}_{\mathsf{H},k,l+1,\zeta}$ is*

$$\mathrm{Adv}^{\mathsf{SelfTargetMSIS}}_{\mathsf{H},k,l+1,\zeta}(\mathsf{A}) := \Pr\left[\begin{array}{l}\|\mathbf{y}\|_\infty \leq \zeta \wedge \\ c = \mathsf{H}\left(\mu \,\|\, \left[\mathbf{A} \mid \mathbf{t} \mid \mathbf{I}_k\right] \cdot \mathbf{y}\right)\end{array} \;\middle|\; \begin{array}{l}\left[\mathbf{A} \mid \mathbf{t}\right] \xleftarrow{\$} R^{k\times(l+1)}; \\ \mathbf{y} \leftarrow \mathsf{A}^{|\mathsf{H}(\cdot)\rangle}(\mathbf{A})\end{array}\right].$$

Note that the SelfTargetMSIS problem is classically at least as hard as MSIS [KLS18]. There is a (non-tight) reduction in the classical random-oracle model from the MSIS to the SelfTargetMSIS problem. Let A and B be the adversaries to the $\mathsf{MSIS}_{k,l,\zeta}$ and $\mathsf{SelfTargetMSIS}_{\mathsf{H},k,l+1,\zeta}$, respectively. If A only has classical access to H, then there is a reduction based on the forking lemma [PS00, BN06] to prove that $\mathrm{Adv}^{\mathsf{SelfTargetMSIS}}(\mathsf{B}) \approx \sqrt{\mathrm{Adv}^{\mathsf{MSIS}}(\mathsf{A})/Q_\mathsf{H}}$, where Q_H is

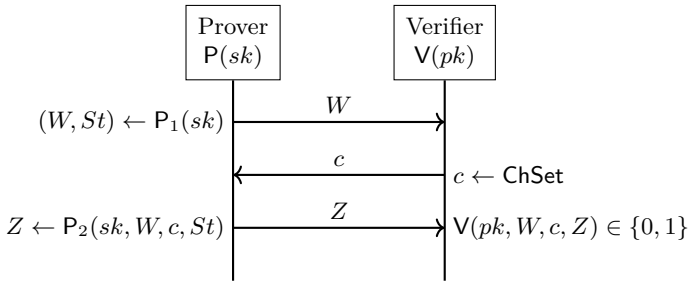

Fig. 1. A canonical identification scheme and its transcript (W, c, Z)

the number of classical queries to H. This reduction is standard and is implicit in the security proofs of digital signatures based on the hardness of the SIS problem (cf. [Lyu12, DKL+18]). We refer the readers to [KLS18] for the details.

2.4 Pseudorandom Functions

A pseudorandom function PRF is a mapping $\mathsf{PRF} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^k$, where $\mathcal{K}$ is a finite space for the key and n and k are integers. To a quantum adversary A and PRF we associate the advantage function

$$\mathrm{Adv}^{\mathsf{PR}}_{\mathsf{PRF}}(\mathsf{A}) := \left|\Pr[\mathsf{A}^{\mathsf{PRF}(K,\cdot)} \Rightarrow 1 | K \leftarrow \mathcal{K}] - \Pr[\mathsf{A}^{\mathsf{RF}(\cdot)} \Rightarrow 1]\right|,$$

where $\mathsf{RF} : \{0,1\}^n \to \{0,1\}^k$ is a perfect random function. Note that while adversary A is a quantum adversary, it only has classical access to the oracles $\mathsf{PRF}(K, \cdot)$ and $\mathsf{RF}(\cdot)$.

Extendable output function. We denote by Sam an extendable output function, that is a function on bit strings the output of which can be extended to arbitrary length. We write $y \sim S := \mathsf{Sam}(x)$ when Sam takes x as the input value and output y that is distributed according to the distribution S (or uniformly distributed over a set S). This procedure is deterministic: for a given x Sam will always output the same y. For simplicity we assume that distribution of the output of Sam is perfect, whereas Sam can be regarded as the random oracles and the output of which is statistically close to the perfect distribution. If K is a secret key, then $\mathsf{Sam}(K \,\|\, x)$ is a pseudorandom function from $\{0,1\}^* \to \{0,1\}^*$.

2.5 Identification Schemes

A canonical identification scheme ID is a three-move protocol as shown in Fig. 1. Firstly, the prover send a message W, which is called commitment, to the verifier. The verifier uniformly sample a challenge c from set ChSet and send it to the prover. The prover send a response Z, and then the verifier makes a deterministic decision.

$\underline{\mathsf{Trans}(sk)}$
1 $(W, St) \leftarrow \mathsf{P}_1(sk)$
2 $c \leftarrow \mathsf{ChSet}$
3 $Z \leftarrow \mathsf{P}_2(sk, W, c, St)$
4 **if** $Z = \perp$ **then return** $(\perp, \perp, \perp)$
5 **return** (W, c, Z)

Fig. 2. $\mathsf{Trans}(sk)$.

Definition 2.5 (Canonical Identification Scheme). *A canonical identification scheme* ID *is defined as a tuple of algorithms* $\mathsf{ID} := (\mathsf{IGen}, \mathsf{P}, \mathsf{ChSet}, \mathsf{V})$.

- *The key generation algorithm* IGen *takes system parameters* par *as input and returns a pair of public and secret keys* (pk, sk). *We assume that* pk *defines the set of challenges* ChSet, *the set of commitments* WSet, *and the set of responses* ZSet.
- *The prover algorithm* P *is a pair of two algorithms* $(\mathsf{P}_1, \mathsf{P}_2)$. P_1 *takes as input the secret key* sk *and returns a commitment* $W \in \mathsf{WSet}$ *and a state* St; P_2 *takes as input the secret key* sk, *a commitment* W, *a challenge* c, *and a state* St *and returns a response* $Z \in \mathsf{ZSet} \cup \{\perp\}$, *where* $\perp \notin \mathsf{ZSet}$ *is a special symbol indicating failure.*
- *The verifier algorithm* V *takes the public key* pk *and the conversation transcript* (W, c, Z) *as input and outputs a deterministic decision,* 1 *(acceptance) or* 0 *(rejection).*

We also define a transcript oracle Trans in Fig. 2 that returns a real interaction transcript, which is a three-tuple $(W, c, Z) \in \mathsf{WSet} \times \mathsf{ChSet} \times \mathsf{ZSet} \cup \{\perp, \perp, \perp\}$. between the prover and the verifier as depicted in Fig. 1, with the important convention that the transcript is defined as $(\perp, \perp, \perp)$ if $Z = \perp$. The transcript is called valid (with respect to public-key pk) if $\mathsf{V}(pk, W, c, Z) = 1$. We define no-abort honest-verifier zero-knowledge (naHVZK), which is a weak variant of honest-verifier zero-knowledge that requires the transcript to be publicly simulatable, conditioned on $Z \neq \perp$.

Definition 2.6 (naHVZK). *A canonical identification scheme* ID *is said to be* ϵ_{zk}*-perfect* naHVZK *if there exists an algorithm* Sim *that, given only the public key* pk, *outputs* (W, c, Z) *such that the following conditions hold:*

- *The distribution of* $(W, c, Z) \leftarrow \mathsf{Sim}(pk)$ *has statistical distance at most* ϵ_{zk} *from* $(W', c', Z') \leftarrow \mathsf{Trans}(sk)$, *where* Trans *is defined in Fig. 2.*
- *The distribution of* c *from* $(W, c, Z) \leftarrow \mathsf{Sim}(pk)$ *conditioned on* $c \neq \perp$ *is uniform random in* ChSet.

Definition 2.7 (Min-Entropy). *If the most likely value of a random variable* W *that is selected from a discrete distribution* D *occurs with probability* $2^{-\alpha}$, *then we say that* $H_\infty(W \mid W \leftarrow D) = \alpha$. *We say that a canonical identification*

GAMES UF-CMA/UF-CMA$_1$/UF-NMA:

1 $(pk, sk) \leftarrow \mathsf{KeyGen}(\mathsf{par})$
2 $(M^*, \varsigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}}(pk)$ // UF-CMA
3 $(M^*, \varsigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}_1}(pk)$ // UF-CMA$_1$
4 $(M^*, \varsigma^*) \leftarrow \mathsf{A}(pk)$ // UF-NMA
5 **return** $[\![M^* \notin \mathcal{M}]\!] \wedge \mathsf{Verify}(pk, M^*, \varsigma^*)$

SIGN(M)
6 $\mathcal{M} = \mathcal{M} \cup \{M\}$
7 $\varsigma \leftarrow \mathsf{Sign}(sk, M)$
8 **return** ς

SIGN$_1(M)$
9 **if** $M \in \mathcal{M}$ **then return** $\perp$
10 $\mathcal{M} = \mathcal{M} \cup \{M\}$
11 $\varsigma \leftarrow \mathsf{Sign}(sk, M)$
12 **return** ς

Fig. 3. Games UF-CMA and UF-NMA

scheme ID *has* α *bits of min-entropy, if*

$$\Pr_{(pk,sk)\leftarrow\mathsf{IGen}(\mathsf{par})}[H_\infty(W|(W, St) \leftarrow \mathsf{P}_1(sk)) \geq \alpha] \geq 1 - 2^{-\alpha}.$$

In other words, over the choice of (pk, sk), the min-entropy of W will be at least α, except with probability $2^{-\alpha}$. An identification scheme has *unique responses* if for all W and c there exists at most one Z such that $\mathsf{V}(pk, W, c, Z) = 1$. We relax this notion to a computational unique response (CUR): An identification scheme has CUR if it is computationally hard to find (W, c, Z, Z') with $\mathsf{V}(pk, W, c, Z) = \mathsf{V}(pk, W, c, Z') = 1$ and $Z' \neq Z$.

Definition 2.8 (Computational Unique Response). *To an adversary* A *we associate the advantage function*

$$\mathrm{Adv}^{\mathsf{CUR}}_{\mathsf{ID}}(\mathsf{A}) := \Pr\left[\begin{array}{l} V(pk, W, c, Z) = 1 \\ V(pk, W, c, Z') = 1 \\ Z \neq Z' \end{array} \middle| \begin{array}{l} (pk, sk) \leftarrow \mathsf{IGen}(\mathsf{par}); \\ (W, c, Z, Z') \leftarrow \mathsf{A}(pk) \end{array}\right].$$

2.6 Digital Signatures

We define the syntax and security of a digital signature scheme. Let par be public system parameters.

Definition 2.9 (Digital Signature). *A digital signature scheme* SIG *is defined as a triple of algorithms* SIG = (KeyGen, Sign, Verify). *The key generation algorithm* KeyGen(par) *returns the public and secret keys* (pk, sk). *We assume that* pk *defines* MSet *that is the space for the message* M. *The signing algorithm* $\mathsf{Sign}(sk, M)$ *returns a signature* ς. *The deterministic verification algorithm* $\mathsf{Verify}(pk, M, \varsigma)$ *returns* 1 *(accept) or* 0 *(reject).*

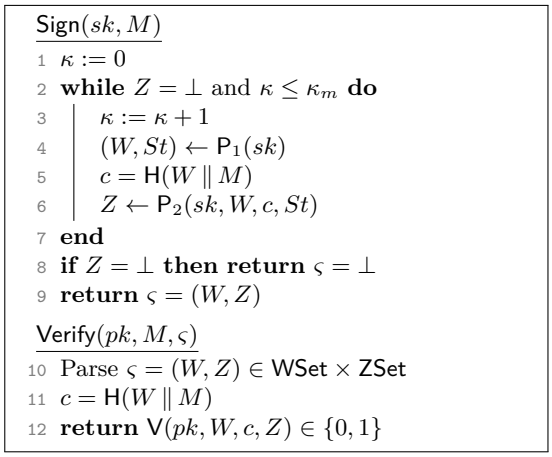

Fig. 4. Sign and Verify of the signature scheme $\mathsf{SIG} := (\mathsf{KeyGen} = \mathsf{IGen}, \mathsf{Sign}, \mathsf{Verify})$ obtained by the Fiat-Shamir transformation with aborts $\mathsf{FS}[\mathsf{ID}, \mathsf{H}, \kappa_m]$.

The signature scheme SIG has a correctness error γ if we have $\Pr[\mathsf{Verify}(pk, M, \mathsf{Sign}(sk, M)) = 0] \leq \gamma$ for all key pairs $(pk, sk) \in \mathsf{KeyGen}(\mathsf{par})$, and all messages $M \in \mathsf{MSet}$.

We define *unforgeability against chosen-message attack* (UF-CMA), *unforgeability against one-per-message chosen-message attack* ($\mathsf{UF\text{-}CMA}_1$), and *unforgeability against no-message attack* (UF-NMA) advantage functions of a (quantum) adversary A against SIG as $\mathrm{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SIG}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}CMA}^{\mathsf{A}} \Rightarrow 1]$, $\mathrm{Adv}^{\mathsf{UF\text{-}CMA}_1}_{\mathsf{SIG}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}CMA}_1^{\mathsf{A}} \Rightarrow 1]$, and $\mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{SIG}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}NMA}^{\mathsf{A}} \Rightarrow 1]$, where the games UF-CMA, $\mathsf{UF\text{-}CMA}_1$ and UF-NMA are shown in Fig. 3. We also consider *strong unforgeability against chosen-message attack* (SUF-CMA) and *strong unforgeability against one-per-message chosen-message attack* ($\mathsf{SUF\text{-}CMA}_1$), where the adversary may return a forgery on a message previously queried to the signing oracle, but with a different signature. In the corresponding experiments SUF-CMA and $\mathsf{SUF\text{-}CMA}_1$, the set $\mathcal{M}$ contains tuples (M, ς) and for the winning condition it is checked that $(M^*, \varsigma^*) \notin \mathcal{M}$.

2.7 Fiat-Shamir Signatures

Let $\mathsf{ID} := (\mathsf{IGen}, \mathsf{P}, \mathsf{ChSet}, \mathsf{V})$ be a canonical identification scheme, κ_m be a positive integer, and let $\mathsf{H} : \{0,1\}^* \to \mathsf{ChSet}$ be a hash function. The following signature scheme $\mathsf{SIG} := (\mathsf{KeyGen} = \mathsf{IGen}, \mathsf{Sign}, \mathsf{Verify})$ described in Fig. 4 is obtained by the Fiat-Shamir transformation with aborts $\mathsf{FS}[\mathsf{ID}, \mathsf{H}, \kappa_m]$ [Lyu09].

