Tweaking the Asymmetry of Asymmetric-Key Cryptography on Lattices: KEMs and Signatures of Smaller Sizes

Jiang Zhang[1(✉)], Yu Yu[2(✉)], Shuqin Fan[1(✉)], Zhenfeng Zhang[3(✉)], and Kang Yang[1(✉)]

[1] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
jiangzhang09@gmail.com, shuqinfan78@163.com, yangk@sklc.org

[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
yuyu@yuyu.hk

[3] Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing, China
zfzhang@tca.iscas.ac.cn

Abstract. Currently, lattice-based cryptosystems are less efficient than their number-theoretic counterparts (based on RSA, discrete logarithm, etc.) in terms of key and ciphertext (signature) sizes. For adequate security the former typically needs thousands of bytes while in contrast the latter only requires at most hundreds of bytes. This significant difference has become one of the main concerns in replacing currently deployed public-key cryptosystems with lattice-based ones. Observing the inherent asymmetries in existing lattice-based cryptosystems, we propose asymmetric variants of the (module-)LWE and (module-)SIS assumptions, which yield further size-optimized KEM and signature schemes than those from standard counterparts.

Following the framework of Lindner and Peikert (CT-RSA 2011) and the Crystals-Kyber proposal (EuroS&P 2018), we propose an IND-CCA secure KEM scheme from the hardness of the asymmetric module-LWE (AMLWE), whose asymmetry is fully exploited to obtain shorter public keys and ciphertexts. To target at a 128-bit quantum security, the public key (resp., ciphertext) of our KEM only has 896 bytes (resp., 992 bytes).

Our signature scheme bears most resemblance to and improves upon the Crystals-Dilithium scheme (ToCHES 2018). By making full use of the underlying asymmetric module-LWE and module-SIS assumptions and carefully selecting the parameters, we construct an SUF-CMA secure signature scheme with shorter public keys and signatures. For a 128-bit quantum security, the public key (resp., signature) of our signature scheme only has 1312 bytes (resp., 2445 bytes).

We adapt the best known attacks and their variants to our AMLWE and AMSIS problems and conduct a comprehensive and thorough analysis of several parameter choices (aiming at different security strengths) and their impacts on the sizes, security and error probability of

A. Kiayias et al. (Eds.): PKC 2020, LNCS 12111, pp. 37–65, 2020.
https://doi.org/10.1007/978-3-030-45388-6_2

lattice-based cryptosystems. Our analysis demonstrates that AMLWE and AMSIS problems admit more flexible and size-efficient choices of parameters than the respective standard versions.

1 Introduction

Despite the tremendous success of traditional public-key cryptography (also known as asymmetric-key cryptography), the typical public-key cryptosystems in widespread deployment on the Internet are based on number-theoretic hardness assumptions such as factoring and discrete logarithms, and thus are susceptible to quantum attacks [31] if large-scale quantum computers become a reality. With the advancement of quantum computing technology in recent years [19], developing post-quantum cryptography (PQC) with resistance to both classical and quantum computers has become a primary problem as well as a priority issue for the crypto community. Actually, several government agencies and standardization organizations have announced plans to solicit and standardize PQC algorithms. In 2015, the NSA [28] has announced its schedule for migration to PQC. In 2016, the NIST initiated its standardization process for post-quantum public-key encryption (PKE), key-establishment (KE) and digital signatures. Among the 69 PQC submissions received worldwide, 17 candidate PKE and KE algorithms (e.g., Kyber [6]), and 9 candidate signature schemes (e.g., Dilithium [12]) have been selected to the 2nd round of the NIST PQC standardization, where 12 out of the total 26 2nd-round candidates are lattice-based algorithms.

Most lattice-based cryptosystems base their security on the conjectured quantum hardness of the Short Integer Solution (SIS) problem [1,27] and the Learning With Errors (LWE) problem [30]. Informally speaking, the two problems are both related to solving systems of linear congruences (and are in some sense dual to each other). Let n, m, q be integers and α, β be reals, and let χ_α be some distribution (e.g., a Gaussian distribution) with parameter α defined over $\mathbb{Z}$. The SIS problem $\mathrm{SIS}^{\infty}_{n,m,q,\beta}$ in the infinity norm asks to find out a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$, given a random matrix $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n\times m}$, such that $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \le \beta$. Correspondingly, the search LWE problem $\mathrm{LWE}_{n,m,q,\alpha}$ searches for $\mathbf{s} \in \mathbb{Z}_q^n$ from samples $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$, where $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$ and $\mathbf{e} \overset{\$}{\leftarrow} \chi_\alpha^m$. Decisional LWE problem asks to distinguish $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$ from uniform distribution over $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$. For certain parameters the two (search and decisional) LWE problems are polynomially equivalent [25,30].

It has been shown that the two average-case problems SIS and LWE are at least as hard as some worst-case lattice problems (e.g., Gap-SIVP) for certain parameter choices [27,30]. Moreover, quantum algorithms are not known to have substantial advantages (beyond polynomial speedup) over classical ones in solving these problems, which makes SIS and LWE ideal candidates for post-quantum cryptography. We mention a useful variant of LWE, called the (Hermite) normal form of LWE, where the secret $\mathbf{s}$ is sampled from noise distribution χ_α^n (instead of uniform). The standard LWE and its normal form were known to be equivalent up to a polynomial number of samples [4]. Furthermore, the use of a "small" secret in LWE comes in handy in certain application scenarios, e.g., for better managing the growth of the noise in fully homomorphic encryption [7,10].

SIS is usually used in constructing signature schemes, and LWE is better suited for PKE schemes. However, the standard LWE and SIS problems seem to suffer some constraints in choosing parameters for some practical cryptographic schemes. For example, the LWE parameter for achieving a 128-bit (quantum) security typically cannot provide a matching decryption failure probability ν (say $\nu = 2^{-128}$) for the resulting LWE-based PKE scheme. Note that a larger ν (i.e., $\nu > 2^{-128}$) may sacrifice the security, and a smaller ν (i.e., $\nu < 2^{-128}$) may compromise the performance. To this end, we introduce special variants of SIS and LWE, referred to as asymmetric SIS (ASIS) and asymmetric LWE (ALWE).

Informally, the ASIS problem $\mathrm{ASIS}^{\infty}_{n,m_1,m_2,q,\beta_1,\beta_2}$ refers to the problem that, given a random $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times(m_1+m_2)}$, find out a non-zero $\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{Z}^{m_1+m_2}$ satisfying $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$, $\|\mathbf{x}_1\|_\infty \leq \beta_1$ and $\|\mathbf{x}_2\|_\infty \leq \beta_2$. It is easy to see that $\mathrm{ASIS}^{\infty}_{n,m_1,m_2,q,\beta_1,\beta_2}$ is at least as hard as $\mathrm{SIS}^{\infty}_{n,m_1+m_2,q,\max(\beta_1,\beta_2)}$. Thus, we have

$$\mathrm{SIS}^{\infty}_{n,m_1+m_2,q,\max(\beta_1,\beta_2)} \preceq \mathrm{ASIS}^{\infty}_{n,m_1,m_2,q,\beta_1,\beta_2} \preceq \mathrm{SIS}^{\infty}_{n,m_1+m_2,q,\min(\beta_1,\beta_2)}.$$

This lays the theoretical foundation for constructing secure signatures based on the ASIS problem. In addition, we investigate a class of algorithms for solving the ASIS problem, and provide a method for selecting appropriate parameters for different security levels with reasonable security margin.

Correspondingly, the ALWE problem $\mathrm{ALWE}_{n,m,q,\alpha_1,\alpha_2}$ asks to find out $\mathbf{s} \in \mathbb{Z}_q^n$ from samples $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}, \mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n, \mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^m$. The hardness of ALWE may depend on the actual distribution from which $\mathbf{s}$ (or $\mathbf{e}$) is sampled, and thus we cannot simply compare the hardness of LWE and ALWE like we did for SIS and ASIS. However, the relation below remains valid for our parameter choices in respect to all known solving algorithms despite the lack of a proof in general:[1]

$$\mathrm{LWE}_{n,m,q,\min(\alpha_1,\alpha_2)} \preceq \mathrm{ALWE}_{n,m,q,\alpha_1,\alpha_2} \preceq \mathrm{LWE}_{n,m,q,\max(\alpha_1,\alpha_2)}.$$

More importantly, the literature [9,16,26] suggests that ALWE can reach comparable hardness to standard LWE as long as the secret is sampled from a distribution (i.e., $\chi_{\alpha_1}^n$) with sufficiently large entropy (e.g., uniform distribution over $\{0,1\}^n$) and appropriate values are chosen for other parameters. This shows the possibility of constructing secure cryptographic schemes based on the ALWE problem. We also note that Cheon et al. [11] introduced a variant of LWE that is quite related to ALWE, where $\mathbf{s}$ and $\mathbf{e}$ are sampled from different distributions (notice that $\mathbf{s}$ and $\mathbf{e}$ in the ALWE problem are sampled from the same distribution χ, albeit with different parameters α_1 and α_2). By comprehensively comparing, analyzing and optimizing the state-of-the-art LWE solving algorithms, we establish approximate relations between parameters of ALWE and LWE, and suggest practical parameter choices for several levels of security strength intended for ALWE.

[1] In the full version, we show that the relations actually hold for discrete Gaussian distributions and binomial distributions under certain choices of parameters.

The definitions of the aforementioned variants can be naturally generalized to the corresponding ring and module versions, i.e., ring-LWE/SIS and module-LWE/SIS. As exhibited in [6,12], module-LWE/SIS allows for better trade-off between security and performance. We will use the asymmetric module-LWE problem (AMLWE) and the asymmetric module-SIS problem (AMSIS) to build a key encapsulation mechanism (KEM) and a signature scheme of smaller sizes.

Technically, our KEM scheme is mainly based on the PKE schemes in [6,22], except that we make several modifications to utilize the inherent asymmetry of the (M)LWE secret and noise in contributing to the decryption failure probabilities, which allow us to obtain smaller public keys and ciphertexts. In Sect. 3.1, we will further discuss this asymmetry in the design of existing schemes, and illustrate our design rationale in more details. For a targeted 128-bit security, the public key (resp., ciphertext) of our KEM only has 896 bytes (resp., 992 bytes).

Our signature scheme bears most resemblance to Dilithium in [12]. The main difference is that we make several modifications to utilize the asymmetric parameterization of the (M)LWE and (M)SIS to reach better trade-offs among computational costs, storage overhead and security, which yields smaller public keys and signatures without sacrificing the security or computational efficiency. In Sect. 4.1, we will further discuss the asymmetries in existing constructions, and illustrate our design rationale in more details. For a targeted 128-bit quantum security, the public key (resp., signature) of our signature scheme only has 1312 bytes (resp., 2445 bytes).

We make a comprehensive and in-depth study on the concrete hardness of AMLWE and AMSIS by adapting the best known attacks (that were originally intended for MLWE and MSIS respectively) and their variants (that were modified to solve AMLWE and AMSIS respectively), and provide several choices of parameters for our KEM and signature schemes aiming at different security strengths. The implementation of our schemes (and its comparison with the counterparts) confirms that our schemes are practical and competitive. We compare our KEM with NIST round2 lattice-based PKEs/KEMs in Sect. 1.1, and compare our signature with NIST round2 lattice-based signatures in Sect. 1.2.

1.1 Comparison with NIST Round2 Lattice-Based PKEs/KEMs

As our KEM is built upon Kyber [6], we would like to first give a slightly detailed comparison between our KEM and Kyber-round2 [6] in Table 1. Our software is implemented in C language with optimized number theory transform (NTT) and vector multiplication using AVX2 instructions. The running times of KeyGen, Encap and Decap algorithms are measured in averaged CPU cycles of 10000 times running on a 64-bit Ubuntu 14.4 LTS ThinkCenter desktop (equipped with Intel Core-i7 4790 3.6 GHz CPU and 4 GB memory). The sizes of public key $|pk|$, secret key $|sk|$, ciphertext $|C|$ are measured in terms of bytes. The column $|ss|$ gives the size of the session key that is encapsulated by each ciphertext. The column "Dec. Failure" lists the probabilities of decryption failure. The last column "Quant. Sec." gives the estimated quantum security level expressed in bits.

