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
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[3] Trusted Computing and Information Assurance Laboratory,
Institute of Software, Chinese Academy of Sciences, China
jiangzhang09@gmail.com,yuyu@yuyu.hk, shuqinfan78@163.com,
zfzhang@tca.iscas.ac.cn,yangk@sklc.org

Abstract. 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 security, the public key (resp., ciphertext) of our KEM only has 896 bytes (resp., 992 bytes), which gives an improvement of 192 bytes (resp.,160 bytes) over Kyber.

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 obtain better compromise between computational costs, storage overheads and security and therefore construct an SUF-CMA secure signature scheme with shorter public keys and signatures. For a 128-bit security, the public key (resp., signature) of our signature scheme only has 1312 bytes (resp., 2445 bytes), which gives an improvement of 160 bytes (resp, 256 bytes) over Dilithium.

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 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. Furthermore, implementations of our proposed schemes appear to be (slightly) more computationally efficient than their counterparts.

1 Introduction

Despite the tremendous success of traditional public-key cryptography (a.k.a. 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 [41] once large-scale quantum computers become a reality. With the rapid advancement of quantum computing technology in recent years [25], 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. In response to the quantum

crisis, several government agencies and standardization organizations have announced plans to solicit and standardize PQC algorithms. In 2015, the NSA [37] 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 key-establishment algorithms (e.g., Kyber [12]), and 9 candidate signature schemes (e.g., Dilithium [19]) 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, 35] and the Learning With Errors (LWE) problem [40]. 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 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, such that $\mathbf{Ax} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq \beta$. Correspondingly, the search LWE problem $\mathrm{LWE}_{n,m,q,\alpha}$ searches for $\mathbf{s} \in \mathbb{Z}_q^n$ from samples $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \xleftarrow{\$} \chi_\alpha^m$. Decisional LWE problem asks to distinguish $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$ from uniform 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 [40, 33].

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 [40, 35]. 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 [6]. Furthermore, the use of a "small" secret in the normal form of LWE comes in handy in certain application scenarios, e.g., for better managing the growth of the noise in homomorphic encryptions [16, 13].

SIS is usually used in constructing signature schemes, and LWE is better suited for public-key encryption 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 security typically cannot provide a matching decryption error probability ν (say $\nu = 2^{-128}$) for the resulting LWE-based PKE scheme. Note that a larger ν (i.e., $\nu > 2^{-128}$) will 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 $\mathbf{Ax} = \mathbf{0} \bmod q$, $\|\mathbf{x}_1\|_\infty \leq \beta_1$ 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 $\mathrm{SIS}^{\infty}_{n,m_1+m_2,q,\max(\beta_1,\beta_2)}$. In particular, 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 parameter values 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 [4]

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

More importantly, the literature [22, 15, 34] 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. [18] 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.

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 [19, 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 construct a key encapsulation mechanism (KEM) and a signature scheme of smaller sizes.

Technically, our KEM scheme is mainly based on the PKE schemes in [12, 30], except that we make several modifications to utilize the inherent asymmetry of the (M)LWE secret and noise in contributing to the decryption error probabilities, which allow us to obtain smaller public keys and ciphertexts. In Section 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), which gives an improvement of 192 bytes (resp.,160 bytes) over Kyber [12].

Our signature scheme bears most resemblance to Dilithium in [19]. 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

[4] In fact, the reductions can be established for certain χ (e.g., discrete Gaussian).

overheads and security, which yields smaller public keys and signatures without sacrificing the security or computational efficiency. In Section 4.1, we will further discuss the asymmetries in existing constructions, and illustrate our design rationale in more details. For a targeted 128-bit security, the public key (resp., signature) of our signature only has 1312 bytes (resp., 2445 bytes), which gives an improvement of 160 bytes (resp, 256 bytes) over Dilithium [19].

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 will compare our KEM with Kyber [12] in Section 1.1, and compare our signature with Dilithium [19] in Section 1.2.

1.1 Comparison with Kyber

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

NIST Category	Schemes	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)	$\|pk\|$ (Bytes)	$\|sk\|$ (Bytes)	$\|C\|$ (Bytes)	$\|ss\|$ (Bytes)	Dec. Failure	Quantum Sec.
I	Kyber	70 665	86 235	73 722	736	1632	800	32	2^{-145}	100
	Our Π_{KEM}	71 352	76 583	64 634	672	1568	672	32	2^{-82}	102
III	Kyber	105 516	131 607	116 223	1088	2400	1152	32	2^{-142}	161
	Our Π_{KEM}	90 335	99 780	85 622	896	2208	992	32	2^{-128}	147
V	Kyber	138 915	176 151	158 931	1440	3168	1504	32	2^{-169}	218
	Our Π_{KEM}	146 304	174 560	140 894	1472	3392	1536	64	2^{-211}	213

In Table 1, we compare our KEM scheme Π_{KEM} with Kyber [12]. The first column provides the targeted security strength in terms of the NIST recommended 5 categories, which aim at achieving the same security levels as AES128, SHA256/SHA3-256, AES192, SHA384/SHA3-384, and AES256, respectively. The 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 4GB 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 error. The last column "Quantum Sec." gives the estimated quantum security level expressed in bits.

Note that the estimated quantum security of our KEM scheme Π_{KEM} is slightly lower than that of Kyber, but we emphasize that our parameter choices have left out sufficient security margin reserved for further development of attacks. For example, the NIST Category III requires a quantum security at most 128 bits (this is why we set the parameter to reach a probability 2^{-128} for decryption error), while our parameter is set to achieve

an estimated quantum security of 147 bits. We also note that our KEM scheme Π_{KEM} at Category V has slightly larger sizes than Kyber. This is because we choose to set the parameters to encapsulate a key of length 64 bytes, which is twice the size of that achieved by Kyber (we note that this is achieved by only slightly increasing the sizse of the public key and ciphertexts, i.e., 32 bytes). This decision is based on the fact that a 32 bytes session key cannot provide more than 128 bits quantum security by the Grover algorithm [23], and we hope the parameter setting at Category V can achieve much higher quantum security bits, e.g., 192 bits. In all, our KEM has better overall performance (in particular, having shorter public keys and ciphertexts) than Kyber.

Very recently, the authors of Kyber [12] had made several modifications in their revised submission to the 2nd round NIST PQC standardization process, which chooses new parameters sets and removes the compression of the public key. We have not tested their new algorithm on our computer, but note that all their updates are not aiming at reducing the size of public key or ciphertext (actually, their new parameters are chosen to maintain the total sizes of the public key and ciphertexts), and the sizes of our KEM at all categories are still smaller than that of their new scheme. Besides, it seems that our techniques can also be used to improve the new Kyber scheme.

1.2 Comparison with Dilithium

Table 2. Comparison between Our Signature Scheme Π_{SIG} and Dilithium

NIST Category	Schemes	KeyGen (AVX2)	Sign (AVX2)	Verify (AVX2)	$\|pk\|$ (Bytes)	$\|sk\|$ (Bytes)	$\|\sigma\|$ (Bytes)	Quantum Sec.
I	Dilithium	151 624	661 854	142 930	1184	2800	2044	91
	Our Π_{SIG}	137 705	453 504	124 819	1056	2448	1852	90
II	Dilithium	222 650	927 313	198 865	1472	3504	2701	125
	Our Π_{SIG}	224 199	696 280	195 537	1312	3376	2445	128
III	Dilithium	273 155	907 609	275 418	1760	3856	3366	158
	Our Π_{SIG}	329 762	902 933	296 631	1568	3888	3046	163

We compare our signature scheme Π_{SIG} with Dilithium [19] in Table 2. Similarly, the running times of the KeyGen, Sign and Verify algorithms are measured in the average number of CPU cycles (over 5000 times) running on a 64-bit Ubuntu 14.4 LTS ThinkCenter desktop (equipped with Intel Core-i7 4790 3.6 GHz CPU and 4GB memory). The sizes of public key $|pk|$, secret key $|sk|$, signature $|\sigma|$ are counted in bytes.

Note that the estimated quantum security of our signature Π_{SIG} at Category I is slightly lower than that of Dilithium, but those at Categories II and III are slightly higher. We also note that the signing time of our signature is (slightly) shorter than that of Dilithium at all Categories, and all other running times are also comparable to each other. In all, our signature has better overall performance (in particular, having shorter public key and signatures) than Dilithium.

We note that the authors of Dilithium [19] had made various optimizations in the implementation of their revised submission to the 2nd round NIST PQC standardization process. We did not test their new implementation on our computers, but it seems that their optimizing techniques can also be used in our implementations.

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 will 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) to 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 $x \xleftarrow{\$} D$ sampling x according to a distribution D and by $x \xleftarrow{\$} S$ 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 $f : \mathbb{N} \to [0, 1]$ is *negligible*, if for every positive c and all sufficiently large κ it holds that $f(\kappa) < 1/\kappa^c$. We denote by $\mathsf{negl} : \mathbb{N} \to [0, 1]$ 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 $w = w_0 + w_1 X + \cdots + w_{n-1} X^{n-1} \in R$ 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 $\lceil \cdot \rfloor_{q \to p}$ as:

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

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

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

When $\lceil \cdot \rfloor_{q \to p}$ 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:

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

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 [19] 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. [19] 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 [19] for the illustration of the algorithms.