Kiltz *et al.* [KLS18] showed the generic framework for constructing tight reductions in the QROM from underlying hard problems to Fiat-Shamir signatures.

$\underline{\mathsf{DSign}(sk, M)}$
1 $\kappa := 0$
2 **while** $Z = \perp$ and $\kappa \leq \kappa_m$ **do**
3 $\quad \kappa := \kappa + 1$
4 $\quad (W, St) := \mathsf{P}_1(sk; \mathsf{PRF}_K(0 \,\|\, m \,\|\, \kappa))$
5 $\quad c = \mathsf{H}(W \,\|\, M)$
6 $\quad Z := \mathsf{P}_2(sk, W, c, St; \mathsf{PRF}_K(1 \,\|\, m \,\|\, \kappa))$
7 **end**
8 **if** $Z = \perp$ **then return** $\varsigma = \perp$
9 **return** $\varsigma = (W, Z)$

Fig. 5. DSign of the deterministic variant of the Fiat-Shamir signature $\mathsf{DFS}[\mathsf{ID}, \mathsf{H}, \mathsf{PRF}, \kappa_m]$

Theorem 2.10 ([KLS18], Theorem 3.2). *Assume the identification scheme* ID *is* ϵ_{zk}*-perfect* naHVZK *and has* α *bits of min entropy. For any* $\mathsf{UF\text{-}CMA}_1$ *(*$\mathsf{SUF\text{-}CMA}_1$*) quantum adversary* A *that issues at most* Q_H *queries to the quantum random oracle* $|\mathsf{H}\rangle$ *and* Q_S *(classical) queries to the signing oracle* SIGN_1*, there exists a quantum adversary* B *against* UF-NMA *security making* Q_H *queries to its own quantum random oracle (and a quantum adversary* C *against* CUR*) such that*

$$\mathrm{Adv}^{\mathsf{UF\text{-}CMA}_1}_{\mathsf{SIG}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{SIG}}(\mathsf{B}) + \kappa_m Q_S \cdot \epsilon_{\mathsf{zk}} + 2^{-\alpha+1},$$
$$\mathrm{Adv}^{\mathsf{SUF\text{-}CMA}_1}_{\mathsf{SIG}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{SIG}}(\mathsf{B}) + \kappa_m Q_S \cdot \epsilon_{\mathsf{zk}} + 2^{-\alpha+1} + \mathrm{Adv}^{\mathsf{CUR}}_{\mathsf{ID}}(\mathsf{C}),$$

and $\mathrm{Time}(\mathsf{B}) = \mathrm{Time}(\mathsf{C}) = \mathrm{Time}(\mathsf{A}) + \kappa_m(Q_H + Q_S) \simeq \mathrm{Time}(\mathsf{A})$.

Consider a deterministic variant $\mathsf{DSIG} := \mathsf{DFS}[\mathsf{ID}, \mathsf{H}, \mathsf{PRF}, \kappa_m] = (\mathsf{KeyGen}, \mathsf{DSign}, \mathsf{Verify})$ of FS where lines 4 and 6 Sign is replaced with deterministic PRF, where the key K is part of the secret key. We show the DFS in Fig. 5. The UF-CMA (SUF-CMA) security of DFS is implied by the $\mathsf{UF\text{-}CMA}_1$ ($\mathsf{SUF\text{-}CMA}_1$) security of FS. This construction is known in the classical setting [BPS16], and the same proof works in the quantum setting [KLS18]. Concretely, the advantages are upper bounded by the same terms as in Theorem 2.10 with an additional term $\mathrm{Adv}^{\mathsf{PR}}_{\mathsf{PRF}}(\mathsf{D})$ accounting for the quantum security of the PRF: We have

$$\mathrm{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SIG}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{SIG}}(\mathsf{B}) + \kappa_m Q_S \cdot \epsilon_{\mathsf{zk}} + 2^{-\alpha+1} + \mathrm{Adv}^{\mathsf{PR}}_{\mathsf{PRF}}(\mathsf{D}), \quad (1)$$
$$\mathrm{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{SIG}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{SIG}}(\mathsf{B}) + \kappa_m Q_S \cdot \epsilon_{\mathsf{zk}} + 2^{-\alpha+1} + \mathrm{Adv}^{\mathsf{PR}}_{\mathsf{PRF}}(\mathsf{D}) + \mathrm{Adv}^{\mathsf{CUR}}_{\mathsf{ID}}(\mathsf{C}). \quad (2)$$

$\underline{\mathsf{IGen}(\mathsf{par})}$

$\rho \xleftarrow{\$} \{0,1\}^{256}$, $\mathbf{A} \leftarrow R_q^{k\times l} := \mathsf{Sam}(\rho)$, $\mathbf{s}_1 \leftarrow S_\eta^l$,
$\mathbf{t} := \lfloor \frac{p}{q}\mathbf{A}\mathbf{s}_1 \rceil \in R_p^k$
$(\mathbf{t}_1, \mathbf{t}_0) := \mathsf{Decompose}_p(\mathbf{t}, 2^d)$
$pk := (\rho, \mathbf{t}_1, \mathbf{t}_0)$, $sk := (\rho, K, tr, \mathbf{s}_1, \mathbf{t}_0)$
return (pk, sk)

$\underline{\mathsf{V}(pk, W = \mathbf{w}_1, c, Z = (\mathbf{z}, \mathbf{h}))}$

return $[\![\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1]\!] \wedge [\![\mathbf{w}_1 = \mathsf{UseHint}_p(\mathbf{h}, \lfloor \frac{p}{q}\mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d \rceil, 2\overline{\gamma}_2)]\!]$

$\underline{\mathsf{P}_1(sk)}$

$\mathbf{A} \leftarrow R_q^{k\times l} := \mathsf{Sam}(\rho)$, $\mathbf{y} \leftarrow S_{\gamma_1 - 1}^l$
$\mathbf{w} := \lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil \in R_p^k$, $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil - \frac{p}{q}\mathbf{A}\mathbf{y}$
$\mathbf{w}_1 := \mathsf{HighBits}_p(\mathbf{w}, 2\overline{\gamma}_2)$
return $(W = \mathbf{w}_1, St = (\mathbf{w}, \boldsymbol{\xi}_1, \mathbf{y}))$

$\underline{\mathsf{P}_2(sk, (W = \mathbf{w}_1, c, St = (\mathbf{w}, \boldsymbol{\xi}_1, \mathbf{y}))}$

$\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$, $\mathbf{s}_2 := (\mathbf{t} - \frac{p}{q}\mathbf{A}\mathbf{s}_1)$
$\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$
$\boldsymbol{\xi}_2 := \lfloor c\mathbf{s}_2 \rceil - c\mathbf{s}_2$, $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_2 - \boldsymbol{\xi}_1 \rceil$
$\mathbf{r}_0 := \mathsf{LowBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)$
if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta_1$ or $\|\mathbf{r}_0\|_\infty \geq \overline{\gamma}_2 - \beta_2$ **then return** $Z = \perp$
$\mathbf{h} := \mathsf{MakeHint}_p(-c\mathbf{t}_0, \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0, 2\overline{\gamma}_2)$
return $Z = (\mathbf{z}, \mathbf{h})$

Fig. 6. Our identification scheme ID– a concrete instantiation based on the hardness of the MLWR problem of the canonical identification scheme in Fig. 1.

3 Our Identification Scheme

We show our identification scheme ID in Sect. 3.2, and we show our simple supporting algorithms for the bit decomposing technique in Sect. 3.1. Our scheme ID will be converted to our deterministic signature scheme MLWRSign in Sect. 4 with the deterministic Fiat-Shamir transform DFS.

3.1 Supporting Algorithms

We show in Fig. 7 the supporting algorithms for our identification scheme and MLWRSign, which are analogues of those of Dilithium. These algorithms are used for extracting higher-order and lower-order bits of elements in $\mathbb{Z}_q$, to decrease the size of the public key. In Dilithium, q is a prime and α is an even number so the algorithm Decompose has to consider the case when $r - r_0 = q - 1$. Since we use moduli q, p in the power of twos, our Decompose can be efficiently performed in a simpler bit-wise manner to break up an element.

The following lemmas state the properties of these supporting algorithms on which the correctness and security of our scheme is based. Since these lemmas are analogues of the Lemmas 1, 2, and 3 in [DLL+19], we omit their proofs.

Lemma 3.1 (Lemma 1 in [DLL+19]). *Suppose that p and α are positive integers such that $p > 2\alpha$, $p \equiv 0 \pmod{\alpha}$ and α even. Let $\mathbf{r}$ and $\mathbf{z}$ be vectors of elements in R_q where $\|\mathbf{z}\|_\infty \leq \alpha/2$, and let $\mathbf{h}, \mathbf{h}'$ be vectors of bits. Then the $\mathsf{HighBits}_p$, $\mathsf{MakeHint}_p$, and $\mathsf{UseHint}_p$ algorithms satisfy the following properties:*

1. $\mathsf{UseHint}_p(\mathsf{MakeHint}_p(\mathbf{z}, \mathbf{r}, \alpha), \mathbf{r}, \alpha) = \mathsf{HighBits}_p(\mathbf{r} + \mathbf{z}, \alpha)$.
2. *Let $\mathbf{v}_1 = \mathsf{UseHint}_p(\mathbf{h}, \mathbf{r}, \alpha)$. Then $\|\mathbf{r} - \mathbf{v}_1 \cdot \alpha\|_\infty \leq \alpha + 1$. Furthermore, if the number of 1s in $\mathbf{h}$ is ψ, then all except at most ψ coefficients of $\mathbf{r} - \mathbf{v}_1 \cdot \alpha$ will have a magnitude of at most $\alpha/2$ after centered reduction modulo q.*
3. *For any $\mathbf{h}$, $\mathbf{h}'$, if $\mathsf{UseHint}_p(\mathbf{h}, \mathbf{r}, \alpha) = \mathsf{UseHint}_p(\mathbf{h}', \mathbf{r}, \alpha)$, then $\mathbf{h} = \mathbf{h}'$.*

Lemma 3.2 (Lemma 2 in [DLL+19]). *If $\|\mathbf{s}\|_\infty \leq \beta$ and $\|\mathsf{LowBits}_p(\mathbf{r}, \alpha)\|_\infty < \alpha/2 - \beta$, then $\mathsf{HighBits}_p(\mathbf{r}, \alpha) = \mathsf{HighBits}_p(\mathbf{r} + \mathbf{s}, \alpha)$ holds.*

The function CRH is a collision resistant hash function that maps to $\{0,1\}^{384}$. The function Sam used in lines 2 and 11 is an extendable output function. In line 2 the function Sam maps a uniform seed $\rho \in \{0,1\}^{256}$ to a matrix $\mathbf{A} \in R^{k \times l}$. In line 11, Sam deterministically generates the randomness of the signature scheme, mapping a concatenation of K, μ and κ to $\mathbf{y} \in S^l_{\gamma_1 - 1}$.

3.2 Identification Scheme

We show our identification protocol $\mathsf{ID} = (\mathsf{IGen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ in Fig. 6, with the concrete parameters $\mathsf{par} = (q, n, k, l, d, \gamma_1, \gamma_2, \overline{\gamma}_1, \overline{\gamma}_2, \eta, \beta_1, \beta_1)$ given in Table 1.

Key generation. The key generation of ID proceeds by selecting a random 256-bit seed ρ and expanding into a matrix $\mathbf{A} \in R_q^{k \times l}$ by an extendable output function Sam modeled as a random oracle. The secret vector $\mathbf{s}_1 \in S^l_\eta$ has uniformly random coefficients in $[-\eta, \eta]$. The value $\mathbf{t} := \lfloor \frac{p}{q}\mathbf{A}\mathbf{s}_1 \rceil \in R_p^k$ is then computed. The public key that is required for verification is $(\rho, \mathbf{t}_1)$ with $\mathbf{t}_1$ output by the $(\mathbf{t}_1, \mathbf{t}_0) := \mathsf{Decompose}_p(\mathbf{t}, 2^d)$ while the secret key is $(\rho, \mathbf{s}_1, \mathbf{t}_0)$. While the verifier does not need the value $\mathbf{t}_0$ (and thus it is not needed to be included in the public key of $\mathsf{MLWRSign}$), we need to include this value to simulate transcripts (see Sect. 5.2). Thus the security of our scheme is constructed in the condition that the adversary gets both $\mathbf{t}_1$ and $\mathbf{t}_0$. In reality the adversary only gets $\mathbf{t}_1$, thus this is conservative condition. The set ChSet is defined as in (10), and $\mathsf{ZSet} = S^l_{\gamma_1 - \beta_1 - 1} \times \{0,1\}^k$. The set of commitments WSet is defined as $\mathsf{WSet} = \{\mathbf{w}_1 \mid \exists \mathbf{y} \in S_{\gamma_1 - 1} \text{ s.t. } \mathbf{w}_1 := \mathsf{HighBits}_p(\lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil, 2\overline{\gamma}_2)\}$.

Protocol execution. Our ID scheme is based on the canonical identification scheme in Fig. 1. The prover starts the identification protocol by $(W = \mathbf{w}_1, St = (\mathbf{w}, \boldsymbol{\xi}_1, \mathbf{y})) \leftarrow \mathsf{P}_1(sk)$ and sends $W = \mathbf{w}_1$ to the verifier, and then the verifier generates a random challenge $c \leftarrow \mathsf{ChSet}$ and sends it to the prover. The prover computes $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$ in line 12 and $\mathbf{r}_0 := \mathsf{LowBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)$ in line 14.