Note that for $X \in \{512, 768, 1024\}$ aiming at NIST Category I, III and V, the estimated quantum security of our KEM Π_{KEM}-X is slightly lower than that

Table 1. Comparison between Our KEM Π_{KEM} and Kyber-round2

Schemes	$\lvert pk\rvert$ (Bytes)	$\lvert sk\rvert$ (Bytes)	$\lvert C\rvert$ (Bytes)	$\lvert ss\rvert$ (Bytes)	KeyGen (AVX2)	Encap (AVX2)	Decap	Dec. failure	Quant. Sec.
Kyber-512	800	1632	736	32	37 792	54 465	41 614	2^{-178}	100
Π_{KEM}-512	**672**	**1568**	**672**	32	66 089	70 546	56 385	2^{-82}	102
Π_{KEM}-512†	800	1632	**640**	32	-	-	-	2^{-82}	99
Kyber-768	1184	2400	1088	32	66 760	86 608	69 449	2^{-164}	164
Π_{KEM}-768	**896**	**2208**	**992**	32	84 504	93 069	76 568	2^{-128}	147
Π_{KEM}-768†	1184	2400	**960**	32	-	-	-	2^{-130}	157
Kyber-1024	1568	3168	1568	32	88 503	116 610	96 100	2^{-174}	230
Π_{KEM}-1024	**1472**	3392	**1536**	**64**	115 268	106 740	92 447	2^{-211}	213
Π_{KEM}-1024†	1728	3648	**1472**	**64**	-	-	-	2^{-198}	206

of Kyber-X, but we emphasize that our parameter choices have left out sufficient security margin reserved for further development of attacks. For example, our Π_{KEM}-768 reaches an estimated quantum security of 147 bits and a 2^{-128} decryption failure probability, which we believe is sufficient to claim the same targeted 128-bit quantum security (i.e., NIST Category III) as Kyber-768. We also note that the parameter choice of Π_{KEM}-1024 is set to encapsulate a 64-byte session key, which is twice the size of that achieved by Kyber-1024. This decision is based on the fact that a 32-byte session key may not be able to provide a matching security strength, say, more than 210-bit quantum security (even if the Grover algorithm [17] cannot provide a real quadratic speedup over classical algorithms in practice).

We note that the Kyber team [6] removed the public-key compression to purely base their Kyber-round2 scheme on the standard MLWE problem and obtained (slightly) better computational efficiency (for saving several operations such as NTT). On the first hand, as commented by the Kyber team that "we strongly believe that this didn't lower actual security", we prefer to use the public-key compression to obtain smaller public key sizes (with the cost of a slightly worse computational performance). On the other hand, one can remove the public-key compression and still obtain a scheme with shorter ciphertext size (see Π_{KEM}-X† in Table 1), e.g., a reduction of 128 bytes in the ciphertext size over Kyber-768 at the targeted 128-bit quantum security by using a new parameter set $(n, k, q, \eta_1, \eta_2, d_u, d_v) = (256, 3, 3329, 1, 2, 9, 3)$ (see Π_{KEM}-X† in Table 5).

We also give a comparison between our KEM and NIST round2 lattice-based PKEs/KEMs in Table 2. For simplicity, we only compare those schemes under the parameter choices targeted at IND-CCA security and 128-bit quantum security in terms of space and time (measured in averaged CPU cycles of running 10000 times) on the same computer. We failed to run the softwares of the schemes marked with '*' on our experiment computer (but a public evaluation on the Round1 submissions suggests that Three-Bears may have better computational efficiency than Kyber and ours). As shown in Table 2, our Π_{KEM} has a very competitive performance in terms of both sizes and computational efficiency.

Table 2. Comparison between Π_{KEM} and NIST Round2 lattice-based PKEs/KEMs

Schemes	$\|pk\|$ (Bytes)	$\|sk\|$ (Bytes)	$\|C\|$ (Bytes)	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)	Problems
Frodo*	15 632	31 296	15 744	-	-	-	LWE
Kyber	1184	2400	1088	66 760	86 608	69 449	MLWE
LAC	1056	2080	1188	108 724	166 458	208 814	RLWE
Newhope	1824	3680	2208	146 909	233 308	237 619	RLWE
NTRU-Prime*	1158	1763	1039	-	-	-	NTRU variant
NTRU	1138	1450	1138	378 728	109 929	75 905	NTRU
Round5*	983	1031	1119	-	-	-	GLWR
Saber	992	2304	1088	117 504	139 044	133 875	MLWER
Three-Bears*	1194	40	1307	-	-	-	MLWE variant
Π_{KEM}-768	**896**	2208	**992**	84 504	93 069	76 568	AMLWE

1.2 Comparison with NIST Round2 Lattice-Based Signatures

We first give a slightly detailed comparison between our signature Π_{SIG} with Dilithium-round2 [12] in Table 3. Similarly, the running times of the KeyGen, Sign and Verify algorithms are measured in the average number of CPU cycles (over 10000 times) on the same machine configuration as before. The sizes of public key $|pk|$, secret key $|sk|$, signature $|\sigma|$ are counted in bytes. As shown in Table 3, the estimated quantum security of Π_{SIG}-1024 is slightly lower than that of Dilithium-1024, but those at Π_{SIG}-1280 and Π_{SIG}-1536 are slightly higher. In all, our scheme has smaller public key and signatures while still providing comparable efficiency to (or even slightly faster than) Dilithium-round2.

Table 3. Comparison between Our Signature Π_{SIG} and Dilithium-round2

Schemes	$\|pk\|$ (Bytes)	$\|sk\|$ (Bytes)	$\|\sigma\|$ (Bytes)	KeyGen (AVX2)	Sign (AVX2)	Verify (AVX2)	Quantum Sec.
Dilithium-1024	1184	2800	2044	140 181	476 598	129 256	91
Π_{SIG}-1024	**1056**	**2448**	**1852**	126 719	407 981	113 885	90
Dilithium-1280	1472	3504	2701	198 333	657 838	187 222	125
Π_{SIG}-1280	**1312**	**3376**	**2445**	198 876	634 128	170 283	128
Dilithium-1536	1760	3856	3366	269 430	639 966	260 503	158
Π_{SIG}-1536	**1568**	3888	**3046**	296 000	800 831	259 855	163

We also compare our signature with NIST round2 lattice-based signatures: Falcon, qTESLA and Dilithium, where the first one is an instantiation of full-domain hash and trapdoor sampling [15] on NTRU lattices (briefly denoted as FDH-like methodology), and the last two follows the more efficient Fiat-Shamir heuristic with rejection sampling (briefly denoted as FS-like methodology) [24].

As we failed to run the softwares of Falcon and qTESLA on our experiment computer (but a public evaluation on the round1 submissions suggests that Falcon and qTESLA are probably much slower than Dilithium), we only compare the sizes of those schemes at all parameter choices in Table 4. Note that the qTESLA team had dropped all the parameter sets of qTESLA-round2, the figures in Table 4 corresponds to their new choices of parameter sets.

Table 4. Comparison between Π_{SIG} and NIST Round2 lattice-based signatures

NIST category	Schemes	$\lvert pk\rvert$ (Bytes)	$\lvert sk\rvert$ (Bytes)	$\lvert\sigma\rvert$ (Bytes)	Problems	Methodology
I	Falcon-512	897	4097	690	NTRU	FDH-like
	qTESLA-1024	14 880	5 184	2 592	RLWE	FS-like
	Dilithium-1024	1184	2800	2044	MLWE, MSIS	
	Π_{SIG}-1024	1056	2448	1852	AMLWE, AMSIS	
II	Dilithium-1280	1472	3504	2701	MLWE, MSIS	FS-like
	Π_{SIG}-1280	1312	3376	2445	AMLWE, AMSIS	
III	Falcon-1024	1793	8193	1330	NTRU	FDH-like
	qTESLA-2048	38 432	12 352	5 664	RLWE	FS-like
	Dilithium-1536	1760	3856	3366	MLWE, MSIS	
	Π_{SIG}-1536	1568	3888	3046	AMLWE, AMSIS	

1.3 Organizations

Section 2 gives the preliminaries and background information. Section 3 describes the KEM scheme from AMLWE. Section 4 presents the digital signature scheme from AMLWE and AMSIS. Section 5 analyzes the concrete hardness of AMLWE and AMSIS by adapting the best known attacks.

2 Preliminaries

2.1 Notation

We use κ to denote the security parameter. For a real number $x \in \mathbb{R}$, $\lceil x \rfloor$ denotes the closest integer to x (with ties being rounded down, i.e., $\lceil 0.5 \rfloor = 0$). We denote by R the ring $R = \mathbb{Z}[X]/(X^n + 1)$ and by R_q the ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$, where n is a power of 2 so that $X^n + 1$ is a cyclotomic polynomial. For any positive integer η, S_η denotes the set of ring elements of R that each coefficient is taken from $\{-\eta, -\eta + 1 \ldots, \eta\}$. The regular font letters (e.g., a, b) represent elements in R or R_q (including elements in $\mathbb{Z}$ or $\mathbb{Z}_q$), and bold lower-case letters (e.g., $\mathbf{a}$, $\mathbf{b}$) denote vectors with coefficients in R or R_q. By default, all vectors will be column vectors. Bold upper-case letters (e.g., $\mathbf{A}$, $\mathbf{B}$) represent matrices. We denote by $\mathbf{a}^T$ and $\mathbf{A}^T$ the transposes of vector $\mathbf{a}$ and matrix $\mathbf{A}$ respectively.

We denote by $x \overset{\$}{\leftarrow} D$ sampling x according to a distribution D and by $x \overset{\$}{\leftarrow} S$ denote sampling x from a set S uniformly at random. For two bit-strings s and t, $s\|t$ denotes the concatenation of s and t. We use $\log_b$ to denote the logarithm function in base b (e.g., 2 or natural constant e) and log to represent $\log_e$. We say that a function $f : \mathbb{N} \rightarrow [0, 1]$ is *negligible*, if for every positive c and all sufficiently large κ it holds that $f(\kappa) < 1/\kappa^c$. We denote by $\mathsf{negl} : \mathbb{N} \rightarrow [0, 1]$ an (unspecified) negligible function. We say that f is *overwhelming* if $1 - f$ is negligible.

2.2 Definitions

Modular Reductions. For an even positive integer α, we define $r' = r \bmod^{\pm} \alpha$ as the unique element in the range $(-\frac{\alpha}{2}, \frac{\alpha}{2}]$ such that $r' = r \bmod \alpha$. For an odd positive integer α, we define $r' = r \bmod^{\pm} \alpha$ as the unique element in the range $[-\frac{\alpha-1}{2}, \frac{\alpha-1}{2}]$ such that $r' = r \bmod \alpha$. For any positive integer α, we define $r' = r \bmod^{+} \alpha$ as the unique element in the range $[0, \alpha)$ such that $r' = r \bmod \alpha$. When the exact representation is not important, we simply write $r \bmod \alpha$.

Sizes of Elements. For an element $w \in \mathbb{Z}_q$, we write $\|w\|_\infty$ to mean $|w \bmod^{\pm} q|$. The ℓ_∞ and ℓ_2 norms of a ring element $w = w_0 + w_1X + \cdots + w_{n-1}X^{n-1} \in R$ are defined as follows:

$$\|w\|_\infty = \max_i \|w_i\|_\infty, \ \|w\| = \sqrt{\|w_0\|_\infty^2 + \ldots + \|w_{n-1}\|_\infty^2}.$$

Similarly, for $\mathbf{w} = (w_1, \ldots, w_k) \in R^k$, we define

$$\|\mathbf{w}\|_\infty = \max_i \|w_i\|_\infty, \ \|\mathbf{w}\| = \sqrt{\|w_1\|^2 + \ldots + \|w_k\|^2}.$$

Modulus Switching. For any positive integers p, q, we define the modulus switching function $\lceil \cdot \rfloor_{q \rightarrow p}$ as:

$$\lceil x \rfloor_{q \rightarrow p} = \lceil (p/q) \cdot x \rfloor \bmod^{+} p.$$

It is easy to show that for any $x \in \mathbb{Z}_q$ and $p < q \in \mathbb{N}$, $x' = \lceil \lceil x \rfloor_{q \rightarrow p} \rfloor_{p \rightarrow q}$ is an element close to x, i.e,

$$|x' - x \bmod^{\pm} q| \leq \left\lceil \frac{q}{2p} \right\rfloor.$$

When $\lceil \cdot \rfloor_{q \rightarrow p}$ is used to a ring element $x \in R_q$ or a vector $\mathbf{x} \in R_q^k$, the procedure is applied to each coefficient individually.

Binomial Distribution. The centered binomial distribution B_η with some positive integer η is defined as follows:

$$B_\eta = \left\{ \sum_{i=1}^{\eta} (a_i - b_i) : (a_1, \dots, a_\eta, b_1, \dots, b_\eta) \xleftarrow{\$} \{0,1\}^{2\eta} \right\}$$

When we write that sampling a polynomial $g \xleftarrow{\$} B_\eta$ or a vector of such polynomials $\mathbf{g} \xleftarrow{\$} B_\eta$, we mean that sampling each coefficient from B_η individually.

2.3 High/Low Order Bits and Hints

Our signature scheme will adopt several simple algorithms proposed in [12] to extract the "higher-order" bits and "lower-order" bits from elements in $\mathbb{Z}_q$. The goal is that given an arbitrary element $r \in \mathbb{Z}_q$ and another small element $z \in \mathbb{Z}_q$, we would like to recover the higher order bits of $r + z$ without needing to store z. Ducas et al. [12] define algorithms that take r, z and generate a 1-bit hint h that allows one to compute the higher order bits of $r + z$ just using r and h. They consider two different ways which break up elements in $\mathbb{Z}_q$ into their "higher-order" bits and "lower-order" bits. The related algorithms are described in Algorithms 1–6. We refer the reader to [12] for the illustration of the algorithms.