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

$r := r \bmod^{+} q$;
$r_0 := r \bmod^{\pm} 2^d$;
$r_1 := (r - r_0)/2^d$;
return (r_1, r_0);

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 [19] 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 $\|\mathbf{z}\|_\infty \leq \alpha/2$. 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:*

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

$r := r \bmod^+ q$;
$r_0 := r \bmod^\pm \alpha$;
if $r - r_0 = q - 1$ **then**
    $r_1 := 0$;
    $r_0 := r_0 - 1$;
else
    $r_1 := (r - r_0)/\alpha$;
end
return (r_1, r_0);

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

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

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

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

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

$r_1 := \mathsf{HighBits}_q(r, \alpha)$;
$v_1 := \mathsf{HighBits}_q(r + z, \alpha)$;
if $r_1 \neq v_1$ **then**
    $h := 1$;
else
    $h := 0$;
end
return h;

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

$k := (q - 1)/\alpha$;
$(r_1, r_0) := \mathsf{Decompose}_q(r, \alpha)$;
if $h = 1$ *and* $r_0 > 0$ **then**
    $r_1 := (r_1 + 1) \bmod^+ k$;
end
if $h = 1$ *and* $r_0 \leq 0$ **then**
    $r_1 := (r_1 - 1) \bmod^+ k$;
end
return r_1;

– $\mathsf{UseHint}_q(\mathsf{MakeHint}_q(\mathbf{z}, \mathbf{r}, \alpha), \mathbf{r}, \alpha) = \mathsf{HighBits}_q(\mathbf{r} + \mathbf{z}, \alpha)$.
– *Let* $\mathbf{v}_1 = \mathsf{UseHint}_q(\mathbf{h}, \mathbf{r}, \alpha)$. *Then* $\|\mathbf{r} - \mathbf{v}_1 \cdot \alpha\|_\infty \leq \alpha + 1$. *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* $\|\mathbf{s}\|_\infty \leq \beta$ *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 [12, 30]. The main difference is that our scheme uses a (slightly) different hard problem, which gives us a flexible way to set the parameters for both performance and security.

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 PKE schemes mainly follow the framework in [30] 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 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$, $\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi_\alpha^n$ 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 $\mathbf{r}, \mathbf{x}_1 \xleftarrow{\$} \chi_\alpha^n, x_2 \xleftarrow{\$} \chi_\alpha$ 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:

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

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 small 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 [16, 14] 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 $C = (\bar{\mathbf{c}}_1 = \lceil \mathbf{c}_1 \rfloor_{q\to p_2}, \bar{c}_2 = \lceil c_2 \rfloor_{q\to p_3})$.
- **Decryption**: given the secret key $sk = \mathbf{s}$ and a ciphertext $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$, compute $z = \lceil \bar{c}_2 \rfloor_{p_3\to q} - \mathbf{s}^T \lceil \bar{\mathbf{c}}_1 \rfloor_{p_2\to q}$ and output $\lceil z \rfloor_{q\to 2} = \lceil z \cdot \frac{2}{q} \rfloor \bmod 2$ as the decryption result.

Let $\bar{\mathbf{e}} = \lceil \lceil \mathbf{b} \rfloor_{q\to p_1} \rfloor_{p_1\to q} - \mathbf{b}$, $\bar{\mathbf{x}}_1 = \lceil \lceil \mathbf{c}_1 \rfloor_{q\to p_2} \rfloor_{p_2\to q} - \mathbf{c}_1$, and $\bar{x}_2 = \lceil \lceil c_2 \rfloor_{q\to p_3} \rfloor_{p_3\to q} - c_2$. 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 $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ that encrypts $\mu \in \{0, 1\}$ we have

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

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 $\bar{\mathbf{e}}, \bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2$ 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 $\bar{c}_2 = \lceil c_2 \rfloor_{q\to p_3})$.
- **Decryption**: Given the secret key $sk = \mathbf{s}$ and the ciphertext $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$, compute $z = \lceil \bar{c}_2 \rfloor_{p_3\to q} - \mathbf{s}^T \lceil \bar{\mathbf{c}}_1 \rfloor_{p_2\to q}$ and output $\lceil z \rfloor_{q\to 2} = \lceil z \cdot \frac{2}{q} \rfloor \bmod 2$ as the decryption result.

Similarly, for ciphertext $C = (\bar{\mathbf{c}}_1, \bar{c}_2)$ 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:

$$\mathrm{ALWE}_{n,m,q,\alpha_1,\alpha_2} \approx \mathrm{LWE}_{n,m,q,\sqrt{\alpha_1\alpha_2}} \ ,$$

where 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

In this section, we give the formal description of a CCA-secure KEM from AMLWE. For ease of implementation, we will use centered binomial distributions instead of Gaussian distributions as in [5, 12]. We first give an intermediate IND-CPA secure PKE, which is then transformed into an IND-CCA secure KEM by applying a slightly tweaked Fujisaki-Okamoto (FO) transformation [24, 21].

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 $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\Pi_{\mathrm{PKE}}.\mathsf{KeyGen}(\kappa)$: randomly choose $\rho \xleftarrow{\$} \{0,1\}^n$, $\mathbf{s} \xleftarrow{\$} B_{\eta_1}^k, \mathbf{e} \xleftarrow{\$} B_{\eta_2}^k$, compute $\mathbf{A} = \mathcal{H}(\rho) \in R_q^{k\times k}, \mathbf{t} = \mathbf{As} + \mathbf{e} \in R_q^k$ 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 $sk = \mathbf{s}$.
- $\Pi_{\mathrm{PKE}}.\mathsf{Enc}(pk, \mu)$: given the public key $pk = (\rho, \bar{\mathbf{t}})$ and a plaintext $\mu \in R_2$, randomly choose $\mathbf{r} \xleftarrow{\$} B_{\eta_1}^k, \mathbf{e}_1 \xleftarrow{\$} B_{\eta_2}^k, e_2 \xleftarrow{\$} B_{\eta_2}$, 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

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

- $\Pi_{\mathrm{PKE}}.\mathsf{Dec}(sk, C)$: 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

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

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 Section 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 [24, 21], we can transform the above IND-CPA secure PKE Π_{PKE} into an IND-CCA secure KEM (with implicit rejection) $\Pi_{\mathrm{KEM}} = (\mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ as follows.

- $\Pi_{\mathrm{KEM}}.\mathsf{KeyGen}(\kappa)$: randomly choose $z \xleftarrow{\$} \{0,1\}^n$, compute $(pk', sk') = \Pi_{\mathrm{PKE}}.\mathsf{KeyGen}(\kappa)$. Then, return the public key $pk = pk'$ and the secret key $sk = (pk', sk', z)$.
- $\Pi_{\mathrm{KEM}}.\mathsf{Encap}(pk)$: given the public key pk, randomly choose $\mu \xleftarrow{\$} \{0,1\}^n$, compute $\mu' = \mathsf{H}(\mu), (\bar{K}, r) = \mathsf{G}(\mu' \| \mathsf{H}(pk))$ $C = \Pi_{\mathrm{PKE}}.\mathsf{Enc}(pk, \mu'; r)$ and $K = \mathsf{H}(\bar{K} \| \mathsf{H}(C))$, where the notation $\Pi_{\mathrm{PKE}}.\mathsf{Enc}(pk, \mu'; r)$ denotes running the algorithm $\Pi_{\mathrm{PKE}}.\mathsf{Enc}(pk, \mu')$ with fixed randomness r. Finally, return the ciphertext C and the encapsulated key K.
- $\Pi_{\mathrm{KEM}}.\mathsf{Decap}(sk, C)$: given the secret key $sk = (pk', sk', z)$ and a ciphertext C, compute $\mu' = \Pi_{\mathrm{KEM}}.\mathsf{Dec}(sk', C)$ and $(\bar{K}', r') = \mathsf{G}(\mu' \| \mathsf{H}(pk')), C' = \Pi_{\mathrm{KEM}}.\mathsf{Enc}(pk, \mu'; r')$. 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 appendix A, 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), our scheme Π_{KEM} is provably IND-CCA secure. Formally, we have the following theorem.

Theorem 1. *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 checks fails (i.e., implicit rejection). Furthermore, the paper [26] 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 [26] and Theorem 5, we have the following theorem.

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

3.4 Choices of Parameters

Table 3. 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. Failure	Quantum Sec.
PARAMS I	$(256, 2, 7681)$	$(2, 12)$	$(10, 9, 3)$	672	1568	672	32	2^{-82}	100
PARAMS II	$(256, 3, 7681)$	$(1, 4)$	$(9, 9, 4)$	896	2208	992	32	2^{-128}	147
PARAMS III	$(512, 2, 12289)$	$(2, 8)$	$(11, 10, 4)$	1472	3392	1536	64	2^{-211}	213

In Table 3, we give three sets of parameters (namely, PARAMS I, PARAMS II and PARAMS III) for Π_{KEM}, aiming at providing quantum security of at least 80, 128 and 192 bits, respectively. Those parameters are carefully chosen such that the decryption error probabilities (i.e., 2^{-82}, 2^{-128} and 2^{-211}, respectively) commensurate with the respective targeted security strengths. A concrete estimation of the security strength provided by the parameter sets will be given in Section 5. Among them, PARAMS II is the recommended parameter set. Since the existence of quantum searching algorithms [23], 2κ-bit session key can only provide at most κ security. In order to achieve security larger than 128, the parameter set PARAMS III is chosen to support an encryption of 64-bytes (512-bit) session key. Note that PARAMS I and PARAMS II only support an encryption of 32-byte (256-bit) session key.