$\underline{\mathsf{Decompose}_p(r,\alpha)}$
1 $r := r \bmod^+ p$, $r_0 := r \bmod^\pm \alpha$, $r_1 := (r - r_0)/\alpha$,
2 **return** (r_1, r_0)
$\underline{\mathsf{HighBits}_p(r,\alpha)}$
3 $(r_1, r_0) := \mathsf{Decompose}_p(r,\alpha)$, **return** r_1
$\underline{\mathsf{LowBits}_p(r,\alpha)}$
4 $(r_1, r_0) := \mathsf{Decompose}_p(r,\alpha)$, **return** r_0
$\underline{\mathsf{MakeHint}_p(z,r,\alpha)}$
5 $r_1 := \mathsf{HighBits}_p(r,\alpha)$, $v_1 := \mathsf{HighBits}_p(r+z,\alpha)$
6 **return** $h := [\![r_1 \neq v_1]\!]$
$\underline{\mathsf{UseHint}_p(h,r,\alpha)}$
7 $m := p/\alpha$, $(r_1, r_0) := \mathsf{Decompose}_p(r,\alpha)$
8 **if** $h = 1$ and $r_0 > 0$ **then return** $(r_1 + 1) \bmod^+ m$
9 **if** $h = 1$ and $r_0 \leq 0$ **then return** $(r_1 - 1) \bmod^+ m$
10 **return** r_1

Fig. 7. Supporting algorithms for MLWRSign

He replies with $\perp$ if $\mathbf{z} \notin S^l_{\gamma_1-\beta_1-1}$ or $\mathbf{r}_0 \notin S^l_{\overline{\gamma}_2-\beta_2-1}$. This part of the protocol is necessary for security, it makes sure that $\mathbf{z}$ does not leak any information about the secret vectors $\mathbf{s}_1, \mathbf{s}_2$.

If the checks pass and a $\perp$ is not sent, then it can be shown (see Sect. 4.1) that $\mathsf{HighBits}_p(\lfloor \frac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}, 2\overline{\gamma}_2) = \mathbf{w}_1$. At this point, if the verifier had known the entire $\mathbf{t}$ and $(\mathbf{z}, c)$, he could have recovered $\mathbf{w}_1$ and checked that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$ and that the high-order bits of $\lfloor \frac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}$ are indeed $\mathbf{w}_1$. However, to compress the size of the public key, the verifier only knows $\mathbf{t}_1$. To allow the verifier to compute $\mathsf{HighBits}_p(\lfloor \frac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}, 2\overline{\gamma}_2)$ without $\mathbf{t}_0$, the signer needs to provide a hint vector $\mathbf{h}$. The verifier checks whether $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$ and that $\mathbf{w}_1$ can be reconstructed from $\lfloor \frac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}_1 \cdot 2^d$ and the hint $\mathbf{h}$.

4 Our Signature Scheme: MLWRSign

We present our scheme MLWRSign in Fig. 8, which is obtained by deterministic Fiat-Shamir transformation on the ID in Sect. 3. The correctness of MLWRSign is shown in Sect. 4.1. We analyze the probability of the rejection sampling of our signing procedure in Sect. 4.2. We explain the concrete settings of parameters in Sect. 4.3, and the values are shown in Table 1.

4.1 Correctness

We prove the correctness of our signature scheme in this subsection. If $\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2$, then by Lemma 3.1 we know that $\mathsf{UseHint}_p(\mathbf{h}, \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + c\mathbf{t}_0, 2\overline{\gamma}_2) =$

$\underline{\mathsf{KeyGen}(\mathsf{par})}$

$\rho \xleftarrow{\$} \{0,1\}^{256}$, $K \xleftarrow{\$} \{0,1\}^{256}$

$\mathbf{A} \in R_q^{k\times l} := \mathsf{Sam}(\rho)$, $\mathbf{s}_1 \leftarrow S_\eta^l$, $\mathbf{t} := \lfloor \frac{p}{q}\mathbf{A}\mathbf{s}_1 \rceil \in R_p^k$

$(\mathbf{t}_1, \mathbf{t}_0) := \mathsf{Decompose}_p(\mathbf{t}, 2^d)$, $tr := \mathsf{CRH}(\rho \,\|\, \mathbf{t}_1)$

return $(pk = (\rho, \mathbf{t}_1), sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{t}_0))$

$\underline{\mathsf{Sign}(pk, sk, M)}$

$\mathbf{A} \in R_q^{k\times l} := \mathsf{Sam}(\rho)$, $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$, $\mathbf{s}_2 := (\mathbf{t} - \frac{p}{q}\mathbf{A}\mathbf{s}_1)$

$\mu := \mathsf{CRH}(tr \,\|\, M)$, $\kappa := 0$

repeat

  repeat

    repeat

      $\kappa := \kappa + 1$

      $\mathbf{y} \in S_{\gamma_1-1}^l := \mathsf{Sam}(K \,\|\, \mu \,\|\, \kappa)$

      $\mathbf{w} := \lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil \in R_p^k$, $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil - \frac{p}{q}\mathbf{A}\mathbf{y}$

      $\mathbf{w}_1 := \mathsf{HighBits}_p(\mathbf{w}, 2\overline{\gamma}_2)$

      $c \in B_{60} := \mathsf{H}(\mu \,\|\, \mathbf{w}_1)$

      $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$

    until $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$

    $\boldsymbol{\xi}_2 := \lfloor c\mathbf{s}_2 \rceil - c\mathbf{s}_2$, $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_2 - \boldsymbol{\xi}_1 \rceil$

    $(\mathbf{r}_1, \mathbf{r}_0) := \mathsf{Decompose}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)$

  until $\|\mathbf{r}_0\|_\infty < \overline{\gamma}_2 - \beta_2$ and $\mathbf{r}_1 = \mathbf{w}_1$

  $\mathbf{h} := \mathsf{MakeHint}_p(-c\mathbf{t}_0, \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0, 2\overline{\gamma}_2)$

until $\mathrm{Hw}(\mathbf{h}) \le \omega$ and $\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2$

return $sig = (\mathbf{z}, \mathbf{h}, c)$

$\underline{\mathsf{Verify}(pk, sig, M)}$

$\mathbf{A} \in R_q^{k\times l} := \mathsf{Sam}(\rho)$, $\mu := \mathsf{CRH}(\mathsf{CRH}(pk) \,\|\, M)$

$\mathbf{w}_1' := \mathsf{UseHint}_p(\mathbf{h}, \lfloor \frac{p}{q}\mathbf{A}\mathbf{z} \rceil - c\mathbf{t}_1 \cdot 2^d, 2\overline{\gamma}_2)$, $c' := \mathtt{hash}(\mu \,\|\, \mathbf{w}_1')$

return $([\![\|\mathbf{z}\|_\infty < \gamma_1 - \beta]\!] \wedge [\![c = c']\!] \wedge [\![\mathrm{Hw}(\mathbf{h}) \le \omega]\!])$

Fig. 8. Our signature scheme MLWRSign

$\mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil, 2\overline{\gamma}_2)$. From the definitions of $\mathbf{w}$, $\mathbf{t}$, and $\mathbf{z}$, we obtain

$$\left\lfloor \frac{p}{q}\mathbf{A}\mathbf{z} \right\rceil - c\mathbf{t} = \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} \tag{3}$$

where $\mathbf{s}_2 = \lfloor \frac{p}{q}\mathbf{A}\mathbf{s}_1 \rceil - \frac{p}{q}\mathbf{A}\mathbf{s}_1$, $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{A}\mathbf{y} \rceil - \frac{p}{q}\mathbf{A}\mathbf{y}$, $\boldsymbol{\xi}_2 := \lfloor c\mathbf{s}_2 \rceil - c\mathbf{s}_2$ and $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_2 - \boldsymbol{\xi}_1 \rceil$.[3] Since $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ are polynomials whose coefficients are rounding errors that are heuristically i.i.d and uniformly distribute on $(-\frac{1}{2}, \frac{1}{2}]$, we have $\|\boldsymbol{\nu}\|_\infty \le 1$. Using $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$, we can rewrite (3) as $\lfloor \frac{p}{q}\mathbf{A}\mathbf{z} \rceil - c\mathbf{t}_1 \cdot 2^d = \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0$. Thus, the verifier computes $\mathbf{w}_1' = \mathsf{UseHint}_p(\mathbf{h}, \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0, 2\overline{\gamma}_2) =$

[3] Eq. (3) is corrected from [OTF+21, Eq. (3)], and the definition of $\boldsymbol{\nu}$ is different from that of [OTF+21]. See Appendix A for more details on corrections performed in this paper.

Table 1. Parameters for MLWRSign.

	I weak	II medium	III recomm.	IV high	V very high	VI paranoia
$(q,\, p)$	$(2^{23},\, 2^{19})$	$(2^{23},\, 2^{19})$	$(2^{23},\, 2^{20})$	$(2^{23},\, 2^{20})$	$(2^{23},\, 2^{21})$	$(2^{23},\, 2^{21})$
d	10	10	11	11	12	12
$(\gamma_1 = q/16,\, \overline{\gamma}_1 = \frac{p}{q}\gamma_1)$	$(2^{19},\, 2^{15})$	$(2^{19},\, 2^{15})$	$(2^{19},\, 2^{16})$	$(2^{19},\, 2^{16})$	$(2^{19},\, 2^{17})$	$(2^{19},\, 2^{17})$
$(\gamma_2 = \gamma_1/2,\, \overline{\gamma}_2 = \overline{\gamma}_1/2)$	$(2^{18},\, 2^{14})$	$(2^{18},\, 2^{14})$	$(2^{18},\, 2^{15})$	$(2^{18},\, 2^{15})$	$(2^{18},\, 2^{16})$	$(2^{18},\, 2^{16})$
$\eta = q/2p$	8	8	4	4	2	2
(k, l)	$(3, 2)$	$(4, 3)$	$(5, 4)$	$(6, 5)$	$(8, 7)$	$(9, 8)$
ω	64	80	96	112	144	160
$(\beta_1,\, \beta_2)$	(425, 25)	(425, 25)	(225, 25)	(225, 25)	(125, 25)	(125, 25)
# of 1 or -1 in c	60	60	60	60	60	60
BKZ block-size b to break MSIS	235	355	475	605	-	-
Core-Sieve bit-cost $2^{0.292b}$	68	103	138	176	-	-
Q-Core-Sieve bit-cost $2^{0.265b}$	62	94	125	160	-	-
BKZ block-size b to break MLWR	208	362	465	619	850	1002
Core-Sieve bit-cost $2^{0.292b}$	60	105	135	180	248	292
Q-Core-Sieve bit-cost $2^{0.265b}$	55	95	123	164	225	265

$\mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)$. Since the signer also checks that $\mathbf{r}_1 = \mathbf{w}_1$ in line 19, we obtain $\mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2) := \mathbf{r}_1 = \mathbf{w}_1$. Therefore, $\mathbf{w}'_1$ that the verifier computes is the same as $\mathbf{w}_1$ that the signer computes, and the verification procedure is always accepted.

4.2 Rejection Sampling

We analyze the probability of the rejection of our signing procedure in this subsection. Our analysis in this subsection for $P_3 := \Pr[\|\lfloor c\mathbf{s}_2 \rceil\|_\infty < \beta'_2]$ in (6), $P_4 := \Pr[\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2]$ in (7), and $P_5 := \Pr[\mathrm{Hw}(\mathbf{h}) < \omega]$ in (8) would also be helpful in analyzing the rejection sampling probability of the Dilithium in more detail. In [DKL+18], it was mentioned that it is difficult to formally compute the probability of the rejection of the Dilithium that corresponds to $1 - P_4 \cdot P_5$, and they heuristically selected parameters such that the probability become less than 1%. It was also mentioned in [DKL+18] that they chose the parameter such that the probability that corresponds to P_3 was higher than $1 - 2^{128}$ for the Dilithium, but its analysis was not shown.

We first calculate the probability of the rejection in line 16, i.e., we calculate $P_1 := \Pr[\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1]$. P_1 can be computed by considering each coefficient separately. For each coefficient σ of $c\mathbf{s}_1$, the corresponding coefficient of $\mathbf{z}$ will be in $(-\gamma_1 + \beta_1 + 1, \gamma_1 - \beta_1 - 1]$ whenever the corresponding coefficient of $\mathbf{y}_i$ is in $(-\gamma_1 + \beta_1 + 1 - \sigma, \gamma_1 - \beta_1 - 1 - \sigma)$. The size of this range is $2(\gamma_1 - \beta_1) - 1$, and the coefficients of $\mathbf{y}$ have $2\gamma_1 - 1$ possibilities since $\mathbf{y} \in S^l_{\gamma_1 - 1}$. Thus, we obtain $P_1 = \left(\frac{2(\gamma_1 - \beta_1) - 1}{2\gamma_1 - 1}\right)^{nl} = \left(1 - \frac{\beta_1}{\gamma_1 - 1/2}\right)^{nl}$. Thus, when γ_1 is large enough, we can

approximate

$$P_1 := \Pr[\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1] \simeq e^{-nl\beta_1/\gamma_1}. \tag{4}$$

Second, we calculate the probability of the rejection in line 19, i.e., $P_2 := \Pr[\|\mathbf{r}_0\|_\infty < \overline{\gamma}_2 - \beta_2]$. In a similar way to calculating (4), we obtain $P_2 = \left(\frac{2(\overline{\gamma}_2-\beta_2)-1}{2\overline{\gamma}_2}\right)^{nk} = \left(1 - \frac{\beta_2+1/2}{\overline{\gamma}_2}\right)^{nk}$. Therefore, when we assume that each coefficient of $\mathbf{r}_0$ is uniformly distributed modulo $2\gamma_2$, and $\overline{\gamma}_2$ is large enough and $\beta_2 \gg 1/2$, we can approximate

$$P_2 := \Pr[\|\mathbf{r}_0\|_\infty < \overline{\gamma}_2 - \beta_2] \simeq e^{-nk\beta_2/\overline{\gamma}_2}. \tag{5}$$

The check of $\mathbf{r}_1 := \mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2) = \mathsf{HighBits}_p(\mathbf{w}, 2\overline{\gamma}_2) := \mathbf{w}_1$ always succeeds if the condition $\|\lfloor c\mathbf{s}_2 \rceil - \boldsymbol{\nu}\|_\infty \leq \beta_2$ and $\|\mathbf{r}_0\|_\infty < \overline{\gamma}_2 - \beta_2$ holds, from Lemma 3.2. Since $\|\boldsymbol{\nu}\|_\infty \leq 1$ holds by definition, we have $\|\lfloor c\mathbf{s}_2 \rceil - \boldsymbol{\nu}\|_\infty \leq \|\lfloor c\mathbf{s}_2 \rceil\|_\infty + 1$.