The following lemmas claim some crucial properties of the above supporting algorithms, which are necessary for the correctness and security of our signature scheme. We refer to [12] for their proofs.

Lemma 1. *Let q and α be positive integers such that $q > 2\alpha$, $q \bmod \alpha = 1$ and α is even. Suppose that $\mathbf{r}, \mathbf{z}$ are vectors of elements in R_q, where $\|\mathbf{z}\|_\infty \le \alpha/2$. Let $\mathbf{h}, \mathbf{h}'$ be vectors of bits. Then, algorithms $\mathsf{HighBits}_q$, $\mathsf{MakeHint}_q$ and $\mathsf{UseHint}_q$ satisfy the following properties:*

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

Lemma 2. *If $\|\mathbf{s}\|_\infty \le \beta$ and $\|\mathsf{LowBits}_q(\mathbf{r}, \alpha)\|_\infty < \alpha/2 - \beta$, then we have:*

$$\mathsf{HighBits}_q(\mathbf{r}, \alpha) = \mathsf{HighBits}_q(\mathbf{r} + \mathbf{s}, \alpha).$$

3 An Improved KEM from AMLWE

Our scheme is based on the key encapsulation mechanism in [6,22]. The main difference is that our scheme uses a (slightly) different hardness problem, which gives us a flexible way to set the parameters for both performance and security.

Algorithm 1: $\mathsf{Power2Round}_q(r, d)$

```
1 $r := r \bmod^+ q$;
2 $r_0 := r \bmod^\pm 2^d$;
3 $r_1 := (r - r_0)/2^d$;
4 return $(r_1, r_0)$;
```

Algorithm 2: $\mathsf{Decompose}_q(r, \alpha)$

```
1 $r := r \bmod^+ q$;
2 $r_0 := r \bmod^\pm \alpha$;
3 if $r - r_0 = q - 1$ then
4     $r_1 := 0$;
5     $r_0 := r_0 - 1$;
6 else
7     $r_1 := (r - r_0)/\alpha$;
8 end
9 return $(r_1, r_0)$;
```

Algorithm 3: $\mathsf{HighBits}_q(r, \alpha)$

```
1 $(r_1, r_0) := \mathsf{Decompose}_q(r, \alpha)$;
2 return $r_1$;
```

Algorithm 4: $\mathsf{LowBits}_q(r, \alpha)$

```
1 $(r_1, r_0) := \mathsf{Decompose}_q(r, \alpha)$;
2 return $r_0$;
```

Algorithm 5: $\mathsf{MakeHint}_q(z, r, \alpha)$

```
1 $r_1 := \mathsf{HighBits}_q(r, \alpha)$;
2 $v_1 := \mathsf{HighBits}_q(r + z, \alpha)$;
3 if $r_1 \neq v_1$ then
4     $h := 1$;
5 else
6     $h := 0$;
7 end
8 return $h$;
```

3.1 Design Rationale

For simplicity and clarity, we explain the core idea using the (A)LWE-based public-key encryption (PKE) scheme as an example. Note that most LWE-based

Algorithm 6: $\mathsf{UseHint}_q(h, r, \alpha)$

```
1 $k := (q-1)/\alpha$;
2 $(r_1, r_0) := \mathsf{Decompose}_q(r, \alpha)$;
3 if $h = 1$ and $r_0 > 0$ then
4     $r_1 := (r_1 + 1) \bmod^+ k$;
5 end
6 if $h = 1$ and $r_0 \leq 0$ then
7     $r_1 := (r_1 - 1) \bmod^+ k$;
8 end
9 return $r_1$;
```

PKE schemes mainly follow the framework in [22] up to the choices of parameters and noise distributions. Let $n, q \in \mathbb{Z}$ be positive integers, and let $\chi_\alpha \subset \mathbb{Z}$ be a discrete Gaussian distribution with standard variance $\alpha \in \mathbb{R}$. The LWE-based PKE works as follows:

- **Key generation**: randomly choose $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times n}$, $\mathbf{s}, \mathbf{e} \overset{\$}{\leftarrow} \chi_\alpha^n$ and compute $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$. Return the public key $pk = (\mathbf{A}, \mathbf{b})$ and secret key $sk = \mathbf{s}$.
- **Encryption**: given the public key $pk = (\mathbf{A}, \mathbf{b})$ and a plaintext $\mu \in \{0, 1\}$, randomly choose $\mathbf{r}, \mathbf{x}_1 \overset{\$}{\leftarrow} \chi_\alpha^n, x_2 \overset{\$}{\leftarrow} \chi_\alpha$ and compute $\mathbf{c}_1 = \mathbf{A}^T\mathbf{r} + \mathbf{x}_1$, $c_2 = \mathbf{b}^T\mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rfloor$. Finally, return the ciphertext $C = (\mathbf{c}_1, c_2)$.
- **Decryption**: given the secret key $sk = \mathbf{s}$ and a ciphertext $C = (\mathbf{c}_1, c_2)$, compute $z = c_2 - \mathbf{s}^T\mathbf{c}_1$ and output $\lceil z \cdot \frac{2}{q} \rfloor \bmod 2$ as the decryption result.

For a honestly generated ciphertext $C = (\mathbf{c}_1, c_2)$ that encrypts plaintext $\mu \in \{0, 1\}$, we have:

$$z = c_2 - \mathbf{s}^T\mathbf{c}_1 = \mu \cdot \left\lceil \frac{q}{2} \right\rfloor + \underbrace{\mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{x}_1 + x_2}_{\text{noise } e'}. \tag{1}$$

Thus, the decryption algorithm is correct as long as $|e'| < \frac{q}{4}$. Since $|x_2| \ll |\mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{x}_1|$, the magnitude of $|e'|$ mainly depends on $|\mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{x}_1|$. That is, the LWE secret $(\mathbf{s}, \mathbf{r})$ and the noise $(\mathbf{e}, \mathbf{x}_1)$ contribute almost equally to the magnitude of $|e'|$. Moreover, for a fixed n the expected magnitude of $|\mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{x}_1|$ is a monotonically increasing function of α:

$$\text{larger } \alpha \Rightarrow \text{ larger } |\mathbf{e}^T\mathbf{r} - \mathbf{s}^T\mathbf{x}_1| \Rightarrow \text{ larger } |e'|.$$

Let ν be the probability that the decryption algorithm fails, and let λ be the complexity of solving the underlying LWE problem. Ideally, for a targeted security strength κ, we hope that $\nu = 2^{-\kappa}$ and $\lambda = 2^\kappa$, since a large ν (i.e., $\nu > 2^{-\kappa}$) will sacrifice the overall security, and a large λ (i.e., $\lambda > 2^\kappa$) may compromise the overall performance. Since both ν and λ are strongly related to the ratio α/q of the Gaussian parameter α and the modulus q, it is hard to come up with an appropriate choice of (α, q) to simultaneously achieve the best of the two worlds.

To obtain smaller public keys and ciphertexts (and thus improve the communication efficiency), many schemes use the modulus switching technique [8,10] to compress public keys and ciphertexts. We refer to the following scheme that adopts modulus switching technique to compress public keys and ciphertexts, where $p_1, p_2, p_3 \in \mathbb{Z}$ are parameters for compression (p_1 for the public key and p_2, p_3 for ciphertexts).

- **Key generation**: pick $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$ and $\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi_\alpha^n$ and compute $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$. Then, return the public key $pk = (\mathbf{A}, \bar{\mathbf{b}} = \lceil \mathbf{b} \rfloor_{q\to p_1})$ and the secret key $sk = \mathbf{s}$.
- **Encryption**: given the public key $pk = (\mathbf{A}, \bar{\mathbf{b}})$ and a plaintext $\mu \in \{0,1\}$, randomly choose $\mathbf{r}, \mathbf{x}_1 \xleftarrow{\$} \chi_\alpha^n, x_2 \xleftarrow{\$} \chi_\alpha$, and compute $\mathbf{c}_1 = \mathbf{A}^T\mathbf{r} + \mathbf{x}_1$ and $c_2 = \lceil \bar{\mathbf{b}} \rfloor_{p_1\to q}^T \mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rfloor$. Return the ciphertext $C = (\bar{\mathbf{c}}_1 = \lceil \mathbf{c}_1 \rfloor_{q\to p_2}, \bar{\mathbf{c}}_2 = \lceil c_2 \rfloor_{q\to p_3})$.
- **Decryption**: given the secret key $sk = \mathbf{s}$ and a ciphertext $C = (\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$, compute $z = \lceil \bar{\mathbf{c}}_2 \rfloor_{p_3\to q} - \mathbf{s}^T \lceil \bar{\mathbf{c}}_1 \rfloor_{p_2\to q}$ and output $\lceil z \rfloor_{q\to 2} = \lceil z \cdot \frac{2}{q} \rfloor \bmod 2$ as the decryption result.

Let

$$\bar{\mathbf{e}} = \lceil \lceil \mathbf{b} \rfloor_{q\to p_1} \rfloor_{p_1\to q} - \mathbf{b}, \bar{\mathbf{x}}_1 = \lceil \lceil \mathbf{c}_1 \rfloor_{q\to p_2} \rfloor_{p_2\to q} - \mathbf{c}_1, \bar{x}_2 = \lceil \lceil c_2 \rfloor_{q\to p_3} \rfloor_{p_3\to q} - c_2.$$

It is easy to verify $\|\bar{\mathbf{e}}\|_\infty \le \frac{q}{2p_1}, \|\bar{\mathbf{x}}_1\|_\infty \le \frac{q}{2p_2}$, and $|\bar{x}_2| \le \frac{q}{2p_3}$. For any valid ciphertext $C = (\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$ that encrypts $\mu \in \{0,1\}$ we have

$$\begin{aligned} z &= \lceil \bar{\mathbf{c}}_2 \rfloor_{p_3\to q} - \mathbf{s}^T \lceil \bar{\mathbf{c}}_1 \rfloor_{p_2\to q} \\ &= \mu \cdot \lceil \tfrac{q}{2} \rfloor + \underbrace{(\mathbf{e} + \bar{\mathbf{e}})^T \mathbf{r} - \mathbf{s}^T(\mathbf{x}_1 + \bar{\mathbf{x}}_1) + (x_2 + \bar{x}_2)}_{\text{noise } e'} \end{aligned} \tag{2}$$

Apparently, the smaller values for p_1, p_2, p_3 the better compression rate is achieved for public keys and ciphertexts. At the same time, however, by the definitions of $\bar{\mathbf{e}}, \bar{\mathbf{x}}_1, \bar{x}_2$ we know that smaller p_1, p_2, p_3 also result in a larger noise e'. Notice that when p_1, p_2, p_3 are much smaller than q, we will have $\|\bar{\mathbf{e}}\|_\infty \gg \|\mathbf{e}\|_\infty$, $\|\bar{\mathbf{x}}_1\|_\infty \gg \|\mathbf{x}_1\|_\infty$ and $|\bar{x}_2| \gg |x_2|$, which further leads to asymmetric roles of $(\mathbf{e}, \mathbf{x}_1, x_2)$ and $(\mathbf{s}, \mathbf{r})$ in contributing to the resulting size of $|e'|$, i.e., for specific (p_1, p_2, p_3) decreasing (resp., increasing) $\|\mathbf{s}\|_\infty$ or $\|\mathbf{r}\|_\infty$ would significantly reducing (resp., enlarging) the noise $|e'|$, and in contrast, changing the size of $\|\mathbf{e}\|_\infty, \|\mathbf{x}_1\|_\infty$ and $|x_2|$ would not result in substantial change to $|e'|$.

The asymmetry observed above motivates the design of our ALWE-based PKE, which uses different noise distributions χ_{α_1} and χ_{α_2} (i.e., same distribution with different parameters α_1 and α_2) for the secrets (i.e., $\mathbf{s}$ and $\mathbf{r}$) and the errors (i.e., $\mathbf{e}, \mathbf{x}_1, x_2$), respectively.

- **Key generation**: pick $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$, $\mathbf{s} \xleftarrow{\$} \chi_{\alpha_1}^n$ and $\mathbf{e} \xleftarrow{\$} \chi_{\alpha_2}^n$, compute $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$. Then, return the public key $pk = (\mathbf{A}, \bar{\mathbf{b}} = \lceil \mathbf{b} \rfloor_{q\to p_1})$ and the secret key $sk = \mathbf{s}$.