We implemented the scheme Π_{KEM} on a 64-bit Windows 10 Thinkpad X1 notebook (equipped with Intel Core-i7 6500U 2.5 GHz CPU and 8GB Memory) and a 64-bit Ubuntu 14.4 LTS ThinkCenter desktop (equipped with Intel Core-i7 4790 3.6 GHz CPU and 4GB memory), respectively. Particularly, the codes are written in the standard C language for the reference implementation. We also give an optimized implemention which uses AVX2 instructions to speedup some basic operations such number theory transform and vector multiplication. Table 4 and table 5 shows the average CPU cycles of running the corresponding algorithms over 10000 times on Windows 10 and Ubuntu, respectively.

Table 4. Performances of Π_{KEM} on Windows 10 (in CPU Cycles)

Parameter	KeyGen	Encap	Decap	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)
PARAMS I	137 751	196 576	213 957	54 838	73 018	65 752
PARAMS II	205 866	268 938	291 920	76 778	102 015	92 686
PARAMS III	308 050	440 070	460 837	128 651	185 796	153 445

Table 5. Performances of Π_{KEM} on Ubuntu 14.4 LTS (in CPU Cycles)

Parameter	KeyGen	Encap	Decap	KeyGen (AVX2)	Encap (AVX2)	Decap (AVX2)
PARAMS I	159 764	216 617	244 179	71 352	76 583	64 634
PARAMS II	232 102	295 222	331 524	90 335	99 780	85 622
PARAMS III	335 478	460 563	506 319	146 304	174 560	140 894

4 An Improved Signature Scheme from AMLWE and AMSIS

Our signature scheme is based on the "Fiat-Shamir with Aborts" technique [31], and bears most resemblance to the signature scheme in [19]. 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 [20] 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 $pk = (\mathbf{A}, \mathbf{t}) \in R_q^{k\times\ell} \times R_q^k$ 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 $\mathbf{A}\mathbf{z} = \mathbf{w} + c\mathbf{t}$.

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 [31, 32] 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 $\mathbf{A} \xleftarrow{\$} R_q^{k\times\ell}, \mathbf{x} \xleftarrow{\$} S_\eta^{\ell}$, and compute $\mathbf{t} = \mathbf{A}\mathbf{x}$. 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 $\mathbf{y} \xleftarrow{\$} S_{\gamma-1}^{\ell}$;
2. compute $\mathbf{w} = \mathbf{A}\mathbf{y}$ 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 (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 $c = \mathsf{H}(\mathbf{A}\mathbf{z} - c\mathbf{t}\|\mu)$, otherwise return 0.

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 $\mathbf{y} \in S_{\gamma}^{\ell}$ 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 (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 [8, 19] 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 (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 $\mathbf{w} = \mathbf{A}\mathbf{y} = \mathbf{A}\mathbf{y} - c\mathbf{s}_2 + c\mathbf{s}_2$, $\mathsf{LowBits}(\mathbf{A}\mathbf{y} - c\mathbf{s}_2, 2\gamma_2) < \gamma_2 - \beta$ 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 $\mathsf{H}_1 : \{0,1\}^{256} \to R_q^{k\times\ell}, \mathsf{H}_2 : \{0,1\}^* \to \{0,1\}^{384}$, $\mathsf{H}_3 : \{0,1\}^* \to S^\ell_{\gamma_1 - 1}$ and $\mathsf{H}_4 : \{0,1\}^* \to B_{60}$ 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 $\rho, K \xleftarrow{\$} \{0,1\}^{256}$, $\mathbf{s}_1 \xleftarrow{\$} S_{\eta_1}^{\ell}, \mathbf{s}_2 \xleftarrow{\$} S_{\eta_2}^{k}$. 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 $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$.
- $\Pi_{\mathrm{SIG}}.\mathsf{Sign}(sk, M)$: given the secret key $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$ 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 $\sigma = (\mathbf{z}, \mathbf{h}, c)$.
- $\Pi_{\mathrm{SIG}}.\mathsf{Verify}(pk, M, \sigma)$: 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 $\mathbf{A} = \mathsf{H}_1(\rho) \in R_q^{k\times\ell}, \mu = \mathsf{H}_2(\mathsf{H}_2(pk) \| M) \in \{0,1\}^{384}$. Then, compute $\mathbf{u} = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, \mathbf{w}_1' = \mathsf{UseHints}_q(\mathbf{h}, \mathbf{u}, 2\gamma_2)$ 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 $\mathbf{y} \xleftarrow{\$} S_{\gamma_1 - 1}^{\ell}$ at random. We also note that the hash function H_4 can be constructed by using an extendable output function such as SHAKE-256 [38] and a so-called "inside-out" version of Fisher-Yates shuffle algorithm [29]. The detailed constructions of hash functions H_3 and H_4 can be found in [19].

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

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

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 [31, 32] to generate $(\mathbf{z}, \mathbf{h})$, the efficiency of the signing algorithm is determined by 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 appendix B, we will 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), our scheme Π_{SIG} is provably SUF-CMA secure. Formally, we have the following theorem.

Theorem 3. *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* $\mathsf{H}_3 : \{0,1\}^* \to S^\ell_{\gamma_1-1}$ *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 analogue of the SelfTargetMSIS problem introduced by Ducas et al. [19]), we can also prove that our scheme Π_{SIG} is provably SUF-CMA secure. Formally, we have that following theorem.

Theorem 4. *In the quantum random oracle model, the 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* $\mathrm{SelfTargetAMSIS}^\infty_{\mathsf{H}_4,n,q,k,\ell_1,\ell_2,4\gamma_2,(\gamma_1-\beta_1)}$.

Table 6. Parameters for Π_{SIG} (the sizes of public key pk, secret key sk and signature σ are counted in Bytes. The column "Exp. Reps" indicates the excepted number of repetitions that the signing algorithm takes to output a valid signature)

Parameters	$(n, k, \ell, q, d, \omega)$	(η_1, η_2)	(β_1, β_2)	(γ_1, γ_2)	$\lvert pk\rvert$	$\lvert sk\rvert$	$\lvert\sigma\rvert$	Exp. Reps	Quantum Sec.
PARAMS I	$(256, 4, 3, 2021377, 13, 80)$	$(2, 3)$	$(120, 175)$	$(131072, 168448)$	1056	2448	1852	5.86	90
PARAMS II	$(256, 5, 4, 3870721, 14, 96)$	$(2, 5)$	$(120, 275)$	$(131072, 322560)$	1312	3376	2445	7.61	128
PARAMS III	$(256, 6, 5, 3870721, 14, 120)$	$(1, 5)$	$(60, 275)$	$(131072, 322560)$	1568	3888	3046	6.67	163

4.4 Choices of Parameters

In Table 6, we provide three sets of parameters (i.e., PARAMS I, PARAMS II and PARAMS III) 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 Section 5. Among them, PARAMS II is the recommended parameter set.

We implemented the scheme Π_{SIG} under the same configurations as in Section 3.4. The codes are written in the standard C language for the reference implementation. We also give an optimized implementation by using the AVX2 instructions to speedup some basic operations such as number theory transform and vector multiplication. Table 7 and Table 8 give the average CPU cycles of running the corresponding algorithms over 5000 times on Windows 10 and Ubuntu operating systems respectively.

Table 7. The Performance of Signature Scheme Π_{SIG} on Windows 10 (in CPU Cycles)

Parameters	KeyGen	Sign	Verify	KeyGen (AVX2)	Sign (AVX2)	Verify (AVX2)
PARAMS I	309 850	1 303 320	315 179	142 510	473 735	136 506
PARAMS II	489 230	2 079 377	479 762	238 702	729 374	211 466
PARAMS III	668 976	2 449 580	656 770	348 429	957 242	318 072

Table 8. The Performance of Signature Scheme Π_{SIG} on Ubuntu (in CPU Cycles)

Parameters	KeyGen	Sign	Verify	KeyGen (AVX2)	Sign (AVX2)	Verify (AVX2)
PARAMS I	306 742	1 559 697	316 905	139 098	456 173	126 537
PARAMS II	469 611	2 507 741	470 712	227 770	705 465	197 961
PARAMS III	635 949	2 941 045	639 161	340 136	916 605	307 146

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 [4]. 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 [19, 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 $\mathrm{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ 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 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \leftarrow \chi_{\alpha_1}^n$ and $\mathbf{e} \leftarrow \chi_{\alpha_2}^m$.

In Appendix C, 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 [19] 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 $\mathrm{ASIS}_{n,q,m_1,m_2,\beta_1,\beta_2}^{\infty}$. As before, when analyzing RSIS or MSIS based cryptosystems, one often translates RSIS/MSIS instances to the corresponding SIS counterparts [19] and then applies the attacks.

In appendix D, 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 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 complexities of primal attacks and dual attacks, by using a python script. Table 9 and Table 10 estimate the complexities of the three parameter sets against primal attacks and dual attacks by taking the minimum of 2^{sec} over all possible values of (m, b). Taking into account the above, Table 11 shows the overall security of Π_{KEM}.