In the following, we calculate $P_3 := \Pr[\|\lfloor c\mathbf{s}_2 \rceil\|_\infty < \beta_2']$, where $\beta_2' := \beta_2 - 1$, i.e., the probability that the check of $\mathbf{r}_1 = \mathbf{w}_1$ always succeeds. Let X_i be the i-th coefficient of an element of the vector $\mathbf{s}_2$, and let Y be a coefficient of an element of the vector $c\mathbf{s}_2$. Then, since $\mathbf{s}_2 \in S^k_{\frac{1}{2}}$, if we assume that $X_1 \dots X_n$ are i.i.d. and $X_i \sim \mathcal{U}(-\frac{1}{2}, \frac{1}{2})$, we can approximate that $Y \sim \mathcal{N}(0, 60\sigma_X^2)$ by the central limit theorem when n is large enough, where $\sigma_X^2 = \mathrm{Var}(X_i) = 1/12$. Thus, we can approximate $\Pr[|Y| < \beta_2'] \simeq 1 - 2F_{\mathcal{N}(0,5)}(-\beta_2')$, and then

$$P_3 := \Pr[\|\lfloor c\mathbf{s}_2 \rceil\|_\infty < \beta_2'] \simeq (1 - 2F_{\mathcal{N}(0,5)}(-\beta_2'))^{nk}, \tag{6}$$

where $F_{\mathcal{N}(0,5)}$ be the c.d.f of $\mathcal{N}(0,5)$. We set the parameter β_2 such that $\|\lfloor c\mathbf{s}_2 \rceil\|_\infty < \beta_2'$ holds with a probability higher than $1 - 2^{-30}$. Thus, the rejection probability in line 19 is dominated by P_2.

Finally, we calculate the probability of rejection in line 21, i.e., $P_4 := \Pr[\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2]$ and $P_5 := \Pr[\mathrm{Hw}(\mathbf{h}) < \omega]$. We first calculate P_4. By construction, $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$ and $\|\mathbf{t}_0\|_\infty \leq 2^{d-1}$. Let X_i be the i-th coefficient of an element of the vector $\mathbf{t}_0$, and let Y be the coefficient of an element of the vector $c\mathbf{t}_0$. Note that $c \in B_{60}$ so Y is the sum of 60 random elements of $\{X_i\}_{i=1}^n$. If we (heuristically) assume that $X_1 \dots X_n$ are i.i.d. and $X_i \sim \mathcal{U}(-2^{d-1}, 2^{d-1})$, we can approximate that $Y \sim \mathcal{N}(0, \sigma_Y^2)$ by the central limit theorem when n is large enough, where $\sigma_Y^2 := 60\sigma_X^2$ and $\sigma_X^2 = \mathrm{Var}(X_i) = (2 \cdot 2^{d-1})^2/12 = 2^{2d}/12$. Thus, we can approximate $\Pr[|Y| < \overline{\gamma}_2] \simeq 1 - 2F_{\mathcal{N}(0,\sigma_Y^2)}(-\overline{\gamma}_2)$, where $F_{\mathcal{N}(0,\sigma_Y^2)}$ is the c.d.f. of $\mathcal{N}(0, \sigma_Y^2)$. Since Y is the coefficient of an element of the vector in R_p^k, we obtain

$$P_4 := \Pr[\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2] \simeq (1 - 2F_{\mathcal{N}(0,\sigma_Y^2)}(-\overline{\gamma}_2))^{nk}. \tag{7}$$

We set the parameter $\overline{\gamma}_2$ so that $\|c\mathbf{t}_0\|_\infty < \overline{\gamma}_2$ holds with overwhelming probability. Also note that we set parameter d to satisfy $60 \cdot 2^{d-1} < 2\overline{\gamma}_2$ (as shown in Sect. 4.3) and the fact that $\|c\mathbf{t}_0\|_\infty \leq \|c\|_1 \cdot \|\mathbf{t}_0\|_\infty$. From these we obtain $\sigma_Y := \frac{1}{6\sqrt{5}} \cdot 60 \cdot 2^{d-1} < \frac{1}{3\sqrt{5}} \cdot \overline{\gamma}_2$, and approximately $\overline{\gamma}_2 > 6.7\sigma_Y$. Thus,

we can also estimate that $F_{\mathcal{N}(0,\sigma_Y^2)}(-\overline{\gamma}_2)$ is negligibly small, without numerical computation of $F_{\mathcal{N}(0,\sigma_Y^2)}(-\overline{\gamma}_2)$.

Next, we calculate $P_5 := \Pr[\mathrm{Hw}(\mathbf{h}) < \omega]$. Let X, Y and h be the coefficient of an element of the vector $\mathbf{r}_0$, $c\mathbf{t}_0$ and $\mathbf{h}$, respectively, and define $Z := X + Y$. Recall that

$$\mathbf{h} = [\![\mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0, 2\overline{\gamma}_2) \neq \mathsf{HighBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)]\!],$$

and $h = 1$ when the corresponding Z satisfies $|Z| > \overline{\gamma}_2$, $h = 0$ otherwise. We now calculate $\Pr[h = 1]$. In line 21, the conditions $\|\mathbf{r}_0\|_\infty < \overline{\gamma}_2 - \beta_2$ and $\|c\mathbf{t}_0\|_\infty \leq \overline{\gamma}_2$ are already satisfied. Thus, we assume that $X \sim \mathcal{U}(-(\overline{\gamma}_2 - \beta_2), (\overline{\gamma}_2 - \beta_2))$ as already derived, then we obtain

$$\begin{aligned} f_Z(z) &:= \int_{z-(\overline{\gamma}_2-\beta_2)}^{z+(\overline{\gamma}_2-\beta_2)} f_X(z-y) f_Y(y) dy \\ &= \frac{1}{2(\overline{\gamma}_2 - \beta_2)} \int_{z-(\overline{\gamma}_2-\beta_2)}^{z+(\overline{\gamma}_2-\beta_2)} f_Y(y) dy \\ &= \frac{1}{2(\overline{\gamma}_2 - \beta_2)} (F_Y(z + (\overline{\gamma}_2 - \beta_2)) - F_Y(z - (\overline{\gamma}_2 - \beta_2))), \end{aligned}$$

and $F_Z(z) = \int_{-\infty}^{z} f_Z(x)dx = \frac{1}{2(\overline{\gamma}_2-\beta_2)} \int_{z-(\overline{\gamma}_2-\beta_2)}^{z+(\overline{\gamma}_2-\beta_2)} F_Y(x)dx$, where f_X, f_Y and f_Z are the p.d.f. of the distribution of X, Y and Z, respectively. Then, we obtain

$$\Pr[h = 1] = \Pr[|Z| > \overline{\gamma}_2] = 2F_Z(-\overline{\gamma}_2) = \frac{1}{\overline{\gamma}_2 - \beta_2} \int_{-2(\overline{\gamma}_2-\beta_2)}^{0} F_Y(x)dx,$$

and thus we obtain $\mathrm{Hw}(\mathbf{h}) \sim \mathcal{B}(nk, \Pr[h = 1])$ since $\mathbf{h} \in R_p^k$. Because we can estimate that $Y \sim \mathcal{N}(0, \sigma_Y^2 = 5 \cdot 2^{2d})$ as we derived before, we obtain $P := \Pr[h = 1] \simeq \frac{1}{\overline{\gamma}_2-\beta_2} \int_{-2(\overline{\gamma}_2-\beta_2)}^{0} F_{\mathcal{N}(0,5\cdot 2^{2d})}(x)dx$. Therefore, let $F_{\mathcal{B}(nk,P)}$ be the c.d.f. of the binomial distribution $\mathcal{B}(nk, P)$, then we can approximate

$$P_5 := \Pr[\mathrm{Hw}(\mathbf{h}) < \omega] \simeq F_{\mathcal{B}(nk,P)}(\omega). \tag{8}$$

We set the parameter ω such that $\mathrm{Hw}(\mathbf{h}) < \omega$ with a probability higher than $1 - 2^{10}$.

To summarize, disregarding the conditions with overwhelming probability, i.e., assuming P_3, P_4, $P_5 \simeq 1$, we can estimate the probability of exiting the loop in lines 6 to 21 using (4) and (5) as follows:

$$P_1 \cdot P_2 \simeq e^{-n(\beta_1 l/\gamma_1 + \beta_2 k/\overline{\gamma}_2)}. \tag{9}$$

Thus, the expected number of iterations of the loop is $e^{n(\beta_1 l/\gamma_1 + \beta_2 k/\overline{\gamma}_2)}$.

4.3 Parameter Settings

We show our parameters in Table 1. In the following, we explain how we select these values.

Moduli q and p. We set $q = 2^{23}$ for all parameter sets of the security category. This value is the nearest power of two of 8380417 that is the value of the modulo q used in Dilithium. We set q and p as the power of twos to perform rounding by simple bit-shift operation, similar to the LWR-based PKE schemes Saber [DKRV18] and Round5 [BBF+19].

Module Dimensions (k, l) and Noise Parameter η. The parameter η corresponds to the standard deviation σ of the LWE problem. Dilithium bases its security on LWE with uniform distribution whose standard deviation is $\sigma = 2\eta/\sqrt{12}$, which is the standard deviation of the uniform distribution $\mathcal{U}(-\eta, \eta)$. For our scheme, the parameter η is defined by $\eta := \lceil \frac{q}{2p} \rceil = \frac{q}{2p}$. We estimate the bit-security based on the values of $\sigma = 2\eta/\sqrt{12}$, k, l, and n, using the lwe-estimator [ACD+18]. See Sect. 5.5 for details of the estimation of the bit-security. Note that η is also restricted to be the power of two since we set q, p as the power of twos. As a limitation, this setting loses a little flexibility to control the rejection rate and bit-security.

Space for challenge c. A cryptographic hash function that hashes onto B_{60} is used in Dilithium and our signature scheme. $B_h \subset R$ is a ring whose h coefficients are either -1 or 1 and the rest are 0. Thus, we obtain $|B_h| = 2^h \cdot \binom{n}{h}$, and then $|B_{60}| = 2^{60} \cdot \binom{256}{60} \simeq 2^{257.01} > 2^{256}$. Thus, let the space of challenge c in our scheme be ChSet, then we have

$$\mathsf{ChSet} := B_{60}, \text{ and } |\mathsf{ChSet}| > 2^{256}. \tag{10}$$

Setting of β_1, β_2. The parameters β_1 and β_2 are the counterpart of β used in Dilithium. In the scheme, the corresponding $\mathbf{s}_1$ and $\mathbf{s}_2$ are the variables that uniformly distribute on S_η, and β is selected such that $\|c\mathbf{s}_i\|_\infty < \beta$ for $i = 1, 2$ with overwhelming probability. Since $c \in B_{60}$, $\mathbf{s}_i \in S_\eta$, we obtain the bound $\|c\mathbf{s}_i\|_\infty \leq \|c\|_1 \cdot \|\mathbf{s}_i\|_\infty = 60\eta$, thus it can be seen that $\beta \leq 60\eta$. In MLWRSign, while we use the same $\mathbf{s}_1 \in S_\eta$ as Dilithium, $\mathbf{s}_2$ is a polynomial whose coefficients uniformly distribute on $(-\frac{1}{2}, \frac{1}{2}]$. Thus, we define the two parameters β_1 and β_2 such that $\|c\mathbf{s}_1\|_\infty < \beta_1$, $\|\lfloor c\mathbf{s}_2 \rceil\|_\infty < \beta_2 - 1 (< \beta_1)$ with overwhelming probability. This probability was analyzed in (6).

Setting of γ_1, γ_2, $\overline{\gamma_1}$, $\overline{\gamma}_2$. We set $\gamma_1 := q/16, \gamma_2 := \gamma_1/2$, $\overline{\gamma}_1 := \frac{p}{q}\gamma_1$, and $\overline{\gamma}_2 := \overline{\gamma}_1/2$. These parameters are related to the rejection rate of the signing and the security, which we describe in Sect. 5.4.

Setting of d. The parameter d defines the length of $\mathbf{t}_0$, which is part of the pk and sk (see also Fig. 10). We select d such that

$$60 \cdot (2^{d-1} + 1) < 2\overline{\gamma}_2 - 1 \tag{11}$$

for the security of our scheme, as discussed in Sect. 5.4. Here, $60 \cdot 2^{d-1}$ is the upper bound of $\|c\mathbf{t}_0\|_\infty$.

$\underline{\mathsf{Trans}(sk)}$

$\mathbf{A} \leftarrow R_q^{k\times l} := \mathsf{Sam}(\rho)$,
$\mathbf{y} \leftarrow S^l_{\gamma_1-1}$
$\mathbf{w} := \lfloor \frac{p}{q}\mathbf{Ay} \rceil \in R_p^k$, $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{Ay} \rceil - \frac{p}{q}\mathbf{Ay}$
$\mathbf{w}_1 := \mathsf{HighBits}_p(\mathbf{w}, 2\overline{\gamma}_2)$
$c \leftarrow \mathsf{ChSet}$
$\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$
$\mathbf{s}_2 := (\mathbf{t} - \frac{p}{q}\mathbf{As}_1)$
$\boldsymbol{\xi}_2 := \lfloor c\mathbf{s}_2 \rceil - c\mathbf{s}_2$, $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_2 - \boldsymbol{\xi}_1 \rceil$
if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta_1$ **then return** $\perp$
if $\|\mathsf{LowBits}_p(\mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}, 2\overline{\gamma}_2)\|_\infty \geq \overline{\gamma}_2 - \beta_2$ **then return** $\perp$
$\mathbf{h} := \mathsf{MakeHint}_p(-c\mathbf{t}_0, \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu} + c\mathbf{t}_0, 2\overline{\gamma}_2)$
return $(c, (\mathbf{z}, \mathbf{h}))$

$\underline{\mathsf{Sim}(pk)}$

$\mathbf{A} \leftarrow R_q^{k\times l} := \mathsf{Sam}(\rho)$,
With probability $1 - \frac{\left|S^l_{\gamma_1-\beta_1-1}\right|}{\left|S^l_{\gamma_1-1}\right|}$, **return** $\perp$
$c \leftarrow \mathsf{ChSet}$
$\mathbf{z} \leftarrow S^l_{\gamma_1-\beta_1-1}$
if $\|\mathsf{LowBits}_p(\lfloor \frac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}, 2\overline{\gamma}_2)\|_\infty \geq \overline{\gamma}_2 - \beta_2$ **then return** $\perp$
$\mathbf{h} := \mathsf{MakeHint}_p(-c\mathbf{t}_0, \frac{p}{q}\lfloor \mathbf{Az} \rceil - c\mathbf{t} + c\mathbf{t}_0, 2\overline{\gamma}_2)$
return $(c, (\mathbf{z}, \mathbf{h}))$

Fig. 9. Left: a real transcript output by the transcript algorithm $\mathsf{Trans}(sk)$; Right: a simulated transcript output by the $\mathsf{Sim}(pk)$ algorithm.