- **Encryption**: given the public key $pk = (\mathbf{A}, \bar{\mathbf{b}})$ and a plaintext $\mu \in \{0,1\}$, randomly choose $\mathbf{r} \xleftarrow{\$} \chi_{\alpha_1}^n, \mathbf{x}_1 \xleftarrow{\$} \chi_{\alpha_2}^n, x_2 \xleftarrow{\$} \chi_{\alpha_2}$, compute $\mathbf{c}_1 = \mathbf{A}^T\mathbf{r} + \mathbf{x}_1$ and $c_2 = \lceil \mathbf{b} \rfloor_{p_1 \to q}^T \mathbf{r} + x_2 + \mu \cdot \lceil \frac{q}{2} \rfloor$, and return the ciphertext $C = (\bar{\mathbf{c}}_1 = \lceil \mathbf{c}_1 \rfloor_{q \to p_2}$ and $\bar{\mathbf{c}}_2 = \lceil c_2 \rfloor_{q \to p_3})$.
- **Decryption**: Given the secret key $sk = \mathbf{s}$ and the ciphertext $C = (\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$, compute $z = \lceil \bar{\mathbf{c}}_2 \rfloor_{p_3 \to q} - \mathbf{s}^T \lceil \bar{\mathbf{c}}_1 \rfloor_{p_2 \to q}$ and output $\lceil z \rfloor_{q \to 2} = \lceil z \cdot \frac{2}{q} \rfloor \bmod 2$ as the decryption result.

Similarly, for ciphertext $C = (\bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2)$ we have the same z and e' as defined in (2), where the difference is that now $\|\mathbf{s}\|_\infty$ and $\|\mathbf{r}\|_\infty$ are determined by α_1, and that $\|\mathbf{e}\|_\infty, \|\mathbf{x}_1\|_\infty$ and $|x_2|$ are determined by α_2. Intuitively, we wish to use small α_1 in order to keep $|e'|$ small, and at the same time choose relatively large α_2 to remedy the potential security loss due to the choice of a small α_1.

While the intuition seems reasonable, it does not shed light on the choices of parameters, in particular, how parameters α_1 and α_2 (jointly) affect security. To this end, we consider the best known attacks and their variants against (A)LWE problems, and obtain the following conclusions: Let χ_{α_1} and χ_{α_2} be subgaussians with standard variances $\alpha_1, \alpha_2 \in \mathbb{R}$ respectively, then we have the following approximate relation between the hardness of ALWE and LWE: the hardness of ALWE with subgaussian standard variances $\alpha_1, \alpha_2 \in \mathbb{R}$ is polynomially equivalent to the hardness of LWE with subgaussian standard variance $\sqrt{\alpha_1\alpha_2}$. Clearly, the equivalence is trivial for $\alpha_1 = \alpha_2$. This confirms the feasibility of our idea: use a small α_1 to keep the probability ν of decryption failures small while pick a relatively larger α_2 remain the security of the resulting PKE scheme.

The above idea can be naturally generalized to the schemes based on the ring and module versions of LWE. Actually, we will use AMLWE for achieving a better trade-off between computational and communication costs.

3.2 The Construction

We now formally describe a CCA-secure KEM from AMLWE (and AMLWE-R). For ease of implementation, we will use centered binomial distributions instead of Gaussian distributions as in [3,6]. We first give an intermediate IND-CPA secure PKE, which is then transformed into an IND-CCA secure KEM by applying a tweaked Fujisaki-Okamoto (FO) transformation [14,18].

An IND-CPA Secure PKE. Let $n, q, k, \eta_1, \eta_2, d_t, d_u, d_v$ be positive integers. Let $\mathcal{H} : \{0,1\}^n \to R_q^{k \times k}$ be a hash function, which is modeled as a random oracle. The PKE scheme Π_{PKE} consists of three algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\Pi_{\text{PKE}}.\mathsf{KeyGen}(\kappa)$: randomly choose $\rho \xleftarrow{\$} \{0,1\}^n$, $\mathbf{s} \xleftarrow{\$} B_{\eta_1}^k$, $\mathbf{e} \xleftarrow{\$} B_{\eta_2}^k$, compute $\mathbf{A} = \mathcal{H}(\rho) \in R_q^{k \times k}, \mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e} \in R_q^k$ and $\bar{\mathbf{t}} = \lceil \mathbf{t} \rfloor_{q \to 2^{d_t}}$. Then, return the public key $pk = (\rho, \bar{\mathbf{t}})$ and the secret key $sk = \mathbf{s}$.

– $\mathit{\Pi}_{\text{PKE}}.\mathsf{Enc}(pk, \mu)$: given the public key $pk = (\rho, \bar{\mathbf{t}})$ and a plaintext $\mu \in R_2$, randomly choose $\mathbf{r} \overset{\$}{\leftarrow} B_{\eta_1}^k, \mathbf{e}_1 \overset{\$}{\leftarrow} B_{\eta_2}^k, e_2 \overset{\$}{\leftarrow} B_{\eta_2}$, compute $\mathbf{A} = \mathcal{H}(\rho)$, $\mathbf{u} = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1$, $v = \lceil \bar{\mathbf{t}} \rfloor_{2^{d_t} \to q}^T \mathbf{r} + e_2$, and return the ciphertext

$$C = (\bar{\mathbf{u}} = \lceil \mathbf{u} \rfloor_{q \to 2^{d_u}}, \bar{v} = \lceil v + \mu \cdot \lceil \frac{q}{2} \rfloor \rfloor_{q \to 2^{d_v}}).$$

– $\mathit{\Pi}_{\text{PKE}}.\mathsf{Dec}(sk, C)$: given the secret key $sk = \mathbf{s}$ and a ciphertext $C = (\bar{\mathbf{u}}, \bar{v})$, compute $z = \lceil \bar{v} \rfloor_{2^{d_v} \to q} - \mathbf{s}^T \lceil \bar{\mathbf{u}} \rfloor_{2^{d_u} \to q}$, output $\lceil z \rfloor_{q \to 2} = \lceil z \cdot \frac{2}{q} \rfloor \bmod 2$.

Let $\mathbf{c}_t \in R^k$ satisfy that

$$\lceil \bar{\mathbf{t}} \rfloor_{2^{d_t} \to q} = \lceil \lceil \mathbf{As} + \mathbf{e} \rfloor_{q \to 2^{d_t}} \rfloor_{2^{d_t} \to q} = \mathbf{As} + \mathbf{e} - \mathbf{c}_t.$$

Let $\mathbf{c}_u \in R^k$ satisfy that

$$\lceil \bar{\mathbf{u}} \rfloor_{2^{d_u} \to q} = \lceil \lceil \mathbf{A}^T\mathbf{r} + \mathbf{e}_1 \rfloor_{q \to 2^{d_u}} \rfloor_{2^{d_u} \to q} = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1 - \mathbf{c}_u.$$

Let $c_v \in R$ satisfy that

$$\begin{aligned} \lceil \bar{v} \rfloor_{2^{d_v} \to q} &= \lceil \lceil \lceil \bar{\mathbf{t}} \rfloor_{2^{d_t} \to q}^T \mathbf{r} + e_2 + \lceil q/2 \cdot \mu \rfloor_{q \to 2^{d_v}} \rfloor_{2^{d_v} \to q} \\ &= \lceil \bar{\mathbf{t}} \rfloor_{2^{d_t} \to q}^T \mathbf{r} + e_2 + \lceil q/2 \rfloor \cdot \mu - c_v \\ &= (\mathbf{As} + \mathbf{e} - \mathbf{c}_t)^T \mathbf{r} + e_2 + \lceil q/2 \rfloor \cdot \mu - c_v \\ &= (\mathbf{As} + \mathbf{e})^T \mathbf{r} + e_2 + \lceil q/2 \rfloor \cdot \mu - c_v - \mathbf{c}_t^T \mathbf{r}. \end{aligned}$$

Using the above equations, we have

$$\begin{aligned} z &= \lceil \bar{v} \rfloor_{2^{d_v} \to q} - \mathbf{s}^T \lceil \bar{\mathbf{u}} \rfloor_{2^{d_u} \to q} \\ &= \underbrace{\mathbf{e}^T\mathbf{r} + e_2 - c_v - \mathbf{c}_t^T\mathbf{r} - \mathbf{s}^T\mathbf{e}_1 + \mathbf{s}^T\mathbf{c}_u}_{= w} + \lceil q/2 \rfloor \cdot \mu \\ &= w + \lceil q/2 \rfloor \cdot \mu. \end{aligned}$$

It is easy to check that for any odd number q, we have that $\mu = \lceil z \rfloor_{q \to 2}$ holds as long as $\|w\|_\infty < \lceil q/4 \rfloor$. In Sect. 3.4, we will choose the parameters such that the decryption algorithm succeeds with overwhelming probability.

IND-CCA Secure KEM. Let $\mathsf{G} : \{0,1\}^* \to \{0,1\}^n$, and $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n \times \{0,1\}^n$ be two hash functions, which are modeled as random oracles. By applying a slightly tweaked Fujisaki-Okamoto (FO) transformation [14,18], we can transform the above IND-CPA secure PKE $\mathit{\Pi}_{\text{PKE}}$ into an IND-CCA secure KEM (with implicit rejection) $\mathit{\Pi}_{\text{KEM}} = (\mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ as follows.

– $\mathit{\Pi}_{\text{KEM}}.\mathsf{KeyGen}(\kappa)$: choose $z \overset{\$}{\leftarrow} \{0,1\}^n$, compute $(pk', sk') = \mathit{\Pi}_{\text{PKE}}.\mathsf{KeyGen}(\kappa)$. Then, return the public key $pk = pk'$ and the secret key $sk = (pk', sk', z)$.

- $\Pi_{\text{KEM}}.\mathsf{Encap}(pk)$: given the public key pk, randomly choose $\mu \xleftarrow{\$} \{0,1\}^n$, compute $\mu' = \mathsf{H}(\mu), (\bar{K}, r) = \mathsf{G}(\mu' \| \mathsf{H}(pk))$ $C = \Pi_{\text{PKE}}.\mathsf{Enc}(pk, \mu'; r)$ and $K = \mathsf{H}(\bar{K} \| \mathsf{H}(C))$, where the notation $\Pi_{\text{PKE}}.\mathsf{Enc}(pk, \mu'; r)$ denotes running the algorithm $\Pi_{\text{PKE}}.\mathsf{Enc}(pk, \mu')$ with fixed randomness r. Finally, return the ciphertext C and the encapsulated key K.
- $\Pi_{\text{KEM}}.\mathsf{Decap}(sk, C)$: given the secret key $sk = (pk', sk', z)$ and a ciphertext C, compute $\mu' = \Pi_{\text{KEM}}.\mathsf{Dec}(sk', C)$ and $(\bar{K}', r') = \mathsf{G}(\mu' \| \mathsf{H}(pk')), C' = \Pi_{\text{KEM}}.\mathsf{Enc}(pk, \mu'; r')$. If $C = C'$, return $K = \mathsf{H}(\bar{K}' \| \mathsf{H}(C))$, else return $\mathsf{H}(z \| \mathsf{H}(C))$.

3.3 Provable Security

In the full version [32], we will show that under the hardness of the AMLWE problem and its rounding variant AMLWE-R (which is needed for compressing the public key, see Appendix A), our scheme Π_{PKE} is provably IND-CPA secure. Formally, we have the following theorem.

Theorem 1. *Let $\mathcal{H} : \{0,1\}^n \to R_q^{k \times k}$ be a random oracle. If both problems* $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ *and* $\text{AMLWE-R}_{n,q,2^{d_t},k,k,\eta_1,\eta_2}$ *are hard, then the scheme Π_{PKE} is IND-CPA secure.*

Since Π_{KEM} is obtained by applying a slightly tweaked Fujisaki-Okamoto (FO) transformation [14,18] to the PKE scheme Π_{PKE}, given the results in [6,18] and Theorem 1, we have the following theorem.

Theorem 2. *Under the* AMLWE *assumption and the* AMLWE-R *assumption, Π_{KEM} is IND-CCA secure in the random oracle model.*

Notice that the algorithm Decap will always return a random "session key" even if the check fails (i.e., implicit rejection). Furthermore, the paper [20] showed that if the underlying PKE is IND-CPA secure, then the resulting KEM with implicit rejection obtained by using the FO transformation is also IND-CCA secure in the quantum random oracle model (QROM). Given the results in [20] and Theorem 1, we have the following theorem.