Table 9. 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)
PARAMS I	Classical	$(761, 390, 114)$	$(531, 405, 118)$	$(\mathbf{476}, \mathbf{385}, \mathbf{112})$
	Quantum	$(761, 390, 103)$	$(531, 405, 107)$	$(\mathbf{476}, \mathbf{385}, \mathbf{102})$
PARAMS II	Classical	$(1021, 640, 187)$	$(646, 575, 168)$	$(\mathbf{556}, \mathbf{560}, \mathbf{163})$
	Quantum	$(1021, 640, 169)$	$(646, 575, 152)$	$(\mathbf{556}, \mathbf{560}, \mathbf{148})$
PARAMS III	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 10. 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)
PARAMS I	Classical	$(766, 385, 112)$	$(736, 395, 115)$	$(595, 380, 111)$	$(\mathbf{711}, \mathbf{380}, \mathbf{111})$
	Quantum	$(766, 385, 102)$	$(736, 395, 104)$	$(\mathbf{596}, \mathbf{380}, \mathbf{100})$	$(\mathbf{711}, \mathbf{380}, \mathbf{100})$
PARAMS II	Classical	$(1021, 620, 181)$	$(881, 570, 166)$	$(\mathbf{586}, \mathbf{555}, \mathbf{162})$	$(\mathbf{776}, \mathbf{555}, \mathbf{162})$
	Quantum	$(1021, 620, 164)$	$(881, 570, 151)$	$(\mathbf{586}, \mathbf{555}, \mathbf{147})$	$(\mathbf{776}, \mathbf{555}, \mathbf{147})$
PARAMS III	Classical	$(1531, 810, 237)$	$(981, 810, 239)$	$(906, 805, 236)$	$(\mathbf{1171}, \mathbf{805}, \mathbf{235})$
	Quantum	$(1531, 810, 215)$	$(981, 810, 217)$	$(906, 805, 214)$	$(\mathbf{1171}, \mathbf{805}, \mathbf{213})$

Further, in order to study the complexity relations of asymmetric (M)LWE and standard (M)LWE, we give a comparison in Table 12 between the AMLWE and the corresponding MLWE, in terms of the parameter choices used by Π_{KEM}. Concrete evaluation results show that the hardness of $\mathrm{AMLWE}_{n,q,m,\alpha_1,\alpha_2}$ and $\mathrm{MLWE}_{n,q,m,\sqrt{\alpha_1\alpha_2}}$ is approximately equivalent in the following sense:

$$\mathrm{AMLWE}_{n,q,m,\alpha_1,\alpha_2} \approx \mathrm{MLWE}_{n,q,m,\sqrt{\alpha_1\alpha_2}}.$$

Recall that the above equivalence is only for security. The corresponding MLWE, for the parameters given in Table 12, 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 (the number of ALWE samples) and b to reach a conservative estimation about the computational complexities of primal attacks and dual attacks by using a python script. Table 13 and Table 14 estimate the complexities of the three parameter sets of the underlying ALWE problem against primal attacks and dual attacks by taking the minimum of 2^{sec} over all possible values of (m, b).

Likewise, we enumerate all possible values of m (the number of samples) and b to reach a conservative estimation about the computational complexities of ℓ_2-norm and ℓ_∞-norm attacks. Table 15 and Table 16 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 2^{sec} over all possible values of (m, b).

In Table 17, 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 overall security of Π_{KEM}

Parameters	Classical Security	Quantum Security
PARAMS I	111	100
PARAMS II	162	147
PARAMS III	235	213

Table 12. Comparison between AMLWE and MLWE under "equivalence" parameters

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

Table 13. 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 PARAMS III 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)
PARAMS I	Classical	$(1021, 555, 162)$	$(671, 345, 100)$	$(\mathbf{741}, \mathbf{340}, \mathbf{99})$
	Quantum	$(1021, 555, 147)$	$(671, 345, 91)$	$(\mathbf{741}, \mathbf{340}, \mathbf{90})$
PARAMS II	Classical	$(1276, 1060, 310)$	$(996, 500, 146)$	$(\mathbf{896}, \mathbf{490}, \mathbf{143})$
	Quantum	$(1276, 1060, 281)$	$(996, 500, 132)$	$(\mathbf{896}, \mathbf{490}, \mathbf{129})$
PARAMS III	Classical	-	$(1101, 660, 193)$	$(\mathbf{1106}, \mathbf{615}, \mathbf{179})$
	Quantum	-	$(1101, 660, 175)$	$(\mathbf{1106}, \mathbf{615}, \mathbf{163})$

Table 14. 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)
PARAMS I	Classical	$(1021, 550, 160)$	$(\mathbf{786}, \mathbf{340}, \mathbf{99})$	$(\mathbf{706}, \mathbf{340}, \mathbf{99})$	$(\mathbf{706}, \mathbf{340}, \mathbf{99})$
	Quantum	$(1021, 550, 145)$	$(\mathbf{786}, \mathbf{340}, \mathbf{90})$	$(\mathbf{706}, \mathbf{340}, \mathbf{90})$	$(\mathbf{706}, \mathbf{340}, \mathbf{90})$
PARAMS II	Classical	$(1276, 1050, 307)$	$(1121, 495, 144)$	$(\mathbf{966}, \mathbf{485}, \mathbf{141})$	$(\mathbf{966}, \mathbf{485}, \mathbf{141})$
	Quantum	$(1276, 1050, 278)$	$(1121, 495, 131)$	$(\mathbf{966}, \mathbf{485}, \mathbf{128})$	$(\mathbf{966}, \mathbf{485}, \mathbf{128})$
PARAMS III	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 15. The security of Π_{SIG} against two-norm attack (for ASIS problem)

Parameters	Attack Model	Traditional (m, b, sec)	Variation 1 (m, b, sec)
PARAMS I	Classical	$(2031, 750, 219)$	$(\mathbf{2031}, \mathbf{665}, \mathbf{194})$
	Quantum	$(2031, 750, 198)$	$(\mathbf{2031}, \mathbf{665}, \mathbf{176})$
PARAMS II	Classical	$(2537, 1100, 321)$	$(\mathbf{2537}, \mathbf{900}, \mathbf{263})$
	Quantum	$(2537, 1100, 291)$	$(\mathbf{2537}, \mathbf{900}, \mathbf{238})$
PARAMS III	Classical	$(3043, 1395, 408)$	$(\mathbf{3043}, \mathbf{1140}, \mathbf{333})$
	Quantum	$(3043, 1395, 370)$	$(\mathbf{3043}, \mathbf{1140}, \mathbf{302})$

Table 16. 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)
PARAMS I	Classical	$(1831, 385, 112)$	$(1781, 385, 112)$	$(\mathbf{1731}, \mathbf{360}, \mathbf{105})$
	Quantum	$(1831, 385, 102)$	$(1781, 385, 102)$	$(\mathbf{1731}, \mathbf{360}, \mathbf{95})$
PARAMS II	Classical	$(2387, 495, 144)$	$(2387, 545, 159)$	$(\mathbf{2187}, \mathbf{485}, \mathbf{141})$
	Quantum	$(2387, 495, 131)$	$(2387, 545, 144)$	$(\mathbf{2187}, \mathbf{485}, \mathbf{128})$
PARAMS III	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 17. The Overall Security of Π_{SIG}

Parameters	Classical Security	Quantum Security
PARAMS I	99	90
PARAMS II	141	128
PARAMS III	179	163

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, STOC '96, pages 99–108, New York, NY, USA, 1996. ACM.
2. Martin R. Albrecht. On dual lattice attacks against small-secret lwe and parameter choices in helib and seal. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, pages 103–129, Cham, 2017. Springer International Publishing.
3. Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the expected cost of solving usvp and applications to lwe. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 297–322, Cham, 2017. Springer International Publishing.
4. Matin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *In Journal of Mathematical Cryptology*, 9:169–203, oct 2015.
5. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange-a new hope. In *USENIX Security Symposium*, volume 2016, 2016.
6. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer Berlin Heidelberg, 2009.
7. Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary lwe. In Willy Susilo and Yi Mu, editors, *Information Security and Privacy*, pages 322–337, Cham, 2014. Springer International Publishing.
8. Shi Bai and StevenD. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *Topics in Cryptology – CT-RSA 2014*, volume 8366 of *Lecture Notes in Computer Science*, pages 28–47. Springer International Publishing, 2014.
9. Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *Proceedings of the 13th ACM conference on Computer and Communications Security*, CCS '06, pages 390–399, New York, NY, USA, 2006. ACM.
10. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM.
11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69. Springer Berlin Heidelberg, 2011.
12. J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle. Crystals - kyber: A cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 353–367, April 2018.
13. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. Fully homomorphic encryption without bootstrapping. *Innovations in Theoretical Computer Science, ITCS*, pages 309–325, 2012.
14. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 97 –106, oct. 2011.

15. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 575–584, New York, NY, USA, 2013. ACM.
16. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524. Springer Berlin Heidelberg, 2011.
17. Yuanmi Chen. Lattice reduction and concrete security of fully homomorphic encryption. *Dept. Informatique, ENS, Paris, France, PhD thesis*, 2013.
18. Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song. Lizard: Cut off the tail! a practical post-quantum public-key encryption from lwe and lwr. In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks*, pages 160–177, Cham, 2018. Springer International Publishing.
19. Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.
20. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In AndrewM. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Berlin Heidelberg, 1987.
21. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, Jan 2013.
22. Shafi Goldwasser, Yael Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In *Proceedings of the Innovations in Computer Science 2010*. Tsinghua University Press, 2010.
23. Lov K. Grover. A fast quantum mechanical algorithm for database search. In *STOC 1996*, pages 212–219. ACM, 1996.
24. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the fujisaki-okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography – TCC 2017*, volume 10677 of *LNCS*, pages 341–371. Springer International Publishing, 2017.
25. 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.
26. Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, volume 10993 of *LNCS*, pages 96–125. Springer International Publishing, 2018.
27. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.
28. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, volume 10822 of *LNCS*, pages 552–586. Springer International Publishing, 2018.
29. Donald Knuth. *The Art of Computer Programming*, volume 2. Addison-Wesley, 3 edition edition, 1997.
30. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *Topics in Cryptology – CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer Berlin Heidelberg, 2011.
31. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer Berlin / Heidelberg, 2009.
32. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755. Springer Berlin Heidelberg, 2012.
33. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer Berlin / Heidelberg, 2010.
34. Daniele Micciancio. On the hardness of learning with errors with binary secrets. *Theory of Computing*, 14(13):1–17, 2018.
35. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 372 – 381, 2004.
36. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In Daniel J. Bernstein, Johannes A. Buchmann, and Erik Dahmen, editors, *Post-Quantum Cryptography*, pages 147–191. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

37. NSA. National secuity agency. cryptography today, August 2015. https://www.nsa.gov/ia/programs/suiteb_cryptography/.
38. 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.
39. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
40. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.
41. P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

A Security Proof for $\Pi_{\mathbf{KEM}}$

AMLWE Problem (with Binomial Distributions). Let $k, \ell \geq 1, \eta_1, \eta_2$ be positive integers. The decisional AMLWE problem
$\mathrm{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ asks to distinguish $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e})$ and uniform over $R_q^{k\times\ell} \times R_q^k$, where $\mathbf{A} \xleftarrow{\$} R_q^{k\times\ell}, \mathbf{s} \xleftarrow{\$} B_{\eta_1}^{\ell}, \mathbf{e} \xleftarrow{\$} B_{\eta_2}^{k}$. Obviously, when $\eta_1 = \eta_2$, the AMLWE problem is the standard MLWE problem.

In order to compress the public key, we also require a variant AMLWE-R (i.e., AMLWE with rounding) of the AMLWE problem for the security proof. Specifically, the (decisional) 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 $\mathbf{A}, \mathbf{A}' \xleftarrow{\$} R_q^{\ell\times k}, \mathbf{s} \xleftarrow{\$} B_{\eta_1}^{\ell}, \mathbf{e} \xleftarrow{\$} \mathbf{B}_{\eta_2}^{k}, e \xleftarrow{\$} B_{\eta_2}, \mathbf{t}, \mathbf{t}' \xleftarrow{\$} R_q^{\ell}, \mathbf{u} \xleftarrow{\$} R_q^k, v \xleftarrow{\$} R_q$.

Definition 1 (AMLWE Assumption). *For appropriate choice of parameters $n, q, k, \ell, \eta_1, \eta_2 \in \mathbb{Z}$, there is no quantum polynomial time adversary solves the* $\mathrm{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ *problem.*

Definition 2 (AMLWE-R Assumption). *For appropriate choice of parameters $n, q, p, k, \ell, \eta_1, \eta_2 \in \mathbb{Z}$, there is no quantum polynomial time adversary solves the* $\text{AMLWE-R}_{n,q,p,k,\ell,\eta_1,\eta_2}$ *problem.*

Under the above two assumptions, we can prove that the PKE scheme Π_{PKE} is IND-CPA secure and the KEM scheme Π_{KEM} is IND-CCA secure in the (quantum) random oracle model. Specifically, in random oracle model (ROM)[10], the adversary $\mathcal{A}$ can query a random oracle for any polynomial number of times. In quantum random oracle model (QROM)[11], the adversary $\mathcal{A}$ can query the quantum random oracle with superpositions of any input string for any polynomial number of times.

A.1 IND-CPA Security of $\Pi_{\mathbf{PKE}}$

We prove that Π_{PKE} is IND-CPA secure under the AMLWE and AMLWE-R assumptions. Formally, we have the following theorem.

Theorem 5. *Let $\mathcal{H} : \{0,1\}^n \to R_q^{k\times k}$ be a random oracle. If both problems* $\mathrm{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ *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.*

Proof. This proof proceeds via a sequence of games $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$. In the final game, we show that the advantage of any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ is negligible.

Game $\mathbf{G}_0$. This game is the real game for the IND-CPA security. In this game, the adversary $\mathcal{A}$ has access to a random oracle $\mathcal{H}$, and is given a public key $pk = (\rho, \bar{\mathbf{t}})$. Adversary $\mathcal{A}$ chooses two plaintexts $\mu_0, \mu_1 \in R_2$, and then obtains a challenge ciphertext $C = (\bar{\mathbf{u}}, \bar{v})$ on message μ_b, where b is a random bit. Finally, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ as the guess of $b \in \{0, 1\}$.

Game $\mathbf{G}_1$. This game is the same as $\mathbf{G}_0$, except that picking $\mathbf{t} \xleftarrow{\$} R_q^k$ at random and using $\bar{\mathbf{t}} = \lceil \mathbf{t} \rfloor_{q \to 2^{d_t}}$ as a part of pk.
If there exists a PPT adversary $\mathcal{A}$ that can distinguish $\mathbf{G}_1$ from $\mathbf{G}_0$, then we can construct a PPT algorithm $\mathcal{B}$ solving the $\text{AMLWE}_{n,q,k,k,\eta_1,\eta_2}$ problem. Specifically, given an instance $(\mathbf{A}, \mathbf{t})$ of the AMLWE problem, $\mathcal{B}$ aims to decide whether $(\mathbf{A}, \mathbf{t})$ is sampled from a uniform distribution of $R_q^{k \times k} \times R_q^k$. Formally, $\mathcal{B}$ behaves exactly as in game G_0, except that it chooses a random $\rho \xleftarrow{\$} \{0, 1\}^n$, programs the random oracle $\mathcal{H}$ such that $\mathcal{H}(\rho) = \mathbf{A}$, and returns $pk = (\rho, \bar{\mathbf{t}} = \lceil \mathbf{t} \rfloor_{q \to 2^{d_t}})$ to $\mathcal{A}$. Since $\mathcal{H}$ is a random oracle, the probability that $\mathcal{H}(\rho)$ has already been defined is negligible. If $(\mathbf{A}, \mathbf{t})$ is uniformly random in $R_q^{k \times k} \times R_q^k$, then $\mathcal{B}$ behaves as in game $\mathbf{G}_1$. Otherwise, $\mathcal{B}$ behaves as in game $\mathbf{G}_0$. In other words, if $\mathcal{A}$ can distinguish $\mathbf{G}_0$ and $\mathbf{G}_1$ with non-negligible probability, then $\mathcal{B}$ can solve the AMLWE problem with non-negligible probability.

Game $\mathbf{G}_2$. This is the same as $\mathbf{G}_1$, except that using uniformly random values to replace $\mathbf{u} = \mathbf{A}^T\mathbf{r} + \mathbf{e}_1$ and $v = \lceil \bar{\mathbf{t}} \rfloor_{2^{d_t} \to q}^T \mathbf{r} + e_2$ used in the generation of the challenge ciphertext.
If there exists a PPT adversary $\mathcal{A}$ who can distinguish $\mathbf{G}_2$ from $\mathbf{G}_1$, then we can construct a PPT algorithm $\mathcal{B}$ that solves the $\text{AMLWE-R}_{n,q,k,k,\eta_1,\eta_2}$ problem. Specifically, given an instance $(\mathbf{A}, \lceil \mathbf{t} \rfloor_{q \to 2^{d_t}}, \mathbf{u}, v)$ of the AMLWE-R problem, $\mathcal{B}$ aims to decide whether $(\mathbf{A}, \mathbf{t}, \mathbf{u}, v)$ is sampled from a uniform distribution in $R_q^{k \times k} \times R_q^k \times R_q^k \times R_q$. Formally, $\mathcal{B}$ chooses a random $\rho \xleftarrow{\$} \{0, 1\}^n$, programs the random oracle $\mathcal{H}$ such that $\mathcal{H}(\rho) = \mathbf{A}$, and returns $pk = (\rho, \bar{\mathbf{t}} = \lceil \mathbf{t} \rfloor_{q \to 2^{d_t}})$ to $\mathcal{A}$. After receiving two plaintexts $\mu_0, \mu_1 \xleftarrow{\$} R_2$ from $\mathcal{A}$, $\mathcal{B}$ picks $b \xleftarrow{\$} \{0, 1\}$ at random, computes $\bar{\mathbf{u}} = \lceil \mathbf{u} \rfloor_{q \to 2^{d_u}}, \bar{v} = \lceil v + \mu_b \cdot \lceil \frac{q}{2} \rfloor \rfloor_{q \to 2^{d_v}}$, and then gives the ciphertext $C = (\bar{\mathbf{u}}, \bar{v})$ to $\mathcal{A}$. If $(\mathbf{A}, \mathbf{t}, \mathbf{u}, v)$ is uniformly random in $R_q^{k \times k} \times R_q^k \times R_q^k \times R_q$, then $\mathcal{B}$ simulates as in game $\mathbf{G}_2$. Otherwise, $\mathcal{B}$ simulates as in in game $\mathbf{G}_1$. Thus, if $\mathcal{A}$ can distinguish $\mathbf{G}_2$ and $\mathbf{G}_1$ with non-negligible probability, then $\mathcal{B}$ can solve the AMLWE-R problem.