5 Security

The goal of this section is to provide full proof for the tight security reduction for MLWRSign in the QROM from the MLWR, SelfTargetMSIS, and MSIS, which is the following theorem:

Theorem 5.1 (QROM security of MLWRSign). *For any quantum adversary A against* SUF-CMA *security that issues at most Q_{H} queries to the quantum random oracle $|\mathsf{H}\rangle$, there exist quantum adversaries B, C, D, and E such that*

$$\begin{aligned}\mathrm{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{MLWRSign}}(\mathsf{A}) \leq\ & \mathrm{Adv}^{\mathsf{MLWR}}_{p,k,l,D}(\mathsf{B}) + \mathrm{Adv}^{\mathsf{SelfTargetMSIS}}_{\mathsf{H},k,l+1,\zeta}(\mathsf{C}) \\ & + \mathrm{Adv}^{\mathsf{PR}}_{\mathsf{Sam}}(\mathsf{D}) + \mathrm{Adv}^{\mathsf{MSIS}}_{k,l,\zeta'}(\mathsf{E}) + 2^{-\alpha+1}, \end{aligned} \tag{12}$$

where D is a uniform distribution over S_η, α is bits of min-entropy of the identification scheme ID *shown in Fig. 6, and ζ and ζ' are defined as follows:*

$$\zeta = \max\{\gamma_1 - \beta_1, \tfrac{q}{p}(2\overline{\gamma}_2 + 1 + 60 \cdot 2^{d-1})\} \leq 4\gamma_2, \tag{13}$$

$$\zeta' = \max\{2(\gamma_1 - \beta_1), 4\gamma_2 + 2\} \leq 4\gamma_2 + 6\eta. \tag{14}$$

We obtain the bound of (12) based on Theorem 2.10, and equations (1) and (2). The proof of this is modular. We constructed in Sect. 3.2 the identification scheme ID from which we obtain MLWRSign via the (deterministic) Fiat-Shamir transform, i.e., ID satisfies MLWRSign = DFS[ID, H, PRF, κ_m]. We show the following properties of ID in the rest of this section:

- ID has α bits of min-entropy, where $\alpha \geq 90$, 180, 255, 255,255 and 255, for parameter sets in Table 1 (I), (II), (III), (IV), (V), and (VI), respectively. (Sect. 5.1)
- ID is perfectly naHVZK, i.e. ϵ_{zk}-perfect naHVZK for $\epsilon_{\mathsf{zk}} = 0$ (Sect. 5.2)
- $\mathrm{Adv}^{\mathsf{CUR}}_{\mathsf{ID}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{MSIS}}_{k,l,\zeta'}(\mathsf{E})$ (Sect. 5.3)
- UF-NMA security of MLWRSign (Sect. 5.4)

Combining all of these properties, we can apply Theorem 2.10 to MLWRSign and we obtain (12). In Sect. 5.5 we show how we derived concrete bit-security shown in Table 1 based on (12).

5.1 Min-entropy

Lemma 5.2. *For a fixed matrix* $\mathbf{A} \leftarrow R_q^{k\times l}$ *and* $\mathbf{w}_1$*, let*

$$P_{\mathbf{A},\mathbf{w}_1} := \Pr_{\mathbf{y}\leftarrow S^l_{\gamma_1-1}}[\mathsf{HighBits}_p(\lfloor \tfrac{p}{q}\mathbf{A}\mathbf{y}\rceil, 2\overline{\gamma}_2) = \mathbf{w}_1] \tag{15}$$

Then,

$$\Pr_{\mathbf{A}\leftarrow R_q^{k\times l}}\left[\forall \mathbf{w}_1 : P_{\mathbf{A},\mathbf{w}_1} \leq \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^n\right] > 1-(n/q)^{kl}. \tag{16}$$

Proof. The probability that a random polynomial $a \leftarrow R_q$ is invertible in $R_q = \mathbb{Z}_q[X]/(X^n+1)$ when the polynomial X^n+1 splits into n linear factors is $(1-1/q)^n > 1-n/q$. Thus the probability that at least one of the kl polynomials in $\mathbf{A} \leftarrow R_q^{k\times l}$ is invertible is greater than $1-(n/q)^{kl}$.

We will now prove that for all $\mathbf{A}$ that contain at least one invertible polynomial, we will have that for all $\mathbf{w}_1$, $P_{\mathbf{A},\mathbf{w}_1} \leq \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^n$, which will prove the lemma. Let us only consider the row of $\mathbf{A}$ which contains the invertible polynomial. Denote the elements in this row by $[a_1,\ldots,a_l]$ and without loss of generality assume that a_1 is invertible. We want to prove that for all w_1 (element of $\mathbf{w}_1$),

$$\Pr_{\mathbf{y}\leftarrow S^l_{\gamma_1-1}}[\mathsf{HighBits}_p(\lfloor \tfrac{p}{q}\textstyle\sum_{i=1}^l a_i y_i\rceil, 2\overline{\gamma}_2) = w_1] \leq \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^n.$$

Let us define $T := \{w \mid \mathsf{HighBits}_p(w, 2\overline{\gamma}_2) = w_1\}$. By the definition of the $\mathsf{Decompose}_p$ routine in Fig. 7, the size of T is at most $(2\overline{\gamma}_2+1)^n$. We can then rewrite the above probability as

$$\Pr_{\mathbf{y}\leftarrow S^l_{\gamma_1-1}}[\lfloor \tfrac{p}{q}\textstyle\sum_{i=1}^l a_i y_i\rceil \in T] = \Pr_{\mathbf{y}\leftarrow S^l_{\gamma_1-1}}[y_i \in a_i^{-1}(\tfrac{q}{p}(T-\xi) - \textstyle\sum_{i=2}^l a_i y_i)]$$

where ξ is a rounding error defined as $\xi := \lfloor \frac{p}{q} \sum_{i=1}^{l} a_i y_i \rceil - (\frac{p}{q} \sum_{i=1}^{l} a_i y_i)$. The size of the set $a_i^{-1}(\frac{q}{p}(T-\xi) - \sum_{i=2}^{l} a_i y_i)$ is at most $(2\frac{q}{p}\overline{\gamma}_2 + 1)^n = (2\gamma_2 + 1)^n$, and the size of the set $S^l_{\gamma_1 - 1}$ is exactly $(2\gamma_1 - 1)^n$, thus we have

$$\Pr_{\mathbf{y} \leftarrow S^l_{\gamma_1 - 1}} [\lfloor \tfrac{p}{q} \textstyle\sum_{i=1}^{l} a_i y_i \rceil \in T] = \left(\frac{2\gamma_2 + 1}{2\gamma_1 - 1}\right)^n. \qquad \square$$

For the values in Table 1, we have that $\left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^n < 2^{-255}$ for every parameter sets and $(n/q)^{kl} = 2^{-90}$, 2^{-180}, 2^{-300}, 2^{-450}, 2^{-840} and 2^{-1080} for parameter sets (I), (II), (III), (IV), (V), and (VI), respectively. Thus, by Definition 2.7, the min-entropy of MLWRSign for parameter sets (I), (II), (III), (IV), (V), and (VI), is greater than 90, 180, 255, 255, 255, and 255, respectively.

It is important to note here that the real min-entropy should be a lot higher since the $\mathsf{HighBits}_p$ function maps onto a set of size larger than 25000 and is heuristically close to uniform over this set. To get a formal proof would be significantly more involved than the proof above which took advantage of the fact that $\gamma_1 = 2\gamma_2$, and gave us a sufficiently high min-entropy bound for practical purposes.

5.2 Non Abort Honest Verifier Zero-Knowledge

In this section, we show that ID is perfect naHVZK (Definition 2.6), in other words, we show that the distribution of the output of the Trans algorithm (Fig. 9, left) that takes the secret key as input is exactly the same as that of the Sim algorithm (Fig. 9, right) that takes only the public key as input.

Lemma 5.3. *If* $\beta_1 \geq \max_{\mathbf{s}_1 \in S_\eta, c \in \mathsf{ChSet}} \|c\mathbf{s}_1\|_\infty$, *then* ID *in Fig. 6 is perfectly* naHVZK.

Proof. Let $\mathbf{s}_1 \in S^l_\eta$ be any polynomials satisfying $\frac{p}{q}\lfloor \mathbf{A}\mathbf{s}_1 \rceil = \mathbf{t}$. We show that the output distributions of Trans and Sim from Fig. 9 are identical. For any $\mathbf{z} \in S^l_{\gamma_1 - \beta_1 - 1}$, we compute the probability that $\mathbf{z}$ is generated in line 6 of Trans. For any $c \in \mathsf{ChSet}$, we have

$$\Pr_{\mathbf{y} \leftarrow S^l_{\gamma_1 - 1}} [\mathbf{y} + c\mathbf{s}_1 = \mathbf{z}] = \Pr_{\mathbf{y} \leftarrow S^l_{\gamma_1 - 1}} [\mathbf{y} = \mathbf{z} - c\mathbf{s}_1]. \tag{17}$$

Because $\|c\mathbf{s}_1\|_\infty \leq \beta_1$, we know $\mathbf{z} - c\mathbf{s}_1 \in S^l_{\gamma_1 - 1}$. Thus,

$$\Pr_{\mathbf{y} \leftarrow S^l_{\gamma_1 - 1}} [\mathbf{y} = \mathbf{z} - c\mathbf{s}_1] = 1/|S^l_{\gamma_1 - 1}|. \tag{18}$$

Therefore, every $\mathbf{z} \in S^l_{\gamma_1 - \beta_1 - 1}$ has an equal probability of being generated. Furthermore, the probability of producing a $\mathbf{z} \in S^l_{\gamma_1 - \beta_1 - 1}$, which equals the probability of not returning $\perp$ in line 9 of Trans, is exactly $\frac{|S^l_{\gamma_1 - \beta_1 - 1}|}{|S^l_{\gamma_1 - 1}|}$. Thus, after line 9, either $\perp$ has been returned (with probability $1 - \frac{|S^l_{\gamma_1 - \beta_1 - 1}|}{|S^l_{\gamma_1 - 1}|}$), or the

distribution of $(c, \mathbf{z})$ is uniform in $\mathsf{ChSet} \times S^l_{\gamma_1-\beta_1-1}$. This is exactly the same distribution as that after line 15 of Sim. To complete the proof, we note that $\left\lfloor \frac{p}{q}\mathbf{Az} \right\rceil - c\mathbf{t} = \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}$ holds from (3), thus all the steps in Trans after line 9 are identical to those after line 15 of Sim. □

5.3 Computational Unique Response

In this section we prove that our ID satisfies the CUR property defined in Definition 2.8 required for strong-unforgeability of the signature scheme. The following Lemma 5.4 directly implies that $\mathrm{Adv}^{\mathsf{CUR}}_{\mathsf{ID}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{MSIS}}_{k,l,\zeta'}(\mathsf{E})$ for ζ' defined in (14).

Lemma 5.4. *If there exist* $(\mathbf{w}_1, c, (\mathbf{z}, \mathbf{h}))$ *and* $(\mathbf{w}_1, c, (\mathbf{z}', \mathbf{h}'))$ *such that* $\mathsf{V}(pk, \mathbf{w}_1, c, (\mathbf{z}, \mathbf{h})) = \mathsf{V}(pk, \mathbf{w}_1, c, (\mathbf{z}', \mathbf{h}')) = 1$ *and* $(\mathbf{z}, \mathbf{h}) \neq (\mathbf{z}', \mathbf{h}')$*, then there exist* $\mathbf{v}, \mathbf{u}$ *such that* $\|\mathbf{v}\|_\infty < 2(\gamma_1 - \beta_1)$*,* $\|\mathbf{u}\|_\infty \leq 4\gamma_2 + 6\eta$*, and* $\mathbf{Av} + \mathbf{u} = 0$*.*

Proof. The two conditions of the Lemma imply that

$$\mathbf{w}_1 = \mathsf{UseHint}_p(\mathbf{h}, \lfloor \tfrac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}_1 \cdot 2^d, 2\overline{\gamma}_2), \text{ and}$$
$$\mathbf{w}_1 = \mathsf{UseHint}_p(\mathbf{h}', \lfloor \tfrac{p}{q}\mathbf{Az}' \rceil - c\mathbf{t}_1 \cdot 2^d, 2\overline{\gamma}_2).$$

We first point out that it must be that $\mathbf{z} \neq \mathbf{z}'$. This is because Lemma 3.1 implies that if $\mathbf{z} = \mathbf{z}'$ then necessarily $\mathbf{h} = \mathbf{h}'$ (and then $Z = Z'$). The above two equations imply (again by Lemma 3.1) that

$$\|\lfloor \tfrac{p}{q}\mathbf{Az} \rceil - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\overline{\gamma}_2\|_\infty \leq 2\overline{\gamma}_2 + 1, \tag{19}$$
$$\|\lfloor \tfrac{p}{q}\mathbf{Az}' \rceil - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\overline{\gamma}_2\|_\infty \leq 2\overline{\gamma}_2 + 1. \tag{20}$$

We have $\mathbf{u} := \lfloor \frac{p}{q}\mathbf{Az} \rceil - \lfloor \frac{p}{q}\mathbf{Az}' \rceil = \lfloor \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil + \boldsymbol{\nu}$ where $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{Az} \rceil - \frac{p}{q}\mathbf{Az}$, $\boldsymbol{\xi}_2 := \lfloor \frac{p}{q}\mathbf{Az}' \rceil - \frac{p}{q}\mathbf{Az}'$, $\boldsymbol{\xi}_3 := \lfloor \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil - \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}')$, and $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 - \boldsymbol{\xi}_3 \rceil$. From (19) and (20), we have $\|\mathbf{u}\|_\infty \leq 4\overline{\gamma}_2 + 2$ by the triangular inequality. Hence, we have $\|\mathbf{u} - \boldsymbol{\nu}\|_\infty \leq 4\overline{\gamma}_2 + 3$ since $\|\boldsymbol{\nu}\|_\infty \leq 1$. Thus, $\lfloor \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil + \mathbf{u}' = 0$ for some $\mathbf{u}'$ such that $\|\mathbf{u}'\|_\infty \leq 4\overline{\gamma}_2 + 3$ and $\|\mathbf{z} - \mathbf{z}'\|_\infty < 2(\gamma_1 - \beta_1)$. Furthermore, we can rewrite $\mathbf{A}(\mathbf{z} - \mathbf{z}') + \mathbf{u}'' = 0$ such that $\|\mathbf{u}''\|_\infty \leq \frac{q}{p}(4\overline{\gamma}_2 + 3) = 4\gamma_2 + 6\eta$. □

5.4 UF-NMA Security

In this section we show UF-NMA security of MLWRSign.