Theorem 3. *Under the* AMLWE *assumption and the* AMLWE-R *assumption, Π_{KEM} is IND-CCA secure in the QROM.*

3.4 Choices of Parameters

In Table 5, we give three sets of parameters (namely, Π_{KEM}-512, Π_{KEM}-768 and Π_{KEM}-1024) for Π_{KEM}, aiming at providing quantum security of at least 80, 128 and 192 bits, respectively. These parameters are carefully chosen such that the decryption failure probabilities (i.e., 2^{-82}, 2^{-128} and 2^{-211}, respectively) are commensurate with the respective targeted security strengths. A concrete estimation of the security strength provided by the parameter sets will be given

Table 5. Parameters sets for Π_{KEM}

Parameters	(n, k, q)	(η_1, η_2)	(d_t, d_u, d_v)	$\lvert pk\rvert$	$\lvert sk\rvert$	$\lvert C\rvert$	$\lvert ss\rvert$	Dec. Fail.	Quant. Sec.
Π_{KEM}-512	$(256, 2, 7681)$	$(2, 12)$	$(10, 9, 3)$	672	1568	672	32	2^{-82}	100
Π_{KEM}-512†	$(256, 2, 3329)$	$(1, 4)$	$(-, 8, 4)$	800	1632	640	32	2^{-82}	99
Π_{KEM}-768	$(256, 3, 7681)$	$(1, 4)$	$(9, 9, 4)$	896	2208	992	32	2^{-128}	147
Π_{KEM}-768†	$(256, 3, 3329)$	$(1, 2)$	$(-, 9, 3)$	1184	2400	960	32	2^{-130}	157
Π_{KEM}-1024	$(512, 2, 12289)$	$(2, 8)$	$(11, 10, 4)$	1472	3392	1536	64	2^{-211}	213
Π_{KEM}-1024†	$(512, 2, 7681)$	$(1, 4)$	$(-, 9, 5)$	1728	3648	1472	64	2^{-198}	206

in Sect. 5. Among them, Π_{KEM}-768 is the recommended parameter set. By the quantum searching algorithm [17], 2κ-bit randomness/session key can only provide at most κ security. Even if the Grover algorithm cannot provide a quadratic speedup over classical algorithms in practice, we still set Π_{KEM}-1024 to support an encryption of 64-bytes (512-bit) randomness/session key, aiming at providing a matching security strength, say, more than 210-bit estimated quantum security. Note that Π_{KEM}-512 and Π_{KEM}-768 only support an encryption of 32-byte (256-bit) session key.

We implemented our Π_{KEM} on a 64-bit Ubuntu 14.4 LTS ThinkCenter desktop (equipped with Intel Core-i7 4790 3.6 GHz CPU and 4 GB memory). Particularly, the codes are mainly written using the C language, with partially optimized codes using AVX2 instructions to speedup some basic operations such as NTT operation and vector multiplications. The average number of CPU cycles (averaged over 10000 times) for running each algorithm is given in Table 1.

4 An Improved Signature from AMLWE and AMSIS

Our signature scheme is based on the "Fiat-Shamir with Aborts" technique [23], and bears most resemblance to Dilithium in [12]. The main difference is that our scheme uses the asymmetric MLWE and MSIS problems, which provides a flexible way to make a better trade-off between performance and security.

4.1 Design Rationale

Several lattice-based signature schemes were obtained by applying the Fiat-Shamir heuristic [13] to three-move identification schemes. For any positive integer n and q, let $R = \mathbb{Z}[x]/(x^n + 1)$ (resp., $R_q = \mathbb{Z}_q[x]/(x^n + 1)$). Let $\mathsf{H} : \{0,1\}^* \to R_2$ be a hash function. Let k, ℓ, η be positive integers, and $\gamma, \beta > 0$ be reals. We first consider an identification protocol between two users A and B based on the $\mathrm{MSIS}^{\infty}_{n,q,k,\ell,\beta}$ problem. Formally, user A owns a pair of public key $pk = (\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{x}) \in R_q^{k\times\ell} \times R_q^k$ and secret key $sk = \mathbf{x} \in R_q^\ell$. In order to convince another user B (who knows the public key pk) of his ownership of sk, A and B can execute the following protocol: (1) A first chooses a vector $\mathbf{y} \in R^\ell$ from some distribution, and sends $\mathbf{w} = \mathbf{A}\mathbf{y}$ to user B; (2) B randomly chooses a bit $c \in R_q$, and sends it as a challenge to A; (3) A computes $\mathbf{z} := \mathbf{y} + c\mathbf{x}$ and sends it back to B; B will accept the response $\mathbf{z}$ by check if $\mathbf{Az} = \mathbf{w} + c\mathbf{t}$.

For the soundness (i.e., user A cannot cheat user B), B also has to make sure that $\beta_2 = \|\mathbf{z}\|_\infty$ is sufficiently small (to ensure that the $\mathrm{MSIS}^\infty_{n,q,k,\ell,\beta}$ problem is hard), otherwise anyone can easily complete the proof by solving a linear equation. Moreover, we require that $\beta_1 = \|\mathbf{x}\|_\infty$ is sufficiently small and $\|\mathbf{y}\|_\infty \gg \|\mathbf{x}\|_\infty$ (and thus $\beta_2 \gg \beta_1$) holds to prevent user B from recovering the secret $\mathbf{x}$ from the public key pk or the response $\mathbf{z}$. Typically, we should require $\beta_2/\beta_1 > 2^{\omega(\log \kappa)}$, where κ is the security parameter. This means that the identification protocol as well as its derived signature from the Fiat-Shamir heuristic will have a very large parameter size. To solve this problem, Lyubashevsky [23,24] introduce the rejection sampling, which allows A to abort and restart the protocol (by choosing another $\mathbf{y}$) if he thinks $\mathbf{z}$ might leak the information of $\mathbf{x}$. This technique could greatly reduce the size of $\mathbf{z}$ (since it allows to set $\beta_2/\beta_1 = \mathsf{poly}(\kappa)$), but the cost is painful for an interactive identification protocol. Fortunately, this technique will only increase the computation time of the signer when we transform the identification protocol into a signature scheme.

For any positive integer η, S_η denotes the set of elements of R that each coefficient is taken from $\{-\eta, -\eta + 1 \ldots, \eta\}$. By the Fiat-Shamir heuristic, one can construct a signature scheme from the MSIS problem as follows:

- **Key generation**: randomly choose $\mathbf{A} \overset{\$}{\leftarrow} R_q^{k\times\ell}, \mathbf{x} \overset{\$}{\leftarrow} S_\eta^\ell$, and compute $\mathbf{t} = \mathbf{Ax}$. Return the public key $pk = (\mathbf{A}, \mathbf{t})$ and secret key $sk = (\mathbf{x}, pk)$.
- **Signing**: given the secret key $sk = (\mathbf{x}, pk)$ and a message $\mu \in \{0,1\}^*$,
  1. randomly choose $\mathbf{y} \overset{\$}{\leftarrow} S^\ell_{\gamma-1}$;
  2. compute $\mathbf{w} = \mathbf{Ay}$ and $c = \mathsf{H}(\mathbf{w}\|\mu)$;
  3. compute $\mathbf{z} = \mathbf{y} + c\mathbf{x}$;
  4. If $\|\mathbf{z}\|_\infty \geq \gamma - \beta$, restart the computation from step 1), where β is a bound such that $\|c\mathbf{x}\|_\infty \leq \beta$ for all possible c and $\mathbf{x}$. Otherwise, return the signature $\sigma = (\mathbf{z}, c)$.
- **Verification**: given the public key $pk = (\mathbf{A}, \mathbf{t})$, a message $\mu \in \{0,1\}^*$ and a signature $\sigma = (\mathbf{z}, c)$, return 1 if $\|\mathbf{z}\|_\infty < \gamma - \beta$ and $c = \mathsf{H}(\mathbf{Az} - c\mathbf{t}\|\mu)$, and 0 otherwise.

Informally, we require the $\mathrm{MSIS}^\infty_{n,q,k,\ell,\eta}$ problem to be hard for the security of the secret key (i.e., it is computationally infeasible to compute sk from pk). Moreover, we also require the $\mathrm{MSIS}^\infty_{n,q,k,\ell,2\gamma}$ problem to be hard for the unforgeability of signatures (i.e., it is computationally infeasible to forge a valid signature). Since $\|c\mathbf{x}\|_\infty \leq \beta$, for any $(c, \mathbf{x})$ and $\mathbf{z}$ output by the signing algorithm there always exists a $\mathbf{y} \in S^\ell_\gamma$ such that $\mathbf{z} = \mathbf{y} + c\mathbf{x}$, which guarantees that the signature will not leak the information of the secret key. In terms of efficiency, the signing algorithm will repeat about $\left(\frac{2(\gamma-\beta)-1}{2\gamma-1}\right)^{-n\cdot\ell}$ times to output a signature, and the signature size is about $n\ell\lceil\log_2(2(\gamma-\beta)-1)\rceil + n$. Clearly, we wish to use a small ℓ for better efficiency, but the hardness of the underlying MSIS problems require a relatively large ℓ.

To mediate the above conflict, one can use the MLWE problem, which can be seen as a special MSIS problem, to reduce the size of the key and signature. Formally, we can obtain the following improved signature scheme:

- **Key generation**: randomly choose $\mathbf{A} \xleftarrow{\$} R_q^{k\times\ell}$, and $\mathbf{s}_1 \xleftarrow{\$} S_\eta^\ell, \mathbf{s}_2 \xleftarrow{\$} S_\eta^k$, compute $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$. Return the public key $pk = (\mathbf{A}, \mathbf{t})$ and secret key $sk = (\mathbf{s}_1, \mathbf{s}_2, pk)$.
- **Signing**: given the secret key $sk = (\mathbf{s}_1, \mathbf{s}_2, pk)$ and a message $\mu \in \{0,1\}^*$,
  1. randomly choose $\mathbf{y} \xleftarrow{\$} S_{\gamma-1}^{\ell+k}$;
  2. compute $\mathbf{w} = (\mathbf{A}\|\mathbf{I}_k)\mathbf{y}$ and $c = \mathsf{H}(\mathbf{w}\|\mu)$;
  3. compute $\mathbf{z} = \mathbf{y} + c\begin{pmatrix}\mathbf{s}_1\\ \mathbf{s}_2\end{pmatrix}$;
  4. If $\|\mathbf{z}\|_\infty \geq \gamma - \beta$, restart the computation from step (1), where β is a bound such that $\left\|c\begin{pmatrix}\mathbf{s}_1\\ \mathbf{s}_2\end{pmatrix}\right\|_\infty \leq \beta$ holds for all possible $c, \mathbf{s}_1, \mathbf{s}_2$. Otherwise, output the signature $\sigma = (\mathbf{z}, c)$.
- **Verification**: given the public key $pk = (\mathbf{A}, \mathbf{t})$, a message $\mu \in \{0,1\}^*$ and a signature $\sigma = (\mathbf{z}, c)$, return 1 if $\|\mathbf{z}\|_\infty < \gamma - \beta$ and $c = \mathsf{H}((\mathbf{A}\|\mathbf{I}_k)\mathbf{z} - c\mathbf{t}\|\mu)$, otherwise return 0.

Furthermore, since $\mathbf{w} = (\mathbf{A}\|\mathbf{I}_k)\mathbf{y} = \mathbf{A}\mathbf{y}_1 + \mathbf{y}_2$ where $\mathbf{y} = (\mathbf{y}_1^T, \mathbf{y}_2^T)$ and $\gamma \ll q$, we have that the higher bits of (each coefficient of) $\mathbf{w}$ is almost determined by high order bits of (the corresponding coefficient of) $\mathbf{A}\mathbf{y}_1$. This fact has been utilized by [5,12] to compress the signature size. Formally, denote $\mathsf{HighBits}(\mathbf{z}, 2\gamma_2)$ and $\mathsf{LowBits}(\mathbf{z}, 2\gamma_2)$ be polynomial vector defined by the high order bits and low order bits of a polynomial vector $\mathbf{z} \in R_q^k$ related to a parameter γ_2. We can obtain the following signature scheme:

- **Key generation**: randomly choose $\mathbf{A} \xleftarrow{\$} R_q^{k\times\ell}$, and $\mathbf{s}_1 \xleftarrow{\$} S_\eta^\ell, \mathbf{s}_2 \xleftarrow{\$} S_\eta^k$, compute $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$. Return the public key $pk = (\mathbf{A}, \mathbf{t})$ and secret key $sk = (\mathbf{s}_1, \mathbf{s}_2, pk)$.
- **Signing**: given the secret key $sk = (\mathbf{s}_1, \mathbf{s}_2, pk)$ and a message $\mu \in \{0,1\}^*$,
  1. randomly choose $\mathbf{y} \xleftarrow{\$} S_{\gamma_1-1}^\ell$;
  2. compute $\mathbf{w} = \mathbf{A}\mathbf{y}$ and $c = \mathsf{H}(\mathsf{HighBits}(\mathbf{w}, 2\gamma_2)\|\mu)$;
  3. compute $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$;
  4. If $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\mathsf{LowBits}(\mathbf{A}\mathbf{y} - c\mathbf{s}_2, 2\gamma_2) \geq \gamma_2 - \beta$, restart the computation from step 1), where β is a bound such that $\|c\mathbf{s}_1\|_\infty, \|c\mathbf{s}_2\|_\infty \leq \beta$ hold for all possible $c, \mathbf{s}_1, \mathbf{s}_2$. Otherwise, output the signature $\sigma = (\mathbf{z}, c)$.
- **Verification**: given the public key $pk = (\mathbf{A}, \mathbf{t})$, a message $\mu \in \{0,1\}^*$ and a signature $\sigma = (\mathbf{z}, c)$, return 1 if $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ and $c = \mathsf{H}(\mathsf{HighBits}(\mathbf{A}\mathbf{z} - c\mathbf{t}, 2\gamma_2)\|\mu)$, otherwise return 0.