In the final game $\mathbf{G}_2$, μ_b in the challenge ciphertext is perfectly hidden by uniformly random v. Therefore, the advantage of $\mathcal{A}$ is 0 in $\mathbf{G}_2$, which completes the proof of Theorem 5.

A.2 IND-CCA Security of Π_{KEM}

Since Π_{KEM} is obtained by applying a slighlty tweaked Fujisaki-Okamoto (FO) transformation [24, 21] to the PKE scheme Π_{PKE}, given the results in [24, 12] and Theorem 5, we have the following theorem.

Theorem 6. *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 "secret key" even if the checks fails (i.e., implicit rejection). Furthermore, the paper [26] 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 [26] and Theorem 5, we have the following theorem.

Theorem 7. *Under the* AMLWE *assumption and the* AMLWE-R *assumption,* Π_{KEM} *is IND-CCA secure in the QROM.*

B Security Proof for $\Pi_{\mathbf{SIG}}$

We present an *asymmetric* variant of a standard lattice hardness problem MSIS, called AMSIS. To compress the public key, we need also a variant of the AMSIS problem, referred to as AMSIS-R, which is used to guarantee the unforgeability of signatures in the *classical random oracle model (ROM).* The unforgeability of our signature scheme in the *quantum random oracle model* (QROM) is based on an *asymmetric* variant of SelfTargetMSIS introduced by Ducas et al. [19], which is called SelfTargetAMSIS. In ROM, there is a (non-tight) reduction from AMSIS to SelfTargetAMSIS, which is very similar to the (non-tight) reduction from MSIS to SelfTargetMSIS [19].

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 $\mathrm{AMSIS}^{\infty}_{n,q,k,\ell_1,\ell_2,\beta_1,\beta_2}$ 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 $\mathbf{x} = \begin{pmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{pmatrix} \in R_q^{\ell_1+\ell_2}, \mathbf{x}_1 \in R_q^{\ell_1}, \mathbf{x}_2 \in R_q^{\ell_2}$.

Obviously, when $\beta_1 = \beta_2$, the AMSIS problem is the standard MSIS problem. In particular, we are easy to prove the following hardness relation of MSIS and AMSIS problems:

$$\mathrm{MSIS}_{n,q,k,\ell_1+\ell_2,\max(\beta_1,\beta_2)} \le \mathrm{AMSIS}_{n,q,k,\ell_1,\ell_2,\beta_1,\beta_2} \le \mathrm{MSIS}_{n,q,k,\ell_1+\ell_2,\min(\beta_1,\beta_2)}.$$

In other words, for suitable parameter choices, we can always guarantee that the AMSIS problem is hard.

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 AMSIS-R problem $\mathrm{AMSIS\text{-}R}^{\infty}_{n,q,d,k,\ell_1,\ell_2,\beta_1,\beta_2}$ 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 $\mathrm{SelfTargetAMSIS}^{\infty}_{n,q,k,\ell_1,\ell_2,\beta_1,\beta_2}$ 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 $\mathsf{H}(\mu, (\mathbf{I}_k\|\mathbf{A}\|\mathbf{t})\mathbf{y}) = c$ holds.

In the ROM, we prove that signature scheme Π_{SIG} satisfies strongly existential unforgeability under chosen message attacks (SUF-CMA) in Theorem 3. For the sake of simplicity, in the proof of Theorem 3, we will assume that the adversary $\mathcal{A}$ obtains $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ instead of $\mathbf{t}_1$ as the public key. Note that this assumption would be favorable to $\mathcal{A}$, as in the real scheme Π_{SIG} it only gets the high order bits $\mathbf{t}_1$ of $\mathbf{t}$. Therefore, in fact, the signature scheme Π_{SIG} may be even more difficult to break for the adversary. For convenience, we restate Theorem 3 in the following.

Theorem 8. *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* $\mathsf{H}_3 : \{0,1\}^* \to S^{\ell}_{\gamma_1-1}$ *are pseudo-random, and* $\mathsf{H}_2 : \{0,1\}^* \to \{0,1\}^{384}$ *is a collision-resistant hash function, then the scheme* Π_{SIG} *is SUF-CMA secure under the* $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ *and* $\text{AMSIS-R}^{\infty}_{n,q,d,k,\ell,4\gamma_2+2,2\gamma_1}$ *assumptions.*

Proof. This proof will proceed via a sequence of games, i.e., $\mathbf{G}_0, \mathbf{G}_1, \ldots, \mathbf{G}_4$. We will show that the difference between two successive games is negligible.

Game $\mathbf{G}_0$. This game is the real game, where the adversary $\mathcal{A}$ will obtain a public key pk and have access to a signing oracle. Finally, $\mathcal{A}$ will forge a signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$ on a message M.

Game $\mathbf{G}_1$. This game is the same as $\mathbf{G}_0$, except for aborting if there exists two messages M and M' output by $\mathcal{A}$ such that $\mathsf{H}_2(\mathsf{H}_2(pk), M) = \mathsf{H}_2(\mathsf{H}_2(pk), M')$ and $M \neq M'$.
Since H_2 is collision-resistant, $\mathbf{G}_1$ will abort with negligible probability. In other words, $\mathbf{G}_1$ is computationally indistinguishable from $\mathbf{G}_0$.

Game $\mathbf{G}_2$. This game is the same as $\mathbf{G}_1$, except that directly picking $\mathbf{y} \xleftarrow{\$} S^{\ell}_{\gamma_1-1}$ instead of computing $\mathbf{y}$ from H_3 and a key K for each signing query, and using the same randomness $\mathbf{y}$ for signing queries with the same message.
H_3 can be considered as a pseudo-random function with a key K and messages in the form of $\mu \| ctr$. If the adversary $\mathcal{A}$ can distinguish $\mathbf{G}_2$ from $\mathbf{G}_1$, then we can easily construct an algorithm breaking the pseudo-randomness of H_3. Therefore, we have that $\mathbf{G}_2$ is computationally indistinguishable from $\mathbf{G}_1$.

Game $\mathbf{G}_3$. This game is the same as $\mathbf{G}_2$, except that using a zero-knowledge simulator $\mathcal{S}$ without knowing secret key sk defined in the following Lemma 3 to respond all signing queries.
Based on Lemma 3, we know that $\mathbf{G}_3$ is computationally indistinguishable from $\mathbf{G}_2$.

Game $\mathbf{G}_4$. This game is the same as $\mathbf{G}_3$, except for using a uniform vector $\mathbf{t} \xleftarrow{\$} R_q^k$ rather than $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ as a part of public key pk.
If there exists a PPT adversary $\mathcal{A}$ distinguishing $\mathbf{G}_4$ from $\mathbf{G}_3$, then we can construct an algorithm $\mathcal{B}$ that solves the $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ problem. Specifically, given an instance $(\mathbf{A}, \mathbf{t}) \in R_q^{k\times\ell} \times R_q^k$, the goal of $\mathcal{B}$ is to decide whether $(\mathbf{A}, t)$ is uniformly distributed over $R_q^{k\times\ell} \times R_q^k$. $\mathcal{B}$ behaves exactly as in $\mathbf{G}_3$, except that it picks $\rho \xleftarrow{\$} \{0,1\}^{256}$ and programs random oracle H_1 such that $\mathsf{H}_1(\rho) = \mathbf{A}$. For a random $\rho \in \{0,1\}^{256}$, the probability that $\mathsf{H}_1(\rho)$ has already been defined is negligible. If $(\mathbf{A}, \mathbf{t})$ is uniformly distributed over $R_q^{k\times\ell} \times R_q^k$, then $\mathcal{B}$ behaves as in $\mathbf{G}_4$. If $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$, then $\mathcal{B}$ behaves as in $\mathbf{G}_3$. Therefore, if $\mathcal{A}$ can distinguish $\mathbf{G}_4$ from $\mathbf{G}_3$, then $\mathcal{B}$ can solve the $\text{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ problem.

In $\mathbf{G}_4$, if the adversary can forge a signature with non-negligible probability, then we can use the extractor $\mathcal{E}$ defined in the following Lemma 4 to find a solution of the AMSIS-R$^{\infty}_{n,q,d,k,\ell,4\gamma_2+2,2\gamma_1}$ problem with non-negligible probability.

Lemma 3. *If* H *is a random oracle, then there exists a PPT simulator* $\mathcal{S}$ *can generate the signature on any message without knowing the secret key* sk*.*

Proof. Prior to giving the construction of zero-knowledge simulator $\mathcal{S}$, we first analyze the probability distribution of $(\mathbf{z} = \mathbf{y} + c\mathbf{s}_1, c = \mathsf{H}_4(\mu\|\mathbf{w}_1))$ computed in $\Pi_{\mathrm{SIG}}.\mathsf{Sign}$, where the probability is taken from the randomness of $\mathbf{y}$ and random oracle H_4. We have the following:

$$\Pr[(\mathbf{z}, c)] = \Pr[c] \cdot \Pr[\mathbf{y} = \mathbf{z} - c\mathbf{s}_1 | c] \tag{3}$$

Since $\|c\mathbf{s}_1\|_\infty \leq \beta_1$ (with overwhelming probability), when $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$ we have $\|\mathbf{y}\|_\infty = \|\mathbf{z} - c\mathbf{s}_1\|_\infty \leq \gamma_1 - 1$, which is a valid value of $\mathbf{y}$. Therefore, if $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$, then the above probability 3 is exactly the same for each such tuple $(\mathbf{z}, c)$. If $\Pi_{\mathrm{SIG}}.\mathsf{Sign}$ outputs $\mathbf{z}$ such that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1$, then the resulting distribution of $(\mathbf{z}, c)$ will be uniformly random over $S^{\ell}_{\gamma_1-\beta_1-1} \times B_{60}$.