Theorem 5.5 (UF-NMA security of MLWRSign). *Let* q, p *be positive integers such that* $q > p \geq 2$ *and* $p \mid q$*. For any quantum adversary* A *against* UF-NMA *security that issues at most* Q_H *queries to the quantum random oracle* $|\mathsf{H}\rangle$*, there exist quantum adversaries* B *and* C *such that*

$$\mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{MLWRSign}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{MLWR}}_{p,k,l,D}(\mathsf{B}) + \mathrm{Adv}^{\mathsf{SelfTargetMSIS}}_{\mathsf{H},k,l+1,\zeta}(\mathsf{C}), \tag{21}$$

and $\mathrm{Time}(\mathsf{B}) = \mathrm{Time}(\mathsf{C}) = \mathrm{Time}(\mathsf{A}) + Q_H$*, where* D *is the uniform distribution over* S_η*, and* ζ *is defined as in* (13)*.*

Proof. The adversary C obtains $\left[\mathbf{A} \mid \mathbf{t}'\right] \in R_q^{k\times(l+1)}$, which is an instance of $\mathsf{SelfTargetMSIS}_{\mathsf{H},k,l+1,\zeta}$, and decompose $\mathbf{t}'$ as $\mathbf{t}' := \frac{q}{p}\mathbf{t} + \mathbf{v}$, where $\mathbf{t} \in R_p^k$ is the higher $\log p$ bits of $\mathbf{t}'$ and $\mathbf{v} \in R_{q/p}^k$ is the lower $(\log q - \log p)$ bits. Note that $\mathbf{t}$ is uniformly random in R_p^k because $\mathbf{t}'$ is uniformly random in R_q^k. Then, C sets $(\mathbf{A}, \mathbf{t})$ as the public key of the signature scheme and sends it to A. The public key pk generated by IGen is indistinguishable from uniform over $R_q^{k\times l} \times R_p^k$ except with the probability $\mathrm{Adv}^{\mathsf{MLWR}}_{p,k,l,D}(\mathsf{B})$. Thus, with probability $\mathrm{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{MLWRSign}}(\mathsf{A}) - \mathrm{Adv}^{\mathsf{MLWR}}_{p,k,l,D}(\mathsf{B})$, A will return a signature $(c, (\mathbf{z}, \mathbf{h}))$ of some message μ such that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$ satisfies the verification equation

$$c = \mathsf{H}(\mu \,\|\, \mathsf{UseHint}_p(\mathbf{h}, \lfloor \tfrac{p}{q}\mathbf{A}\mathbf{z} \rceil - c\mathbf{t}_1 \cdot 2^d, 2\overline{\gamma}_2)).$$

From Lemma 3.1 we can write

$$2\overline{\gamma}_2 \cdot \mathsf{UseHint}_p(\mathbf{h}, \lceil \tfrac{p}{q}\mathbf{A}\mathbf{z} \rfloor - c\mathbf{t}_1 \cdot 2^d, 2\overline{\gamma}_2) = \lceil \tfrac{p}{q}\mathbf{A}\mathbf{z} \rfloor - c\mathbf{t}_1 \cdot 2^d + \mathbf{u},$$

where, $\|\mathbf{u}\|_\infty \le 2\overline{\gamma}_2 + 1$. Since $\lceil \frac{p}{q}\mathbf{A}\mathbf{s}_1 \rfloor = \mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$ and $\|\mathbf{t}_0\|_\infty \le 2^{d-1}$, we can rewrite

$$\begin{aligned}\lceil \tfrac{p}{q}\mathbf{A}\mathbf{z} \rfloor - c\mathbf{t}_1 \cdot 2^d + \mathbf{u} &= \tfrac{p}{q}\mathbf{A}\mathbf{z} + \boldsymbol{\xi} - c\mathbf{t} + c\mathbf{t}_0 - c\tfrac{p}{q}\mathbf{v} + c\tfrac{p}{q}\mathbf{v} + \mathbf{u} \\ &= \tfrac{p}{q}\left[\mathbf{A} \mid \tfrac{q}{p}\mathbf{t} + \mathbf{v} \mid \mathbf{I}_k\right]\begin{bmatrix}\mathbf{z} \\ -c \\ \frac{q}{p}\mathbf{u}'\end{bmatrix},\end{aligned}$$

where $\mathbf{u}' := (c(\mathbf{t}_0 + \frac{p}{q}\mathbf{v}) + \mathbf{u} + \boldsymbol{\xi})$, $\boldsymbol{\xi} := \lceil \frac{p}{q}\mathbf{A}\mathbf{z} \rfloor - \frac{p}{q}\mathbf{A}\mathbf{z}$. Since $\mathbf{v}$ is a random vector uniformly distributed on $R_{q/p}^k$, the upper-bound for $\|\mathbf{u}'\|_\infty$ is given as

$$\begin{aligned}\|\mathbf{u}'\|_\infty &\le \|c\|_1 \cdot \|\mathbf{t}_0 + \tfrac{p}{q}\mathbf{v}\|_\infty + \|\mathbf{u} + \boldsymbol{\xi}\|_\infty \\ &\le 60 \cdot (2^{d-1} + 1) + 2\overline{\gamma}_2 + 1 < 4\overline{\gamma}_2 = 4\tfrac{p}{q}\gamma_2.\end{aligned}$$

Note that we select d such that $60 \cdot (2^{d-1} + 1) < 2\overline{\gamma}_2 - 1$, as we described in Sect. 4.3 (see also Table 1). Thus, the adversary A can find $(\mathbf{z}, c, \mathbf{u}')$ and $\mu \in \{0,1\}^*$ such that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$, $\|c\|_\infty = 1$, $\|\mathbf{u}'\|_\infty < 4\frac{p}{q}\gamma_2$ and

$$\mathsf{H}'\left(\mu \,\|\, \left[\mathbf{A} \mid \mathbf{t}' \mid \mathbf{I}_k\right]\begin{bmatrix}\mathbf{z} \\ -c \\ \frac{q}{p}\mathbf{u}'\end{bmatrix}\right) = c, \tag{22}$$

where $\mathsf{H}'(\mu \,\|\, x) = \mathsf{H}(\mu \,\|\, \frac{1}{2\gamma_2}x)$, and $\mathbf{t}' := \frac{q}{p}\mathbf{t} + \mathbf{v}$ by definition. Since $\mathbf{A} \in R_q^{k\times l}$ and $\mathbf{t}' \in R_q^k$ are random, $\mathbf{y} := \left[\mathbf{z}^\intercal \mid -c \mid \frac{q}{p}\mathbf{u}'^\intercal\right]^\intercal$ is a solution to $\mathsf{SelfTargetMSIS}_{\mathsf{H}',k,l+1,\zeta}$ defined in Definition 2.4, where $\zeta = \max\{\|\mathbf{z}\|_\infty, \|\frac{q}{p}\mathbf{u}'\|_\infty\} \le 4\gamma_2$ as shown in (13). □

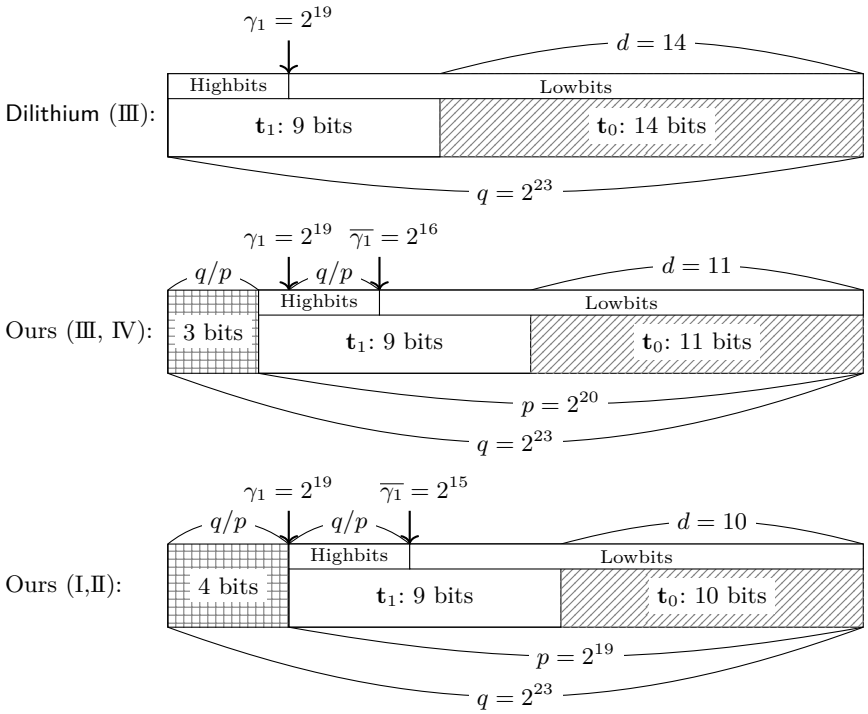

Fig. 10. Illustration of the bit length of $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$ (pk part: $\mathbf{t}_1$, sk part: $\mathbf{t}_0$)

5.5 Concrete Security

We follow the methodology of [DLL+19] to derive the security parameters in Table 1 with minor adaptations considering the MLWR problem. Since there are no known attacks that benefit the module structure, we view MLWR and MSIS problems as the LWR and SIS problems. The LWR and SIS problems are exactly the same as those in the definitions of MLWR and MSIS in Sect. 2.3 with the ring R_q being replaced by $\mathbb{Z}_q$.

Concrete hardness of $\mathsf{MLWR}_{p,k,l,D}$. We can view an $\mathsf{MLWR}_{p,k,l,D}$ instance as an LWR instance of dimensions $256l$ and $256k$: we can rewrite $\mathsf{MLWR}_{p,k,l,D}$ as finding $vec(\mathbf{s}_1) \in \mathbb{Z}^{256l} \times \mathbb{Z}^{256k}$ from $(\mathrm{rot}(\mathbf{A}), vec(\mathbf{t}))$, where $vec(\cdot)$ maps a vector of R_q to the vector obtained by concatenating the coefficients of its coordinates, and $\mathrm{rot}(\mathbf{A}) \in \mathbb{Z}_q^{256k \times 256l}$ is obtained by replacing all entries $a \in R$ of $\mathbf{A}$ by the 256×256 matrix whose z-th column is $vec(x^{z-1} \cdot a_{ij})$. Given an LWR instance $(\mathbf{A}, \mathbf{t} := \lfloor \frac{p}{q}\mathbf{As} \rceil)$, we convert it to a LWE instance $(\mathbf{A}, \frac{q}{p}\mathbf{t} = \mathbf{As} + \frac{q}{p}\boldsymbol{\xi})$, where $\boldsymbol{\xi} := \lfloor \frac{p}{q}\mathbf{As} \rceil - \frac{p}{q}\mathbf{As}$ is a vector of rounding error uniformly distributed over $(-\frac{1}{2}, \frac{1}{2})$. Thus, we obtain the variance of noise of the converted LWE sample as

$\sigma^2 = \frac{q^2}{12p^2}$, and we estimate the concrete hardness (BKZ block size b) based on the value of $256l, q$ and σ using the lwe-estimator [ACD+18].

Concrete hardness of $\mathsf{SelfTargetMSIS}_{\mathsf{H},k,l+1,\zeta}$. It is shown in [DLL+19] that, by using a standard forking lemma argument, an adversary to solve the above problem in the random oracle model can solve the MSIS problem. As discussed in the paper, since the reduction using the forking lemma lacks tightness, our scheme also relies on the exact hardness of analogues of the problem of (22). Under the assumption H is a cryptographic hash function, the only approach for solving the problem of (22) appears to be picking some $\mathbf{w}$ such that $\mathsf{H}'(\mu \,\|\, \mathbf{w}) = c$, and then finding a pair $\mathbf{z}$, $\mathbf{u}'$ that satisfies $\mathbf{w} = \mathbf{A}\mathbf{z} - c\frac{q}{p}\mathbf{t} + \frac{q}{p}\mathbf{u}'$. Let $\mathbf{t}' := \mathbf{w} + c\frac{q}{p}\mathbf{t}$, then we can rewrite this as

$$\left[\mathbf{A} \mid \mathbf{I}_k\right] \begin{bmatrix} \mathbf{z} \\ \frac{q}{p}\mathbf{u}' \end{bmatrix} = \mathbf{t}'. \tag{23}$$

The concrete security that we are concerned with is the hardness of the problem of finding a pair $\mathbf{z}, \frac{q}{p}\mathbf{u}'$ that satisfies (23) and $\|\frac{q}{p}\mathbf{u}'\|_\infty, \|\mathbf{z}\|_\infty < 4\gamma_2$. This amounts to solving the $\mathsf{MSIS}_{k,(l+1),\zeta}$ problem for the matrix $\left[\mathbf{A} \mid \mathbf{t}'\right]$.

Concrete hardness of $\mathsf{MSIS}_{k,l,\zeta}$. Furthermore, the $\mathsf{MSIS}_{k,(l+1),\zeta}$ instance can be mapped to a $\mathsf{SIS}_{256k,256(l+1),\zeta}$ instance with the matrix $\mathrm{rot}(\mathbf{A} \mid \mathbf{t}') \in \mathbb{Z}_q^{256\cdot k \times 256\cdot(l+1)}$. Similarly, the $\mathsf{MSIS}_{k,l,\zeta'}$ instance can be mapped to the $\mathsf{SIS}_{256\cdot k,256\cdot l,\zeta'}$ instance. Since the values of q, k, l, and ζ' in (14) of our scheme are almost the same as those of Dilithium (only the value of q is slightly different), the MSIS instances above are also the same. Thus, in Table 1, we refer to the BKZ block size b to break SIS given in [DLL+19].