Essentially, the checks in step (4) are used to ensure that (1) the signature $(\mathbf{z}, c)$ will not leak the information of $\mathbf{s}_1$ and $\mathbf{s}_2$; and (2) $\mathsf{HighBits}(\mathbf{A}\mathbf{z} - c\mathbf{t}, 2\gamma_2) = \mathsf{HighBits}(\mathbf{A}\mathbf{y} - c\mathbf{s}_2, 2\gamma_2) = \mathsf{HighBits}(\mathbf{w}, 2\gamma_2)$ (note that $\mathbf{w} = \mathbf{A}\mathbf{y} = \mathbf{A}\mathbf{y} - c\mathbf{s}_2 +$

$c\mathbf{s}_2$, $\mathsf{LowBits}(\mathbf{Ay} - c\mathbf{s}_2, 2\gamma_2) < \gamma_2 - \beta$ and $\|c\mathbf{s}_2\|_\infty \leq \beta$). By setting $\gamma_1 = 2\gamma_2$, we require the $\mathrm{MLWE}_{n,k,\ell,q,\eta}$ problem and the (variant of) $\mathrm{MSIS}^\infty_{n,k,(\ell+k+1),q,2\gamma_1+2}$ problem to be hard to ensure the security of the secret key and the unforgeability of the signature, respectively.

By a careful examination on the above scheme, one can find that the computational efficiency of the signing algorithm is determined by the expected number of repetitions in step (4):

$$\underbrace{\left(\frac{2(\gamma_1-\beta)-1}{2\gamma_1-1}\right)^{-n\cdot\ell}}_{=N_1} \cdot \underbrace{\left(\frac{2(\gamma_2-\beta)-1}{2\gamma_2-1}\right)^{-n\cdot k}}_{=N_2},$$

where N_1 and N_2 are determined by the first and second checks in step (4), respectively. Clearly, it is possible to modify N_1 and N_2 while keeping the total number of repetitions $N = N_1 \cdot N_2$ unchanged. Note that the size of the signature is related to γ_1 and is irrelevant to γ_2, which means that a shorter signature can be obtained by using a smaller γ_1. However, simply using a smaller γ_1 will also give a bigger N_1, and thus a worse computational efficiency. In order to obtain a short signature size without (significantly) affecting the computational efficiency:

- We use the AMLWE problem for the security of the secret key, which allows us to use a smaller γ_1 by reducing $\|\mathbf{s}_1\|_\infty$ (and thus $\beta = \|c\mathbf{s}_1\|_\infty$ in the expression of N_1);
- We use the AMSIS problem for the unforgeability of the signatures, which further allows us to use a smaller γ_1 by increasing γ_2 to keep $N = N_1 \cdot N_2$ unchanged.

Note that reducing $\|\mathbf{s}_1\|_\infty$ (by choosing a smaller η_1) may weaken the hardness of the underlying AMLWE problem (if we do not change other parameters). We choose to increase η_2 (and thus $\|\mathbf{s}_2\|_\infty$) to remain the hardness. Similarly, increasing γ_2 will weaken the hardness of the underlying AMSIS problem, and we choose to reduce γ_1 to remain the hardness. Both strategies crucially rely on the asymmetries of the underlying problems.

4.2 The Construction

Let $n, k, \ell, q, \eta_1, \eta_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \omega \in \mathbb{Z}$ be positive integers. Let $R = \mathbb{Z}[x]/(x^n + 1)$ and $R_q = \mathbb{Z}_q[x]/(x^n + 1)$. Denote B_{60} as the set of elements of R that have 60 coefficients are either -1 or 1 and the rest are 0, and $|B_{60}| = 2^{60} \cdot \binom{n}{60}$. When $n = 256$, $|B_{60}| > 2^{256}$. Let $\mathsf{H}_1 : \{0,1\}^{256} \rightarrow R_q^{k\times\ell}, \mathsf{H}_2 : \{0,1\}^* \rightarrow \{0,1\}^{384}$, $\mathsf{H}_3 : \{0,1\}^* \rightarrow S^\ell_{\gamma_1-1}$ and $\mathsf{H}_4 : \{0,1\}^* \rightarrow B_{60}$ be four hash functions. We now present the description of our scheme $\Pi_{\mathrm{SIG}} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$:

- $\Pi_{\mathrm{SIG}}.\mathsf{KeyGen}(\kappa)$: first randomly choose $\rho, K \xleftarrow{\$} \{0,1\}^{256}$, $\mathbf{s}_1 \xleftarrow{\$} S^\ell_{\eta_1}, \mathbf{s}_2 \xleftarrow{\$} S^k_{\eta_2}$. Then, compute $\mathbf{A} = \mathsf{H}_1(\rho) \in R_q^{k\times\ell}, \mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 \in R_q^k, (\mathbf{t}_1, \mathbf{t}_0) = \mathsf{Power2Round}_q(\mathbf{t}, d)$ and $tr = \mathsf{H}_2(\rho\|\mathbf{t}_1) \in \{0,1\}^{384}$. Finally, return the public key $pk = (\rho, \mathbf{t}_1)$ and secret key $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$.

- $\Pi_{\text{SIG}}.\mathsf{Sign}(sk, M)$: given $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$ and a message $M \in \{0,1\}^*$, first compute $\mathbf{A} = \mathsf{H}_1(\rho) \in R_q^{k\times\ell}$, $\mu = \mathsf{H}_2(tr\|M) \in \{0,1\}^{384}$, and set $ctr = 0$. Then, perform the following computations:
  1. $\mathbf{y} = \mathsf{H}_3(K\|\mu\|ctr) \in S_{\gamma_1-1}^{\ell}$ and $\mathbf{w} = \mathbf{A}\mathbf{y}$;
  2. $\mathbf{w}_1 = \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$ and $c = \mathsf{H}_4(\mu\|\mathbf{w}_1) \in B_{60}$;
  3. $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$ and $\mathbf{u} = \mathbf{w} - c\mathbf{s}_2$;
  4. $(\mathbf{r}_1, \mathbf{r}_0) = \mathsf{Decompose}_q(\mathbf{u}, 2\gamma_2)$;
  5. if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta_1$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta_2$ or $\mathbf{r}_1 \neq \mathbf{w}_1$, then set $ctr = ctr + 1$ and restart the computation from step (1);
  6. compute $\mathbf{v} = c\mathbf{t}_0$ and $\mathbf{h} = \mathsf{MakeHint}_q(-\mathbf{v}, \mathbf{u} + \mathbf{v}, 2\gamma_2)$;
  7. if $\|\mathbf{v}\|_\infty \geq \gamma_2$ or the number of 1's in $\mathbf{h}$ is greater than ω, then set $ctr = ctr + 1$ and restart the computation from step 1);
  8. return the signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$.
- $\Pi_{\text{SIG}}.\mathsf{Verify}(pk, M, \sigma)$: given the public key $pk = (\rho, \mathbf{t}_1)$, a message $M \in \{0,1\}^*$ and a signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$, first compute $\mathbf{A} = \mathsf{H}_1(\rho) \in R_q^{k\times\ell}, \mu = \mathsf{H}_2(\mathsf{H}_2(pk)\|M) \in \{0,1\}^{384}$. Let $\mathbf{u} = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, \mathbf{w}_1' = \mathsf{UseHints}_q(\mathbf{h}, \mathbf{u}, 2\gamma_2)$ and $c' = \mathsf{H}_4(\mu\|\mathbf{w}_1')$. Finally, return 1 if $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1, c = c'$ and the number of 1's in $\mathbf{h}$ is $\leq \omega$, otherwise return 0.

We note that the hash function H_3 is basically used to make the signing algorithm Sign deterministic, which is needed for a (slightly) tighter security proof in the quantum random oracle model. One can remove H_3 by directly choosing $\mathbf{y} \overset{\$}{\leftarrow} S_{\gamma_1-1}^{\ell}$ at random, and obtain a probabilistic signing algorithm. We also note that the hash function H_4 can be constructed by using an extendable output function such as SHAKE-256 [29] and a so-called "inside-out" version of Fisher-Yates shuffle algorithm [21]. The detailed constructions of hash functions H_3 and H_4 can be found in [12].

Correctness. Note that if $\|c\mathbf{t}_0\|_\infty < \gamma_2$, by Lemma 1 we have $\mathsf{UseHint}_q(\mathbf{h}, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2) = \mathsf{HighBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$. Since $\mathbf{w} = \mathbf{A}\mathbf{y}$ and $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$, we have that

$$\begin{gathered}\mathbf{w} - c\mathbf{s}_2 = \mathbf{A}\mathbf{y} - c\mathbf{s}_2 = \mathbf{A}(\mathbf{z} - c\mathbf{s}_1) - c\mathbf{s}_2 = \mathbf{A}\mathbf{z} - c\mathbf{t}, \\ \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0 = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d,\end{gathered}$$

where $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + t_0$. Therefore, the verification algorithm computes

$$\mathsf{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2) = \mathsf{HighBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2).$$

As the signing algorithm checks that $\mathbf{r}_1 = \mathbf{w}_1$, this is equivalent to

$$\mathsf{HighBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2) = \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2).$$

Hence, the $\mathbf{w}_1$ computed by the verification algorithm is the same as that of the signing algorithm, and thus the verification algorithm will always return 1.

Number of Repetitions. Since our signature scheme uses the rejection sampling [23, 24] to generate $(\mathbf{z}, \mathbf{h})$, the efficiency of the signing algorithm is determined by the number of repetitions that will be caused by steps (5) and (7) of the signing algorithm. We first estimate the probability that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$ holds in step (5). Assuming that $\|c\mathbf{s}_1\|_\infty \leq \beta_1$ holds, then we always have $\|\mathbf{z}\|_\infty \leq \gamma_1 - \beta_1 - 1$ whenever $\|\mathbf{y}\|_\infty \leq \gamma_1 - 2\beta_1 - 1$. The size of this range is $2(\gamma_1 - \beta_1) - 1$. Note that each coefficient of $\mathbf{y}$ is chosen randomly from $2\gamma_1 - 1$ possible values. That is, for a fixed $c\mathbf{s}_1$, each coefficient of vector $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$ has $2\gamma_1 - 1$ possibilities. Therefore, the probability that $\|\mathbf{z}\|_\infty \leq \gamma_1 - \beta_1 - 1$ is

$$\left(\frac{2(\gamma_1 - \beta_1) - 1}{2\gamma_1 - 1}\right)^{n\cdot\ell} = \left(1 - \frac{\beta_1}{\gamma_1 - 1/2}\right)^{n\cdot\ell} \approx e^{-n\ell\beta_1/\gamma_1}.$$

Now, we estimate the probability that

$$\|\mathbf{r}_0\|_\infty = \|\mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)\|_\infty < \gamma_2 - \beta_2$$

holds in step (5). If we (heuristically) assume that each coefficient of $\mathbf{r}_0$ is uniformly distributed modulo $2\gamma_2$, the probability that $\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta_2$ is

$$\left(\frac{2(\gamma_2 - \beta_2) - 1}{2\gamma_2}\right)^{n\cdot k} \approx e^{-nk\beta_2/\gamma_2}.$$

By Lemma 2, if $\|c\mathbf{s}_2\|_\infty \leq \beta_2$, then $\|\mathbf{r}_0\|_\infty < \gamma_2 - \beta_2$ implies that $\mathbf{r}_1 = \mathbf{w}_1$. This means that the overall probability that step (5) will not cause a repetition is

$$\approx e^{-n(\ell\beta_1/\gamma_1 + k\beta_2/\gamma_2)}.$$

Finally, under our choice of parameters, the probability that step (7) of the signing algorithm will cause a repetition is less than 1%. Thus, the expected number of repetitions is roughly $e^{n(\ell\beta_1/\gamma_1 + k\beta_2/\gamma_2)}$.

4.3 Provable Security

In the full version [32], we show that under the hardness of the AMLWE problem and a rounding variant AMSIS-R of AMSIS (which is needed for compressing the public key, see Appendix A), our scheme Π_{SIG} is provably SUF-CMA secure in the ROM. Formally, we have the following theorem.

Theorem 4. *If* $\mathsf{H}_1 : \{0,1\}^{256} \to R_q^{k\times\ell}$ *and* $\mathsf{H}_4 : \{0,1\}^* \to B_{60}$ *are random oracles, the outputs of* $\mathsf{H}_3 : \{0,1\}^* \to S_{\gamma_1 - 1}^{\ell}$ *are pseudo-random, and* $\mathsf{H}_2 : \{0,1\}^* \to \{0,1\}^{384}$ *is a collision-resistant hash function, then* Π_{SIG} *is SUF-CMA secure under the* $\mathrm{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ *and* $\mathrm{AMSIS\text{-}R}^{\infty}_{n,q,d,k,\ell,4\gamma_2+2,2\gamma_1}$ *assumptions.*

Furthermore, under an interactive variant SelfTargetAMSIS of the AMSIS problem (which is an asymmetric analogue of the SelfTargetMSIS problem introduced by Ducas et al. [12]), we can also prove that our scheme Π_{SIG} is provably SUF-CMA secure. Formally, we have that following theorem.