The simulation of signatures follows [8, 32, 19]. Specifically, simulator $\mathcal{S}$ is only given a public key $pk = (\rho, \mathbf{t})$, and can simulate a valid signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$ on any message M by programming the random oracle H_4. Firstly, $\mathcal{S}$ computes $\mathbf{A} = \mathsf{H}_4(\rho)$ and $\mu = \mathsf{H}_2(\mathsf{H}_2(pk)\|M)$. Then, $\mathcal{S}$ picks $(\mathbf{z}, c) \xleftarrow{\$} S^{\ell}_{\gamma_1-\beta_1-1} \times B_{60}$ such that the following relation holds:

$$\|\mathsf{LowBits}_q(\mathbf{Az} - c\mathbf{t}, 2\gamma_2)\|_\infty < \gamma_2 - \beta_2.$$

By the correctness, we have $\mathbf{w} - c\mathbf{s}_2 = \mathbf{Az} - c\mathbf{t}$. Thus, we have:

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

In other words, when $\|c\mathbf{s}_2\|_\infty \leq \beta_2$, by Lemma 2, we have:

$$\mathbf{r}_1 = \mathsf{HighBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2) = \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2) = \mathbf{w}_1$$

Thus, $\mathcal{S}$ does not need to perform the check that $\mathbf{r}_1 = \mathbf{w}_1$, and can always assume that it passes. Then, $\mathcal{S}$ can compute $\mathbf{w}_1 = \mathsf{HighBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2) = \mathsf{HighBits}_q(\mathbf{Az} - c\mathbf{t}, 2\gamma_2)$, and program H_4 such that

$$\mathsf{H}_4(\mu\|\mathbf{w}_1) = c.$$

If $\mathsf{H}_4(\mu\|\mathbf{w}_1)$ has not been defined, then the pair $(\mathbf{z}, c)$ simulated by $\mathcal{S}$ has the same distribution as the one in a genuine signature. Furthermore, based on the randomness of $\mathbf{y}$, we know the probability that $\mathsf{H}_4(\mu\|\mathbf{w}_1)$ have been defined is negligible. By the correctness, we have $\mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0 = \mathbf{Az} - c\mathbf{t}_1 \cdot 2^d$. Therefore, $\mathcal{S}$ can compute

$$\mathbf{h} = \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0) = \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{Az} - c\mathbf{t}_1 \cdot 2^d),$$

where $(\mathbf{t}_1, \mathbf{t}_0) = \mathsf{Power2Round}_q(\mathbf{t})$.

In conclusion, if we choose parameters such that $\|c\mathbf{s}_1\|_\infty \leq \beta_1$ and $\|c\mathbf{s}_2\|_\infty \leq \beta_2$ holds with overwhelming probability, then $\mathcal{S}$ can simulate a signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$ which has the same distribution as a genuine signature with overwhelming probability.

Lemma 4. *Forging a signature would imply that solving the* $\text{AMSIS-R}^{\infty}_{n,q,d,k,\ell,4\gamma_2+2,2\gamma_1}$ *problem. In particular, for uniformly random matrix* $\mathbf{A} \in R_q^{k\times\ell}$ *and vector* $\mathbf{t} \in R_q^k$*, this problem requires finding* $\mathbf{u}_1$*,* $\mathbf{u}_2$ *and* u_3 *such that*

$$\begin{aligned}
&\|\mathbf{u}_1\|_\infty \le 2\gamma_1, \|\mathbf{u}_2\|_\infty \le 4\gamma_2 + 2, \|u_3\|_\infty \le 2,\\
&\mathbf{A}\mathbf{u}_1 + \mathbf{u}_2 = u_3\mathbf{t}_1 \cdot 2^d,\\
&(\mathbf{u}_1, \mathbf{u}_2, u_3) \neq \mathbf{0},\\
&\mathbf{u}_2 \textit{has at most } 2\omega \textit{ coefficients of absolute value greater than } 2\gamma_2,
\end{aligned}$$

where $(\mathbf{t}_1, \mathbf{t}_0) = \mathsf{Power2Round}_q(\mathbf{t}, d)$.

Proof. If a PPT adversary $\mathcal{A}$ forges a valid signature in game $\mathbf{G}_4$, we show that there exists an extractor $\mathcal{E}$ who can solve the hardness problem stated in the above lemma. Given $(\mathbf{A}, \mathbf{t})$, $\mathcal{E}$ picks $\rho \xleftarrow{\$} \{0,1\}^{256}$ and programs H_1 such that $\mathsf{H}_1(\rho) = \mathbf{A}$. As ρ is chosen uniformly at random, $\mathsf{H}_1(\rho)$ has not been defined with overwhelming probability. $\mathcal{E}$ gives $(\rho, \mathbf{t})$ to $\mathcal{A}$ as the public key. For all signing queries from $\mathcal{A}$, $\mathcal{E}$ uses the simulator $\mathcal{S}$ defined in Lemma 3 to respond.

If $\mathcal{A}$ forges a valid signature $\sigma = (\mathbf{z}, \mathbf{h}, c)$ on a message M, then $\mathcal{A}$ must make the following query with overwhelming probability:

$$\mathsf{H}_4\big(\mu\|\mathbf{w}_1 = \mathsf{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2)\big) = c, \tag{4}$$

where $\mu = \mathsf{H}_2(\mathsf{H}_2(pk), M)$.

Case 1. If c is queried during a signing query, then $\mathcal{S}$ has responded another signature $(\mathbf{z}', \mathbf{h}', c)$ on a message M'. According to the winning condition of the SUF-CMA experiment, we have $(M', (\mathbf{z}', \mathbf{h}', c)) \neq (M, (\mathbf{z}, \mathbf{h}, c))$. Let $\mu' = \mathsf{H}_2(\mathsf{H}_2(pk), M')$. Thus, there exists the associated $\mathbf{w}_1'$ such that $\mathsf{H}_4(\mu\|\mathbf{w}_1) = c = \mathsf{H}_4(\mu'\|\mathbf{w}_1')$. Since H_4 is a random oracle, this implies $\mu = \mu'$ and $\mathbf{w}_1 = \mathbf{w}_1'$ with overwhelming probability. If $\mathbf{G}_4$ does not abort, then we have $M = M'$, as H_2 is collision-resistant. Since $\mathbf{w}_1 = \mathbf{w}_1'$, it must be that

$$\begin{aligned}
\mathsf{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2) &= \mathbf{w}_1,\\
\mathsf{UseHint}_q(\mathbf{h}', \mathbf{A}\mathbf{z}' - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2) &= \mathbf{w}_1.
\end{aligned}$$

If $\mathbf{z} = \mathbf{z}'$, then by Lemma 1 we must have $\mathbf{h} = \mathbf{h}'$, which is in contradiction with that $(M', (\mathbf{z}', \mathbf{h}', c)) \neq (M, (\mathbf{z}, \mathbf{h}, c))$. Thus, we must have that $\mathbf{z} \neq \mathbf{z}'$. By Lemma 1, we have the following relations:

$$\begin{aligned}
\|\mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2\|_\infty &\le 2\gamma_2 + 1,\\
\|\mathbf{A}\mathbf{z}' - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2\|_\infty &\le 2\gamma_2 + 1.
\end{aligned}$$

By the triangular inequality, this implies that

$$\|\mathbf{A}(\mathbf{z} - \mathbf{z}')\|_\infty \le 4\gamma_2 + 2.$$

In other words, there exists vectors $\mathbf{u}_1 \in R_q^\ell, \mathbf{u}_2 \in R_q^k$ such that $\|\mathbf{u}_1\|_\infty \le 2\gamma_1, \|\mathbf{u}_2\|_\infty \le 4\gamma_2 + 2$ and $\mathbf{A}\mathbf{u}_1 + \mathbf{u}_2 = \mathbf{0}$, where $\mathbf{u}_1 = (\mathbf{z} - \mathbf{z}') \neq \mathbf{0}$. As $\mathbf{h}$ and $\mathbf{h}'$ have all except for ω elements equal to 0, by Lemma 1, we know that all but ω coefficients of $\mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2$ and $\mathbf{A}\mathbf{z}' - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1' \cdot 2\gamma_2$ have magnitude at most γ_2. Hence, all but 2ω coefficients

of $\mathbf{u}_2$ have magnitude at most $2\gamma_2$. In all, $\mathcal{E}$ finds a solution $(\mathbf{u}_1, \mathbf{u}_2, 0) \neq \mathbf{0}$ of the AMSIS problem.

Case 2. Now we handle the case that c is queried by adversary $\mathcal{A}$. That is, $\mathcal{A}$ made a query $(\mu', \mathbf{w}_1')$ to random oracle H_4 and obtained c such that $\mathsf{H}(\mu' \| \mathbf{w}_1') = c$. In the ROM, we have $\mu' = \mu = \mathsf{H}_2(\mathsf{H}_2(pk), M)$ and $\mathbf{w}_1' = \mathbf{w}_1 = \mathsf{UseHint}_q(\mathbf{h}, \mathbf{Az} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2)$, with overwhelming probability.