6 Results and Comparison

6.1 Data Size

Public key. The size of public key $pk = (\rho, \mathbf{t}_1)$ in MLWRSign is $32(\lceil \log p \rceil - d) \cdot k + 1)$ bytes, while that of Dilithium is $32((\lceil \log q \rceil - 14) \cdot k + 1)$ bytes. The bit-length of a coefficient of a polynomial of vector $\mathbf{t}_1$ is always 9 bits, as you can see in Fig. 10. This is because we select d such that $\lceil \log_2(60 \cdot 2^{d-1}) \rceil = \log_2(2\overline{\gamma}_2)$, thus $d := \log(2\overline{\gamma}_2) - 5$. Therefore, the bit length of $\mathbf{t}_1$ is $\log p - \log(2\overline{\gamma}_2) + 5 = \log q - \log(2\gamma_2) + 5$, which is equivalent to that of Dilithium.

Secret key. The size of secret key $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{t}_0)$ in MLWRSign is $112 + 32(l\lceil \log_2(2\eta+1) \rceil + dk)$ bytes, while that of Dilithium is $112+32((k+l)\lceil \log_2(2\eta+1) \rceil + 14k)$ bytes. While in Dilithium the noise vector $\mathbf{s}_2$ has to be included in the secret key, it is not needed to be stored since we can generate it in the Sign procedure thanks to the deterministic characteristic of LWR. Furthermore, as the modulus of $\mathbf{t}$ is reduced from q to p, the length of d is less than the value fixed

Table 2. Data sizes and CPU cycles* of MLWRSign. The parameter sets are from Table 1. For Sign, we measure the lower quartile (L), median (M), and upper quartile (U) of the cycles. For Verify and KeyGen, we write only the median of the cycles since they do not fluctuate significantly.

	I	II	III	IV	V	VI
Public key size (bytes)	896	1184	1472	1760	2336	2624
Secret key size (bytes)	1392	1872	2384	2864	3856	4336
Signature size (bytes)	1387	2044	2701	3358	4672	5329
Expected repeats (9)	4.89	8.89	4.12	5.59	3.35	3.92
Average repeats observed	3.94	8.27	3.19	4.80	2.42	3.02
Sign cycles	L: 427K M: 677K U: 1071K	L: 900K M: 1625K U: 2858K	L: 1037K M: 1412K U: 2246K	L: 1511K M: 2354K U: 3931K	L: 1932K M: 2850K U: 4538K	L: 3194K M: 4147K U: 6086K
Verify cycles	180K	309K	477K	707K	1209K	1523K
KeyGen cycles	157K	284K	436K	630K	1125K	1455K
Sign cycles (AVX2)	L: 271K M: 430K U: 697K	L: 520K M: 934K U: 1630K	L: 553K M: 757K U: 1175K	L: 812K M: 1275K U: 2108K	L: 990K M: 1465K U: 2231K	L: 1575K M: 2049K U: 2981K
Verify cycles (AVX2)	114K	170K	248K	352K	578K	719K
KeyGen cycles (AVX2)	106K	153K	212K	317K	528K	662K

* This table shows the timing results of the corrected version of MLWRSign. See Appendix A for more details on corrections performed in this paper.

in Dilithium ($d < 14$), as you can see in Fig. 10. The concrete sizes of the secret keys in Dilithium [DLL+19] are 2096, 2800, 3504, and 3856 bytes for "weak", "medium", "recommended", and "very high" parameter sets, respectively. Thus, our secret key sizes are short by 26% to 34%.

Signature. The size of the signature $sig = (\mathbf{z}, \mathbf{h}, c)$ is $32l \log_2(2\gamma_1) + \omega + k + 40$ bytes. This is the same as that of Dilithium, since the values of γ_1, β_1 (corresponds to β in Dilithium) and ω in our scheme are the same as those of Dilithium.

6.2 CPU Cycles

We implemented our scheme, and the results are shown in Table 2. They are the number of CPU cycles for KeyGen, Sign, and Verify. The numbers for Sign are lower quartile (L), median (M), and upper quartile (U) of 10,000 executions each. For Verify and KeyGen, we presented only the median of the cycles since those values did not fluctuate significantly. Signing was performed with a 32-byte message. Throughout this paper, we performed the experiments on a laptop with an Intel Core i7-9700 that runs at a base clock frequency of 3.0 GHz, and the Hyperthreading and Turbo Boost options were switched off. The code was compiled with gcc 7.5.0. Our implementation is based on the reference implementation of Dilithium that is available at [DLL+19]. Furthermore, we presented an optimized implementation of MLWRSign for CPUs

Table 3. Comparison with lattice signatures in reference implementations.

Scheme	Sec.	Cycles	Cycles (AVX2)	Bytes	Assumption	Framework
MLWRSign-III‡ **(this paper)**	123	Sign: L: 1037K, M: 1412K, U: 2246K Verify: 499K KeyGen: 456K	Sign: L: 553K, M: 757K, U: 1175K Verify: 256K KeyGen: 218K	pk: 1472 sk: 2384 sig: 2701	MLWR, MSIS	FS with abort
Dilithium-III [DLL+20]	125	Sign: L: 1363K, M: 2092K, U: 3308K Verify: 634K KeyGen: 647K	Sign: L: 313K, M: 453K, U: 688K Verify: 204K KeyGen: 262K	pk: 1472 sk: 3504 sig: 2701	RLWE, MSIS	FS with abort
Falcon-512 [FHK+20]	108	Sign: L: 890K, M: 898K, U: 924K Verify: 122K KeyGen: 23381K	Sign: L: 964K, M: 974K, U: 998K Verify: 122K KeyGen: 27128K	pk: 897 sk: 1281 sig: 666	NTRU-SIS	Hash-and-sign
qTESLA-p-III [ABB+19]	129*	Sign: L: 3753K, M: 6774K, U: 12002K Verify: 2122K KeyGen: 28445K	—†	pk: 38432 sk: 12392 sig: 5664	RLWE	FS with abort

‡ This row shows the timing results of corrected version of MLWRSign.
* Calculated from $2^{0.265b}$ with BKZ block size $b = 489$ † No AVX2-optimized version is publicly available

that supports the AVX2 instruction set. The optimized implementation speeds up the polynomial multiplication and expansion of the matrix and vectors since these computations are the most time-consuming operations.

As we stated before, we could not utilize the NTT for polynomial multiplication since we selected the modulus q in the powers of 2. To mitigate this disadvantage, we used Toom-Cook and Karatsuba polynomial multiplication instead of NTT. Additionally, we efficiently implemented the rounding operation with a simple bit shift following the method used in [BBF+19, DKRV18]. As a result, the running time of our scheme is comparable with that of Dilithium, although our secret key is short. Furthermore, the results show that our AVX2-optimized version is faster than our reference implementation in total CPU cycles by 1.53x, 1.77x, 1.93x, 1.92x, 2.05x, and 2.06x, for parameter sets (I), (II), (III), (IV), (V), and (VI), respectively.

Note that CPU cycles of Sign for the parameter set III (in the median or upper quartile) are lower than those for the parameter set II, although the parameter set III achieves higher security. This is because we use lower η in III and due to this, the expected number of rejections is less than that of the parameter set II.

6.3 Comparison with Other Lattice Signatures

Table 3 compares MLWRSign to lattice-based signature schemes that are proposed for NIST PQC, in terms of security, signature, and key sizes, and the performance of portable C reference implementations.

The most compact, in terms of key and signature sizes, lattice-based schemes are NTRU-based schemes, e.g., Falcon [FHK+19, Por19]. However, they contain several disadvantages. One disadvantage is that the security of these schemes is based on NTRU rather than (ring or module variants of) LWE. The geometric structure of NTRU lattices has recently been exploited [KF17] to produce significantly better attacks against the NTRU problem with large-modulus or small-secret, although these attacks are not applicable to the recent parameter set used in the digital signatures. The other disadvantage is that changing the security levels of those schemes is not easy since it requires a reconstruction of the schemes.

The other lattice constructions are digital signatures based on the hardness of RLWE/LWE, e.g., [ABB+19, ABB+17, Lyu12]. The disadvantage of these schemes is that both key and signature sizes and running times are high. As you can see in Table 3, data sizes and CPU cycles of the latest implementation of qTESLA [ABB+19] are much higher than other schemes.

The RLWE-based signature scheme, Dilithium, offers reasonably small signatures and public keys, and high speeds of signing and verification. In particular, the sum of the size of the public key and signature of the scheme is smaller than all the non-lattice-based schemes, to the best of our knowledge. By basing its security on MLWR, our scheme MLWRSign offers a smaller secret key than Dilithium, while the size of the public key and signature are exactly the same, and speeds of signing and verification are at the same level.

7 Conclusion

We proposed an MLWR-based digital signature scheme MLWRSign, which is a variant of Dilithium that is one of the third-round finalists of NIST PQC. To the best of our knowledge, our scheme MLWRSign is the first signature scheme whose security is based on the (variants of) LWR problem. By utilizing the simplicity of LWR in our scheme, we reduced the size of the secret key by approximately 30% compared to Dilithium, while achieving the same level of security. We efficiently implemented MLWRSign using the Toom-Cook and Karatsuba polynomial multiplication, and observed that the running time of MLWRSign is comparable to that of the reference implementation of Dilithium.

References

[ACD+18] Albrecht, M.R., Curtis, B.R., Deo, A., Davidson, A., Player, R., Postlethwaite, E.W., Virdia, F., and Wunderer, T.: Estimate All the {LWE, NTRU} Schemes! In: Catalano, D., and De Prisco, R. (eds.) SCN 2018, pp. 351–367 (2018). https://doi.org/10.1007/978-3-319-98113-0_19

[ABB+19] Alkim, E., Barreto, P.S.L.M., Bindel, N., Kramer, J., Longa, P., and Ricardini, J.E.: The Lattice-Based Digital Signature Scheme qTESLA, Cryptology ePrint Archive, Report 2019/085 (2019), https://eprint.iacr.org/2019/085

[ABB+17] Alkim, E., Bindel, N., Buchmann, J., Dagdelen, Ö., Eaton, E., Gutoski, G., Krämer, J., and Pawlega, F.: Revisiting TESLA in the Quantum Random Oracle Model. In: Lange, T., and Takagi, T. (eds.) PQCrypto 2017, pp. 143–162 (2017). https://doi.org/10.1007/978-3-319-59879-6_9

[ADPS16] Alkim, E., Ducas, L., Pöppelmann, T., and Schwabe, P.: Post-Quantum Key Exchange – A New Hope. In: USENIX Security, pp. 327–343 (2016). https://doi.org/10.5555/3241094.3241120

[BBF+19] Baan, H., Bhattacharya, S., Fluhrer, S., Garcia-Morchon, O., Laarhoven, T., Rietman, R., Saarinen, M.-J.O., Tolhuizen, L., and Zhang, Z.: Round5: Compact and Fast Post-quantum Public-Key Encryption. In: Ding, J., and Steinwandt, R. (eds.) PQCrypto 2019, pp. 83–102 (2019). https://doi.org/10.1007/978-3-030-25510-7_5

[BPR12] Banerjee, A., Peikert, C., and Rosen, A.: Pseudorandom Functions and Lattices. In: Pointcheval, D., and Johansson, T. (eds.) EUROCRYPT 2012, pp. 719–737 (2012). https://doi.org/10.1007/978-3-642-29011-4_42

[BBC+01] Beals, R., Buhrman, H., Cleve, R., Mosca, M., and de Wolf, R.: Quantum Lower Bounds by Polynomials. J. ACM 48(4), 778–797 (2001). https://doi.org/10.1145/502090.502097

[BDK+21] Beirendonck, M.V., D'anvers, J.-P., Karmakar, A., Balasch, J., and Verbauwhede, I.: A Side-Channel-Resistant Implementation of SABER. J. Emerg. Technol. Comput. Syst. 17(2) (2021). https://doi.org/10.1145/3429983

[BN06] Bellare, M., and Neven, G.: Multi-signatures in the Plain public-Key Model and a General Forking Lemma. In: Proceedings of the 13th ACM Conference on Computer and Communications Security. CCS '06, pp. 390–399. ACM (2006). https://doi.org/10.1145/1180405.1180453

[BPS16] Bellare, M., Poettering, B., and Stebila, D.: From Identification to Signatures, Tightly: A Framework and Generic Transforms. In: Cheon, J.H., and Takagi, T. (eds.) ASIACRYPT 2016, pp. 435–464 (2016). https://doi.org/10.1007/978-3-662-53890-6_15

[BR93] Bellare, M., and Rogaway, P.: Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. In: CCS '93, pp. 62–73. ACM (1993). https://doi.org/10.1145/168588.168596

[BR06] Bellare, M., and Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006, pp. 409–426 (2006). https://doi.org/10.1007/11761679_25

[BCL+19] Bernstein, D.J., Chou, T., Lange, T., von Maurich, I., Misoczki, R., Niederhagen, R., Persichetti, E., Peters, C., Schwabe, P., Sendrier, N., Szefer, J., and Wang, W.: Classic McEliece: conservative code-based cryptography, Technical Report, NIST (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

[BCLv18] Bernstein, D.J., Chuengsatiansup, C., Lange, T., and van Vredendaal, C.: NTRU Prime: Reducing Attack Surface at Low Cost. In: Adams, C., and Camenisch, J. (eds.) SAC 2017, pp. 235–260 (2018). https://doi.org/10.1007/978-3-319-72565-9_12

[BGM+16] Bogdanov, A., Guo, S., Masny, D., Richelson, S., and Rosen, A.: On the Hardness of Learning with Rounding over Small Modulus. In: Kushilevitz, E., and Malkin, T. (eds.) TCC 2016, pp. 209–224 (2016). https://doi.org/10.1007/978-3-662-49096-9_9

[BDF+11] Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., and Zhandry, M.: Random Oracles in a Quantum World. In: Lee, D.H., and Wang, X. (eds.) ASIACRYPT 2011, pp. 41–69 (2011). https://doi.org/10.1007/978-3-642-25385-0_3

[BCD+16] Bos, J., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., and Stebila, D.: Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In: CCS 2016, pp. 1006–1018. ACM (2016). https://doi.org/10.1145/2976749.2978425

[BDK+18] Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., and Stehlé, D.: CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM. In: Euro S&P 2018, pp. 353–367 (2018). https://doi.org/10.1109/EuroSP.2018.00032