Theorem 5. *In the quantum random oracle model (QROM), signature scheme Π_{SIG} is SUF-CMA secure under the following assumptions:* $\mathrm{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$*,* $\mathrm{AMSIS}^{\infty}_{n,q,d,k,\ell,4\gamma_2+2,2(\gamma_1-\beta_1)}$ *and* $\mathrm{SelfTargetAMSIS}^{\infty}_{\mathsf{H}_4,n,q,k,\ell_1,\ell_2,4\gamma_2,(\gamma_1-\beta_1)}$*.*

4.4 Choices of Parameters

In Table 6, we provide three sets of parameters (i.e., Π_{SIG}-1024, Π_{SIG}-1280 and Π_{SIG}-1536) for our signature scheme Π_{SIG}, which provide 80-bit, 128-bit and 160-bit quantum security, respectively (corresponding to 98-bit, 141-bit and 178-bit classical security, respectively). A concrete estimation of the security provided by the parameter sets will be given in Sect. 5. Among them, Π_{SIG}-1280 is the recommended parameter set.

Table 6. Parameters for Π_{SIG} (The column "Reps." indicates the excepted number of repetitions that the signing algorithm takes to output a valid signature)

Parameters	(k, ℓ, q, d, ω)	(η_1, η_2)	(β_1, β_2)	(γ_1, γ_2)	Reps.	Quant. Sec.
Π_{SIG}-1024	$(4, 3, 2021377, 13, 80)$	$(2, 3)$	$(120, 175)$	$(131072, 168448)$	5.86	90
Π_{SIG}-1280	$(5, 4, 3870721, 14, 96)$	$(2, 5)$	$(120, 275)$	$(131072, 322560)$	7.61	128
Π_{SIG}-1536	$(6, 5, 3870721, 14, 120)$	$(1, 5)$	$(60, 275)$	$(131072, 322560)$	6.67	163

Our scheme Π_{SIG} under the same machine configuration as in Sect. 3.4 is implemented using standard C, where some partial optimization techniques (e.g., AVX2 instructions) are adopted to speedup basic operations such as NTT operation. The average CPU cycles (averaged over 10000 times) needed for running the algorithms are given in Table 3.

5 Known Attacks Against AMLWE and AMSIS

Solvers for LWE mainly include primal attacks, dual attacks (against the underlying lattice problems) and direct solving algorithms such as BKW and Arora-Ge [2]. BKW and Arora-Ge attacks need sub-exponentially (or even exponentially) many samples, and thus they are not relevant to the public-key cryptography scenario where only a restricted amount of samples is available. Therefore, for analyzing and evaluating practical lattice-based cryptosystems, we typically consider only primal attacks and dual attacks. Further, these two attacks, which are the currently most relevant and effective, seem not to have additional advantages in solving RLWE/MLWE over standard LWE. Thus, when analyzing RLWE or MLWE based cryptosystems, one often translates RLWE/MLWE instances to the corresponding LWE counterparts [6,12] and then applies the attacks. In particular, one first transforms $\mathrm{AMLWE}_{n,q,k,\ell,\alpha_1,\alpha_2}$ into $\mathrm{ALWE}_{nk,q,k\ell,\alpha_1,\alpha_2}$, and then applies, generalizes and optimizes the LWE solving algorithms to ALWE. Since any bounded centrally symmetric distribution can be regarded as subgaussian for a certain parameter, for simplicity and without loss of generality, we consider the case that secret vector and error vector in $\text{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ are sampled from subgaussians with parameters α_1 and α_2 respectively. Formally, the problem is to recover $\mathbf{s}$ from samples

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m,$$

where $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \leftarrow \chi_{\alpha_1}^n$ and $\mathbf{e} \leftarrow \chi_{\alpha_2}^m$.

In the full version [32], we will not only consider the traditional primal attack and dual attack against ALWE, but also consider two variants of primal attack and three variants of dual attack, which are more efficient to solve the ALWE problem by taking into account the asymmetry of ALWE.

As for the best known attacks against (A)SIS, the BKZ lattice basis reduction algorithm and its variants are more useful for solving the ℓ_2-norm (A)SIS problem than the ℓ_∞-norm counterpart. Note that a solution $\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{Z}^{m_1+m_2}$ to the infinity-norm ASIS instance $\mathbf{A} \in \mathbb{Z}_q^{n\times(m_1+m_2-n)}$, where $(\mathbf{I}_n\|\mathbf{A})\mathbf{x} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq \max(\beta_1, \beta_2) < q$, may have $\|\mathbf{x}\| > q$, whose ℓ_2-norm is even larger than that of a trivial solution $\mathbf{u} = (q, 0, \ldots, 0)^T$. We will follow [12] to solve the ℓ_∞-norm SIS problem. Further, we can always apply an ℓ_2-norm SIS solve to the ℓ_∞-norm SIS problem due to the relation $\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|$. Hereafter we refer to the above two algorithms as ℓ_∞-norm and ℓ_2-norm attacks respectively, and use them to estimate the concrete complexity of solving $\text{ASIS}^\infty_{n,q,m_1,m_2,\beta_1,\beta_2}$. As before, when analyzing RSIS or MSIS based cryptosystems, one often translates RSIS/MSIS instances to the corresponding SIS counterparts [12] and then applies the attacks.

In the full version [32], we will not only consider the traditional ℓ_2 norm attack and ℓ_∞ norm attack against ASIS, but also consider one variant of ℓ_2 norm attack and two variants of ℓ_∞ norm attack, which are more efficient to solve the ASIS problem by taking into consideration the asymmetry of ASIS.

In the following two subsections, we will summarize those attacks against our Π_{KEM} and Π_{SIG} schemes.

5.1 Concrete Security of Π_{KEM}

The complexity varies for the type of attacks, the number m of samples used and choice of $b \in \mathbb{Z}$ to run the BKZ-b algorithm. Therefore, in order to obtain an overall security estimation *sec* of the Π_{KEM} under the three proposed parameter settings, we enumerate all possible values of m (the number of ALWE samples) and b to reach a conservative estimation about the computational complexity of primal attacks and dual attacks, by using a python script (which is planned to be uploaded together with the implementation of our schemes to a public repository later). Tables 7 and 8 estimate the complexities of the three parameter sets against primal attacks and dual attacks by taking the minimum of *sec* over all possible values of (m, b). Taking into account the above, Table 9 shows the overall security of Π_{KEM}.

Table 7. The security of Π_{KEM} against primal attacks

Parameters	Attack model	Traditional (m, b, sec)	Variant 1 (m, b, sec)	Variant 2 (m, b, sec)
Π_{KEM}-512	Classical	$(761, 390, 114)$	$(531, 405, 118)$	$(\mathbf{476}, \mathbf{385}, \mathbf{112})$
	Quantum	$(761, 390, 103)$	$(531, 405, 107)$	$(\mathbf{476}, \mathbf{385}, \mathbf{102})$
Π_{KEM}-768	Classical	$(1021, 640, 187)$	$(646, 575, 168)$	$(\mathbf{556}, \mathbf{560}, \mathbf{163})$
	Quantum	$(1021, 640, 169)$	$(646, 575, 152)$	$(\mathbf{556}, \mathbf{560}, \mathbf{148})$
Π_{KEM}-1024	Classical	$(1526, 825, 241)$	$(886, 835, 244)$	$(\mathbf{786}, \mathbf{815}, \mathbf{238})$
	Quantum	$(1531, 825, 218)$	$(886, 835, 221)$	$(\mathbf{786}, \mathbf{815}, \mathbf{216})$

Table 8. The security of Π_{KEM} against dual attacks

Parameters	Attack model	Traditional (m, b, sec)	Variation 1 (m, b, sec)	Variation 2 (m, b, sec)	Variation 3 (m, b, sec)
Π_{KEM}-512	Classical	$(766, 385, 112)$	$(736, 395, 115)$	$(595, 380, 111)$	$(\mathbf{711}, \mathbf{380}, \mathbf{111})$
	Quantum	$(766, 385, 102)$	$(736, 395, 104)$	$(\mathbf{596}, \mathbf{380}, \mathbf{100})$	$(\mathbf{711}, \mathbf{380}, \mathbf{100})$
Π_{KEM}-768	Classical	$(1021, 620, 181)$	$(881, 570, 166)$	$(\mathbf{586}, \mathbf{555}, \mathbf{162})$	$(\mathbf{776}, \mathbf{555}, \mathbf{162})$
	Quantum	$(1021, 620, 164)$	$(881, 570, 151)$	$(\mathbf{586}, \mathbf{555}, \mathbf{147})$	$(\mathbf{776}, \mathbf{555}, \mathbf{147})$
Π_{KEM}-1024	Classical	$(1531, 810, 237)$	$(981, 810, 239)$	$(906, 805, 236)$	$(\mathbf{1171}, \mathbf{805}, \mathbf{235})$
	Quantum	$(1531, 810, 215)$	$(981, 810, 217)$	$(906, 805, 214)$	$(\mathbf{1171}, \mathbf{805}, \mathbf{213})$

Table 9. The overall security of Π_{KEM}

Parameters	Classical security	Quantum security
Π_{KEM}-512	111	100
Π_{KEM}-768	162	147
Π_{KEM}-1024	235	213

Table 10. Comparison between AMLWE and MLWE under "comparable" parameters

Parameters	$(n, k, q, \eta_1, \eta_2)$	Classical security	Quantum security	$\eta_1 \cdot \eta_2$
Π_{KEM}-512	$(256, 2, 7681, 2, 12)$	111	100	24
MLWE I	$(256, 2, 7681, 5, 5)$	112	102	25
Π_{KEM}-768	$(256, 3, 7681, 1, 4)$	162	147	4
MLWE II	$(256, 3, 7681, 2, 2)$	163	148	4
Π_{KEM}-1024	$(512, 2, 12289, 2, 8)$	235	213	16
MLWE III	$(512, 2, 12289, 4, 4)$	236	214	16

Further, in order to study the complexity relations of asymmetric (M)LWE and standard (M)LWE, we give a comparison in Table 10 between the AMLWE and the corresponding MLWE, in terms of the parameter choices used by Π_{KEM}, which shows that the hardness of AMLWE with Gaussian standard variances

α_1, α_2 is "comparable" to that of MLWE with Gaussian standard variance $\sqrt{\alpha_1\alpha_2}$. We note that the comparison only focuses on security, and the corresponding MLWE, for the parameters given in Table 10, if ever used to build a KEM, cannot achieve the same efficiency and correctness as our Π_{KEM} does.

5.2 Concrete Security of Π_{SIG}

As before, in order to obtain an overall security estimation of the Π_{SIG} under the three proposed parameter settings against key recovery attacks, we enumerate all possible values of m and b to reach a conservative estimation *sec* about the computational complexities of primal attacks and dual attacks by using a python script. Tables 11 and 12 estimate the complexities of the three parameter sets of the underlying ALWE problem against primal attacks and dual attacks by taking the minimum of *sec* over all possible values of (m, b).

Likewise, we enumerate all possible values of m and b to reach a conservative estimation *sec* about the computational complexities of ℓ_2-norm and ℓ_∞-norm attacks. Tables 13 and 14 estimate the complexities of the three parameter sets of the underlying ASIS problem against ℓ_2-normal and ℓ_∞-normal attacks by taking the minimum of *sec* over all possible values of (m, b).

In Table 15, we give the overall security of Π_{SIG} under the three parameter settings against key recovery and forgery attacks, which takes account of both AMLWE and AMSIS attacks.