[CZZ18] Chen, L., Zhang, Z., and Zhang, Z.: On the Hardness of the Computational Ring-LWR Problem and Its Applications. In: Peyrin, T., and Galbraith, S. (eds.) ASIACRYPT 2018, pp. 435–464 (2018). https://doi.org/10.1007/978-3-030-03326-2_15

[DKRV18] D'Anvers, J.-P., Karmakar, A., Roy, S.S., and Vercauteren, F.: Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. In: Joux, A., Nitaj, A., and Rachidi, T. (eds.) AFRICACRYPT 2018, pp. 282–305 (2018). https://doi.org/10.1007/978-3-319-89339-6_16

[DCP+19] Ding, J., Chen, M., Petzoldt, A., Schmidt, D., and Yang, B.: Rainbow - Algorithm Specification and Documentation, Technical Report, NIST (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

[DKL+18] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and Stehlé, D.: CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme. TCHES 2018(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268

[DLL+19] Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and Stehlé, D.: CRYSTALS–Dilithium: Algorithm Specifications and Supporting Documentationn, Technical Report, NIST (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

[DLL+20] Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and Stehlé, D.: CRYSTALS–Dilithium: Algorithm Specifications and Supporting Documentationn, Technical Report, NIST (2020), https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions

[FHK+20] Fouque, P.-A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., and Zhang, Z.: Falcon: Fast-Fourier Lattice-Based Compact Signatures over NTRU – Specifications v1.2. 2020, Technical Report, NIST (2020), https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions

[FHK+19] Fouque, P.-A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., and Zhang, Z.: Falcon: Fast-Fourier Lattice-Based Compact Signatures over NTRU, Technical

Report, NIST (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

[HRSS17] Hülsing, A., Rijneveld, J., Schanck, J., and Schwabe, P.: High-Speed Key Encapsulation from NTRU. In: Fischer, W., and Homma, N. (eds.) CHES 2017, pp. 232–252 (2017). https://doi.org/10.1007/978-3-319-66787-4_12

[KLS18] Kiltz, E., Lyubashevsky, V., and Schaffner, C.: A Concrete Treatment of Fiat-Shamir Signatures in the Quantum Random-Oracle Model. In: Nielsen, J.B., and Rijmen, V. (eds.) EUROCRYPT 2018, pp. 552–586 (2018). https://doi.org/10.1007/978-3-319-78372-7_18

[KF17] Kirchner, P., and Fouque, P.-A.: Revisiting Lattice Attacks on Overstretched NTRU Parameters. In: Coron, J.-S., and Nielsen, J.B. (eds.) EUROCRYPT 2017, pp. 3–26 (2017). https://doi.org/10.1007/978-3-319-56620-7_1

[Lyu09] Lyubashevsky, V.: Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures. In: Matsui, M. (ed.) ASIACRYPT 2009, pp. 598–616 (2009). https://doi.org/10.1007/978-3-642-10366-7_35

[Lyu12] Lyubashevsky, V.: Lattice Signatures without Trapdoors. In: Pointcheval, D., and Johansson, T. (eds.) EUROCRYPT 2012, pp. 738–755 (2012). https://doi.org/10.1007/978-3-642-29011-4_43

[Nat19] National Institute of Standards and Technology: Post-Quantum Cryptography, (2019), https://csrc.nist.gov/Projects/Post-Quantum-Cryptography (visited on 12/10/2021)

[Nat20] National Institute of Standards and Technology: Post-Quantum Cryptography – Round 2 Submissions, (2020), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions (visited on 12/10/2021)

[Nat21] National Institute of Standards and Technology: Post-Quantum Cryptography – Round 3 Submissions, (2021), https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions (visited on 12/10/2021)

[OSPG18] Oder, T., Schneider, T., Pöppelmann, T., and Güneysu, T.: Practical CCA2-Secure and Masked Ring-LWE Implementation. TCHES 2018(1), 142–174 (2018). https://doi.org/10.13154/tches.v2018.i1.142-174

[OTF+20] Okada, H., Takayasu, A., Fukushima, K., Kiyomoto, S., and Takagi, T.: A Compact Digital Signature Scheme Based on the Module-LWR Problem. In: Meng, W., Gollmann, D., Jensen, C.D., and Zhou, J. (eds.) ICICS 2020, pp. 73–90 (2020). https://doi.org/10.1007/978-3-030-61078-4_5

[OTF+21] Okada, H., Takayasu, A., Fukushima, K., Kiyomoto, S., and Takagi, T.: A Compact Digital Signature Scheme Based on the Module-LWR Problem. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E104.A(9), 1219–1234 (2021). https://doi.org/10.1587/transfun.2020DMP0012

[PS00] Pointcheval, D., and Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology 13(3), 361–396 (2000). https://doi.org/10.1007/s001450010003

[Por19] Pornin, T.: New Efficient, Constant-Time Implementations of Falcon, Cryptology ePrint Archive, Report 2019/893 (2019), https://eprint.iacr.org/2019/893

[Reg05] Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In: STOC '05, pp. 84–93. ACM (2005). https://doi.org/10.1145/1060590.1060603

[Zha12] Zhandry, M.: Secure Identity-Based Encryption in the Quantum Random Oracle Model. In: Safavi-Naini, R., and Canetti, R. (eds.) CRYPTO 2012, pp. 758–775 (2012). https://doi.org/10.1007/978-3-642-32009-5_44

A Corrections to the Previous Version [OTF+21]

In this section, we describe the details on changes from the previous version of this paper [OTF+21]. We noticed that [OTF+21, Eq. (3)] is incorrect, and thus we correct the equation and also slightly modify the Sign procedure of MLWRSign.

A.1 An Error in [OTF+21, Eq. (3)]

We used the (standard) rounding function $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ in [OTF+21] (although it was not explicitly defined in the paper). Any definition of rounding function does not simultaneously holds $\mathbb{Z}$-additivity and symmetricity. Concretely, $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ satisfies $\mathbb{Z}$-additivity, but it does not satisfy symmetricity, for half integers:

- $\mathbb{Z}$-additivity: $\forall x \in \mathbb{R}, \forall a \in \mathbb{Z}, f(x+a) = f(x) + a$
  - e.g., $\lfloor 1/2 + a \rceil = \lfloor 1/2 \rceil + a = 1 + a$
- Symmetricity: $\forall x \in \mathbb{R}, f(-x) = -f(x)$
  - e.g., $0 = \lfloor -1/2 \rceil \neq -\lfloor 1/2 \rceil = -1$

However, we incorrectly assumed the rounding function $\lfloor x \rceil := \lfloor x + \frac{1}{2} \rfloor$ simultaneously satisfy $\mathbb{Z}$-additivity and symmetricity: [OTF+21, Eq. (3)] has an error described as follows

$$\begin{aligned}
\left\lfloor \tfrac{p}{q}\mathbf{Az} \right\rceil - c\mathbf{t} &= \left\lfloor \tfrac{p}{q}\mathbf{Az} - c\mathbf{t} \right\rceil \ (\because \mathbb{Z}\text{-additivity}) \\
&= \left\lfloor \tfrac{p}{q}\mathbf{Ay} - c\mathbf{s}_2 \right\rceil \\
&= \left\lfloor \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil - (\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \right\rceil \\
&= \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil + \lfloor -(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \rceil \ (\because \mathbb{Z}\text{-additivity}) \\
&\neq \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil - \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil \ (: \text{Symmetricity does not hold.}) \\
&= \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil - \boldsymbol{\nu} \qquad (24)
\end{aligned}$$

where $\mathbf{s}_2 := \lfloor \frac{p}{q}\mathbf{As}_1 \rceil - \frac{p}{q}\mathbf{As}_1$, $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{Ay} \rceil - \frac{p}{q}\mathbf{Ay}$, $\boldsymbol{\xi}_2 := \lfloor c\mathbf{s}_2 \rceil - c\mathbf{s}_2$, and $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil$.

A.2 Actual Implementation of MLWRSign in [OTF+21]

Indeed, when we performed experiments of MLWRSign to write [OTF+21, Table 2], due to the error in (24), we observed that Verif fails when the vector $(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2)$ includes a element of (exact) 1/2. We (mistakenly) thought the failure is caused by some implementation "bug", and we heuristically tweaked the rounding function $\lfloor x \rceil' := \lceil x + \frac{1}{2} \rceil$ only for the definition of $\boldsymbol{\nu}$, i.e., we used $\boldsymbol{\nu}' := \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil'$ (instead of $\boldsymbol{\nu} := \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil$), to fix the "bug". Interestingly, this rounding function $\lfloor \cdot \rceil'$ satisfies $\lfloor -x \rceil = -\lfloor x \rceil'$, (c.f., $0 = \lfloor -1/2 \rceil = -\lfloor 1/2 \rceil' = 0$). Hence, we have:

$$\begin{aligned} \left\lfloor \tfrac{p}{q}\mathbf{Az} \right\rceil - c\mathbf{t} &= \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil + \lfloor -(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \rceil \\ &= \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil - \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil' \\ &= \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil - \boldsymbol{\nu}'. \end{aligned} \tag{25}$$

Thus, the actual implementation of MLWRSign worked correctly. We mistakenly regarded that the equations (24) and (25) are equivalent, and, as mentioned, we thought failure of Verif is a matter of implantation bugs.

A.3 Corrections Performed in This Paper

We can correct [OTF+21, Eq. (3)] in a more natural way. We simply define $\boldsymbol{\nu}'' := \lfloor -(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \rceil$, then we have,

$$\begin{aligned} \left\lfloor \tfrac{p}{q}\mathbf{Az} \right\rceil - c\mathbf{t} &= \left\lfloor \tfrac{p}{q}\mathbf{Ay} \right\rceil - \lfloor c\mathbf{s}_2 \rceil + \lfloor -(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \rceil \\ &= \mathbf{w} - \lfloor c\mathbf{s}_2 \rceil + \boldsymbol{\nu}''. \end{aligned} \tag{26}$$

Here, neither the tweaked rounding function nor the symmetricity of rounding function is needed.

Throughout this the paper, we define $\boldsymbol{\nu} := \lfloor -(\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2) \rceil$ and use the above correct equation (26). This slightly changes the Sign procedure of MLWRSign. For clarity, in Fig. 11, we highlight the corrections performed in the Sign algorithm. These minor modifications have almost no effect on the performance of Sign algorithm. For clarity, we implemented the corrected version of MLWRSign, and Table 2 (and Table 3) shows new timing results, which are almost the same as those of [OTF+21].

Apart from the above corrections, also note that a minor error in the proof of Lemma 5.4 is also corrected in this paper. In [OTF+21], we incorrectly assumed

$$\mathbf{u} := \lfloor \tfrac{p}{q}\mathbf{Az} \rceil - \lfloor \tfrac{p}{q}\mathbf{Az}' \rceil = \lfloor \tfrac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil + \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \rceil,$$

where $\boldsymbol{\xi}_1 := \lfloor \frac{p}{q}\mathbf{Az} \rceil - \frac{p}{q}\mathbf{Az}$ and $\boldsymbol{\xi}_2 := \lfloor \frac{p}{q}\mathbf{Az}' \rceil - \frac{p}{q}\mathbf{Az}'$. In this paper, we correct this equation into:

$$\mathbf{u} := \lfloor \tfrac{p}{q}\mathbf{Az} \rceil - \lfloor \tfrac{p}{q}\mathbf{Az}' \rceil = \lfloor \tfrac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil + \lfloor \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 - \boldsymbol{\xi}_3 \rceil, \tag{27}$$

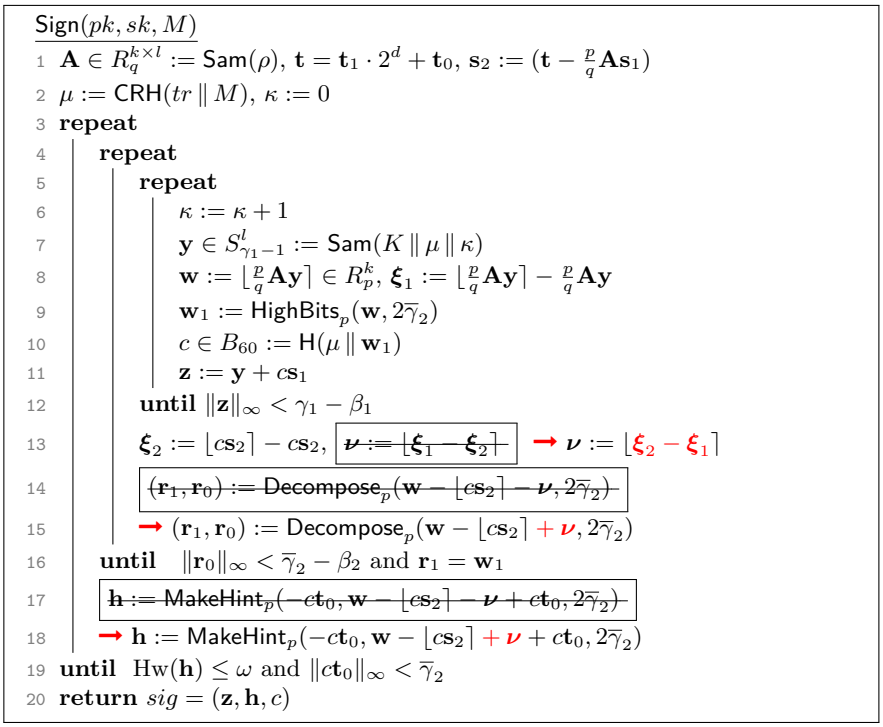

Fig. 11. Corrections performed in Sign algorithm

by adding a term of $\boldsymbol{\xi}_3 := \lfloor \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \rceil - \frac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}')$. The equation (27) holds since we have the following.

$$\begin{aligned}
\mathbf{u} &= \left\lfloor \lfloor \tfrac{p}{q}\mathbf{A}\mathbf{z} \rceil - \lfloor \tfrac{p}{q}\mathbf{A}\mathbf{z}' \rceil \right\rceil \\
&= \left\lfloor \tfrac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') + \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \right\rceil \\
&= \left\lfloor \left\lfloor \tfrac{p}{q}\mathbf{A}(\mathbf{z} - \mathbf{z}') \right\rceil + \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 - \boldsymbol{\xi}_3 \right\rceil .
\end{aligned}$$