Table 11. The security of Π_{SIG} against AMLWE primal attacks (The last row of the third column has no figures, because the complexity (i.e., *sec*) of the traditional attack for Π_{SIG}-1536 is too large, and our python script fails to compute it)

Parameters	Attack model	Traditional (m, b, sec)	Variant 1 (m, b, sec)	Variant 2 (m, b, sec)
Π_{SIG}-1024	Classical	$(1021, 555, 162)$	$(671, 345, 100)$	$(\mathbf{741}, \mathbf{340}, \mathbf{99})$
	Quantum	$(1021, 555, 147)$	$(671, 345, 91)$	$(\mathbf{741}, \mathbf{340}, \mathbf{90})$
Π_{SIG}-1280	Classical	$(1276, 1060, 310)$	$(996, 500, 146)$	$(\mathbf{896}, \mathbf{490}, \mathbf{143})$
	Quantum	$(1276, 1060, 281)$	$(996, 500, 132)$	$(\mathbf{896}, \mathbf{490}, \mathbf{129})$
Π_{SIG}-1536	Classical	-	$(1101, 660, 193)$	$(\mathbf{1106}, \mathbf{615}, \mathbf{179})$
	Quantum	-	$(1101, 660, 175)$	$(\mathbf{1106}, \mathbf{615}, \mathbf{163})$

Table 12. The security of Π_{SIG} against AMLWE dual attacks

Parameters	Attack model	Traditional (m, b, sec)	Variant 1 (m, b, sec)	Variant 2 (m, b, sec)	Variant 3 (m, b, sec)
Π_{SIG}-1024	Classical	$(1021, 550, 160)$	$(\mathbf{786}, \mathbf{340}, \mathbf{99})$	$(\mathbf{706}, \mathbf{340}, \mathbf{99})$	$(\mathbf{706}, \mathbf{340}, \mathbf{99})$
	Quantum	$(1021, 550, 145)$	$(\mathbf{786}, \mathbf{340}, \mathbf{90})$	$(\mathbf{706}, \mathbf{340}, \mathbf{90})$	$(\mathbf{706}, \mathbf{340}, \mathbf{90})$
Π_{SIG}-1280	Classical	$(1276, 1050, 307)$	$(1121, 495, 144)$	$(\mathbf{966}, \mathbf{485}, \mathbf{141})$	$(\mathbf{966}, \mathbf{485}, \mathbf{141})$
	Quantum	$(1276, 1050, 278)$	$(1121, 495, 131)$	$(\mathbf{966}, \mathbf{485}, \mathbf{128})$	$(\mathbf{966}, \mathbf{485}, \mathbf{128})$
Π_{SIG}-1536	Classical	$(1535, 1535, 464)$	$(1381, 650, 190)$	$(\mathbf{1031}, \mathbf{615}, \mathbf{179})$	$(\mathbf{1036}, \mathbf{615}, \mathbf{179})$
	Quantum	$(1235, 1535, 422)$	$(1381, 650, 172)$	$(\mathbf{1031}, \mathbf{615}, \mathbf{163})$	$(\mathbf{1036}, \mathbf{615}, \mathbf{163})$

Table 13. The security of Π_{SIG} against two-norm attack (for ASIS problem)

Parameters	Attack model	Traditional (m, b, sec)	Variation 1 (m, b, sec)
Π_{SIG}-1024	Classical	$(2031, 750, 219)$	$(\mathbf{2031}, \mathbf{665}, \mathbf{194})$
	Quantum	$(2031, 750, 198)$	$(\mathbf{2031}, \mathbf{665}, \mathbf{176})$
Π_{SIG}-1280	Classical	$(2537, 1100, 321)$	$(\mathbf{2537}, \mathbf{900}, \mathbf{263})$
	Quantum	$(2537, 1100, 291)$	$(\mathbf{2537}, \mathbf{900}, \mathbf{238})$
Π_{SIG}-1536	Classical	$(3043, 1395, 408)$	$(\mathbf{3043}, \mathbf{1140}, \mathbf{333})$
	Quantum	$(3043, 1395, 370)$	$(\mathbf{3043}, \mathbf{1140}, \mathbf{302})$

Table 14. The security of Π_{SIG} against infinity-norm attack (for ASIS problem)

Parameters	Attack model	Traditional (m, b, sec)	Variant 1 (m, b, sec)	Variant 2 (m, b, sec)
Π_{SIG}-1024	Classical	$(1831, 385, 112)$	$(1781, 385, 112)$	$(\mathbf{1731}, \mathbf{360}, \mathbf{105})$
	Quantum	$(1831, 385, 102)$	$(1781, 385, 102)$	$(\mathbf{1731}, \mathbf{360}, \mathbf{95})$
Π_{SIG}-1280	Classical	$(2387, 495, 144)$	$(2387, 545, 159)$	$(\mathbf{2187}, \mathbf{485}, \mathbf{141})$
	Quantum	$(2387, 495, 131)$	$(2387, 545, 144)$	$(\mathbf{2187}, \mathbf{485}, \mathbf{128})$
Π_{SIG}-1536	Classical	$(2743, 630, 184)$	$(2793, 690, 201)$	$(\mathbf{2543}, \mathbf{615}, \mathbf{179})$
	Quantum	$(2743, 630, 167)$	$(2793, 690, 183)$	$(\mathbf{2543}, \mathbf{615}, \mathbf{163})$

Table 15. The overall security of Π_{SIG}

Parameters	Classical security	Quantum security
Π_{SIG}-1024	99	90
Π_{SIG}-1280	141	128
Π_{SIG}-1536	179	163

Acknowledgments. We thank the anonymous reviewers for their helpful suggestions. Jiang Zhang is supported by the National Natural Science Foundation of China (Grant Nos. 61602046, 61932019), the National Key Research and Development Program of China (Grant Nos. 2017YFB0802005, 2018YFB0804105), the Young Elite Scientists Sponsorship Program by CAST (2016QNRC001), and the Opening Project of Guangdong Provincial Key Laboratory of Data Security and Privacy Protection (2017B030301004). Yu Yu is supported by the National Natural Science Foundation of China (Grant No. 61872236), the National Cryptography Development Fund (Grant No. MMJJ20170209). Shuqin Fan and Zhenfeng Zhang are supported by the National Key Research and Development Program of China (Grant No. 2017YFB0802005).

A Definitions of Hard Problems

The AMLWE Problem (with Binomial Distributions). The decisional AMLWE problem $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ asks to distinguish $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e})$ and uniform over $R_q^{k\times\ell} \times R_q^k$, where $\mathbf{A} \overset{\$}{\leftarrow} R_q^{k\times\ell}, \mathbf{s} \overset{\$}{\leftarrow} B_{\eta_1}^{\ell}, \mathbf{e} \overset{\$}{\leftarrow} B_{\eta_2}^{k}$. Obviously, when $\eta_1 = \eta_2$, the AMLWE problem is the standard MLWE problem.

The AMLWE-R Problem. The AMLWE-R problem $\text{AMLWE-R}_{n,q,p,k,\ell,\eta_1,\eta_2}$ asks to distinguish

$$(\mathbf{A}, \bar{\mathbf{t}} = \lceil \mathbf{t} \rfloor_{q\to p}, \mathbf{A}^T\mathbf{s} + \mathbf{e}, \lceil \bar{\mathbf{t}} \rfloor_{p\to q}^T \mathbf{s} + e)$$

from $(\mathbf{A}', \lceil \mathbf{t}' \rfloor_{q\to p}, \mathbf{u}, v) \in R_q^{\ell\times k} \times R_p^{\ell} \times R_q^{k} \times R_q$, where $\mathbf{A}, \mathbf{A}' \overset{\$}{\leftarrow} R_q^{\ell\times k}, \mathbf{s} \overset{\$}{\leftarrow} B_{\eta_1}^{\ell}, \mathbf{e} \overset{\$}{\leftarrow} \mathbf{B}_{\eta_2}^{k}, e \overset{\$}{\leftarrow} B_{\eta_2}, \mathbf{t}, \mathbf{t}' \overset{\$}{\leftarrow} R_q^{\ell}, \mathbf{u} \overset{\$}{\leftarrow} R_q^{k}, v \overset{\$}{\leftarrow} R_q$.

The AMSIS Problem. Given a uniform matrix $\mathbf{A} \in R_q^{k\times(\ell_1+\ell_2-k)}$, the (Hermite Normal Form) AMSIS problem $\text{AMSIS}_{n,q,k,\ell_1,\ell_2,\beta_1,\beta_2}^{\infty}$ over ring R_q asks to find a non-zero vector $\mathbf{x} \in R_q^{\ell_1+\ell_2}\backslash\{\mathbf{0}\}$ such that $(\mathbf{I}_k\|\mathbf{A})\mathbf{x} = \mathbf{0} \bmod q$, $\|\mathbf{x}_1\|_\infty \le \beta_1$ and $\|\mathbf{x}_2\|_\infty \le \beta_2$, where $\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \in R_q^{\ell_1+\ell_2}, \mathbf{x}_1 \in R_q^{\ell_1}, \mathbf{x}_2 \in R_q^{\ell_2}$.

The AMSIS-R Problem. Given a uniformly random matrix $\mathbf{A} \in R_q^{k\times(\ell_1+\ell_2-k)}$ and a uniformly random vector $\mathbf{t} \in R_q^k$, the (Hermite Normal Form) AMSIS-R problem $\text{AMSIS-R}_{n,q,d,k,\ell_1,\ell_2,\beta_1,\beta_2}^{\infty}$ over ring R_q asks to find a non-zero vector $\mathbf{x} \in R_q^{\ell_1+\ell_2+1}\backslash\{\mathbf{0}\}$ such that $\left(\mathbf{I}_k\|\mathbf{A}\|\mathbf{t}_1 \cdot 2^d\right)\mathbf{x} = \mathbf{0} \bmod q$, $\|\mathbf{x}_1\|_\infty \le \beta_1$, $\|\mathbf{x}_2\|_\infty \le \beta_2$ and $\|x_3\|_\infty \le 2$, where $\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ x_3 \end{pmatrix} \in R_q^{\ell_1+\ell_2+1}, \mathbf{x}_1 \in R_q^{\ell_1}, \mathbf{x}_2 \in R_q^{\ell_2}, x_3 \in R_q$ and $(\mathbf{t}_1, \mathbf{t}_0) = \mathsf{Power2Round}_q(\mathbf{t}, d)$.

The SelfTargetAMSIS Problem. Let $\mathsf{H} : \{0,1\}^* \to B_{60}$ is a (quantum) random oracle. Given a uniformly random matrix $\mathbf{A} \in R_q^{k\times(\ell_1+\ell_2-k)}$ and a uniform vector $\mathbf{t} \in R_q^k$, the SelfTargetAMSIS problem $\text{SelfTargetAMSIS}_{n,q,k,\ell_1,\ell_2,\beta_1,\beta_2}^{\infty}$ over ring R_q asks to find a vector $\mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ c \end{pmatrix}$ and $\mu \in \{0,1\}^*$, such that $\|\mathbf{y}_1\|_\infty \le \beta_1$, $\|\mathbf{y}_2\|_\infty \le \beta_2$, $\|c\|_\infty \le 1$ and $\mathsf{H}\left(\mu, (\mathbf{I}_k\|\mathbf{A}\|\mathbf{t})\mathbf{y}\right) = c$ holds.

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC 1996, pp. 99–108. ACM, New York, NY, USA (1996)
2. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Math. Cryptol. **9**, 169–203 (2015)

3. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange-a new hope. In: USENIX Security Symposium 2016 (2016)
4. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_35
5. Bai, S., Galbraith, S.D.: An improved compression technique for signatures based on learning with errors. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 28–47. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_2
6. Bos, J., et al.: CRYSTALS - Kyber: a CCA-secure module-lattice-based KEM. In: 2018 IEEE European Symposium on Security and Privacy (EuroS P), pp. 353–367, April 2018
7. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. In: Innovations in Theoretical Computer Science, ITCS, pp. 309–325 (2012)
8. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS), pp. 97–106, October 2011
9. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing, STOC 2013, pp. 575–584. ACM, New York, NY, USA (2013)
10. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_29
11. Cheon, J.H., Kim, D., Lee, J., Song, Y.: Lizard: cut off the tail! A practical post-quantum public-key encryption from LWE and LWR. In: Catalano, D., De Prisco, R. (eds.) SCN 2018. LNCS, vol. 11035, pp. 160–177. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_9
12. Ducas, L., et al.: Crystals-dilithium: a lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 238–268 (2018)
13. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
14. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptol. **26**(1), 80–101 (2013)
15. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 197–206. ACM, New York, NY, USA (2008)
16. Goldwasser, S., Kalai, Y., Peikert, C., Vaikuntanathan, V.: Robustness of the learning with errors assumption. In: Proceedings of the Innovations in Computer Science 2010. Tsinghua University Press (2010)
17. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: STOC 1996, pp. 212–219. ACM (1996)
18. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 341–371. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12

19. IBM: IBM unveils world's first integrated quantum computing system for commercial use (2019). https://newsroom.ibm.com/2019-01-08-IBM-Unveils-Worlds-First-Integrated-Quantum-Computing-System-for-Commercial-Use
20. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10993, pp. 96–125. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_4
21. Knuth, D.: The Art of Computer Programming, vol. 2, 3rd edn. Addison-Wesley, Boston (1997)
22. Lindner, R., Peikert, C.: Better key sizes (and Attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_21
23. Lyubashevsky, V.: Fiat-Shamir with aborts: applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10366-7_35
24. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43
25. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
26. Micciancio, D.: On the hardness of learning with errors with binary secrets. Theory Comput. **14**(13), 1–17 (2018)
27. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science 2004, pp. 372–381 (2004)
28. NSA National Security Agency. Cryptography today, August 2015. https://www.nsa.gov/ia/programs/suiteb_cryptography/
29. National Institute of Standards and Technology. SHA-3 standard: permutation-based hash and extendable-output functions. FIPS PUB 202 (2015). http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf
30. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, STOC 2005, pp. 84–93. ACM, New York, NY, USA (2005)
31. Shor, P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
32. Zhang, J., Yu, Y., Fan, S., Zhang, Z., Yang, K.: Tweaking the asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes. Cryptology ePrint Archive, Report 2019/510 (2019)