From the standard forking lemma [39, 9], $\mathcal{E}$ can with non-negligible probability extract two signatures $(\mathbf{z}, \mathbf{h}, c)$ and $(\mathbf{z}', \mathbf{h}', c')$ for $c \neq c'$ such that

$$\begin{aligned} &\mathsf{UseHint}_q(\mathbf{h}, \mathbf{Az} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2) = \mathbf{w}_1, \\ &\mathsf{UseHint}_q(\mathbf{h}', \mathbf{Az}' - c'\mathbf{t}_1 \cdot 2^d, 2\gamma_2) = \mathbf{w}_1. \end{aligned}$$

By Lemma 1, we have:

$$\begin{aligned} \|\mathbf{Az} - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2\|_\infty \leq 2\gamma_2 + 1, \\ \|\mathbf{Az}' - c'\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2\|_\infty \leq 2\gamma_2 + 1. \end{aligned}$$

By the triangular inequality, this implies that

$$\|\mathbf{A}(\mathbf{z} - \mathbf{z}') - (c - c')\mathbf{t}_1 \cdot 2^d\|_\infty \leq 4\gamma_2 + 2.$$

In other words, there exists $\mathbf{u}_1 \in R_q^\ell, \mathbf{u}_2 \in R_q^k, u_3 \in R_q$ such that $\|\mathbf{u}_1\|_\infty \leq 2\gamma_1, \|\mathbf{u}_2\|_\infty \leq 4\gamma_2 + 2, \|u_3\|_\infty \leq 2$ and $\mathbf{Au}_1 + \mathbf{u}_2 = u_3\mathbf{t}_1 \cdot 2^d$, where $\mathbf{u}_1 = (\mathbf{z} - \mathbf{z}'), u_3 = c - c' \neq 0$. Since the number of 1's of $\mathbf{h}$ and $\mathbf{h}'$ is at most ω, by Lemma 1 we know that all but ω coefficients of $\mathbf{Az} - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2$ and $\mathbf{Az}' - c\mathbf{t}_1 \cdot 2^d - \mathbf{w}_1 \cdot 2\gamma_2$ have magnitude at most γ_2. Therefore, $\mathbf{u}$ has at most 2ω coefficients whose absolute value is greater than $2\gamma_2$. In conclusion, $\mathcal{E}$ finds a solution $(\mathbf{u}_1, \mathbf{u}_2, u_3) \neq \mathbf{0}$ of the AMSIS-R problem.

For convenience, we restate Theorem 4 in the following.

Theorem 9. *In the quantum random oracle model, the 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* $\mathrm{SelfTargetAMSIS}^{\infty}_{\mathsf{H}_4,n,q,k,\ell_1,\ell_2,4\gamma_2,(\gamma_1-\beta_1)}$.

Intuitively, the AMLWE assumption is used to guarantee the security against key recovery, the SelfTargetAMSIS assumption is used to protect against the forgery of signature on a new message, and the AMSIS assumption is required for strong unforgeability.

The proof of Theorem 4 can be obtained by extending the proof of the Dilithium signature scheme in QROM by Kiltz et al. [28]. Here, we only give a proof sketch. For the sake of simplicity, we can assume that no collision occurs for hash function H_2 and H_3 is a perfect pseudo-random function, based on the proof of Theorem 3.

Kiltz et al. [28] show that for a zero-knowledge deterministic signature scheme, if an adversary, who has quantum access to oracle H_4 and classical access to a signing oracle, can forge a signature on a new message, then there exists another adversary who can produce a forgery without querying to the signing oracle (i.e., the adversary only obtains the public key). The latter security model is called unforgeability under no message attacks (UF-NMA). In QROM, we can assume that the public key is $(\mathbf{A}, \mathbf{t})$ rather than $(\rho, \mathbf{t})$ for the sake of simplicity, as we can always program H_1 such that $\mathsf{H}_1(\rho) = \mathbf{A}$. Under

the AMLWE assumption, we know that the public key $(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2) \in R_q^{k\times\ell} \times R_q^k$ is computationally indistinguishable from $(\mathbf{A}, \mathbf{t}) \in R_q^{k\times\ell} \times R_q^k$ for a uniformly random $\mathbf{t} \in R_q^k$. Based on Lemma 3 and the formal proof in [28], one can prove that the signature scheme Π_{SIG} is zero-knowledge.

Thus, under the $\mathrm{AMLWE}_{n,q,k,\ell,\eta_1,\eta_2}$ assumption, we only need to analyze the UF-NMA security of Π_{SIG}, when the adversary is given a uniformly random $(\mathbf{A}, \mathbf{t})$. The adversary needs to output a valid message-signature pair $(M, \sigma = (\mathbf{z}, \mathbf{h}, c))$ such that

$$\begin{aligned}
&\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1 \\
&\mathsf{H}_4(\mu \| \mathsf{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2)) = c \\
&\text{the number of 1's in } \mathbf{h} \text{ is } \leq \omega,
\end{aligned}$$

where $\mu = \mathsf{H}_2(\mathsf{H}_2(pk)\|M)$. From Lemma 1, we can rewrite

$$\mathsf{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2) = \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d + \mathbf{u}, \tag{5}$$

where $\|\mathbf{u}\|_\infty \leq 2\gamma_2 + 1$. Furthermore, at most ω coefficients of $\mathbf{u}$ have magnitude greater than γ_2. If we write $\mathbf{t} = \mathbf{t}_1 \cdot 2^d + \mathbf{t}_0$ where $\|\mathbf{t}_0\|_\infty \leq 2^{d-1}$, then we can re-write equation (5) as follows:

$$\mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d + \mathbf{u} = \mathbf{A}\mathbf{z} - c(\mathbf{t} - \mathbf{t}_0) + \mathbf{u} = \mathbf{A}\mathbf{z} - c\mathbf{t} + (c\mathbf{t}_0 + \mathbf{u}) = \mathbf{A}\mathbf{z} - c\mathbf{t} + \mathbf{u}'$$

We note that the worst-case upper-bound of $\mathbf{u}'$ is

$$\begin{aligned}
\|\mathbf{u}'\|_\infty \leq \|c\mathbf{t}_0\|_\infty + \|\mathbf{u}\|_\infty &\leq \|c\|_1 \cdot \|\mathbf{t}_0\|_\infty + \|\mathbf{u}\|_\infty \\
&\leq 60 \cdot 2^{d-1} + 2\gamma_2 + 1 < 4\gamma_2
\end{aligned}$$

Therefore, if a (quantum) adversary $\mathcal{A}$ can successfully forge a signature on a new message, then it is able to find $(M, (\mathbf{z}, c, \mathbf{u}'))$ such that $\|\mathbf{z}\|_\infty < \gamma_1 - \beta_1, \|c\|_\infty = 1, \|\mathbf{u}'\|_\infty < 4\gamma_2$, $M \in \{0,1\}^*$, and the following holds:

$$\mathsf{H}_4\left(\mathsf{H}_2(\mathsf{H}_2(pk), M),\ [\mathbf{I}_k \| \mathbf{A} \| \mathbf{t}] \cdot \begin{bmatrix} \mathbf{u}' \\ \mathbf{z} \\ -c \end{bmatrix}\right) = c, \tag{6}$$

where $\mu = \mathsf{H}_2(\mathsf{H}_2(pk)\|M)$. Since $(\mathbf{A}, \mathbf{t})$ is uniformly random, the problem defined in the above equation (6) is exactly the definition of the SelfTargetAMSIS problem.

To prove strong unforgeability, we only have to consider the case that the adversary sees a message/signature pair $(M, \sigma = (\mathbf{z}, \mathbf{h}, c))$ from the signing oracle and forges a signature $\sigma' = (\mathbf{z}', \mathbf{h}', c)$ on the same message M. From Case 1 of Lemma 4, this implies that finding non-zero vectors $\mathbf{u}_1 \in R_q^\ell, \mathbf{u}_2 \in R_q^k$ such that $\|\mathbf{u}_1\|_\infty \leq 2(\gamma_1 - \beta_1), \|\mathbf{u}_2\|_\infty \leq 4\gamma_2 + 2$ and $\mathbf{A}\mathbf{u}_1 + \mathbf{u}_2 = \mathbf{0}$. In other words, the adversary needs to find a solution of the AMSIS problem.

C Concrete Attacks against AMLWE

Solvers for LWE mainly include primal attacks, dual attacks (against the underlying lattice problems) and direct solving algorithms such as BKW and Arora-Ge [4]. 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 [19, 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 lose generality, we consider the case that secret vector and error vector in $\mathrm{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ 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 $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \leftarrow \chi_{\alpha_1}^n$ and $\mathbf{e} \leftarrow \chi_{\alpha_2}^m$.

C.1 Primal Attack and Its Variants

The basic intuition behind primal attack is to convert ALWE problem to the corresponding bounded distance decoding (BDD) problem or shortest vector problem by embedding. The classification of primal attacks is mainly determined by the ways of embedding. We consider primal attack and its variants aiming at $\mathrm{ALWE}_{n,q,m,\alpha_1,\alpha_2}$.

Traditional primal attack. The primal attack using Kannan's embedding [27, 3] translates the LWE problem to unique Shortest Vector Problem (uSVP) on lattice. First, define a m-dimensional lattice

$$\Lambda = \{\mathbf{y} \in \mathbb{Z}^m | \mathbf{y} = \mathbf{A}\mathbf{x} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\} \ ,$$

For $m > n$ the $m \times n$ matrix $\mathbf{A}$ has full rank with high probability. Assume WLOG that the first n rows of $\mathbf{A}$ are independent (otherwise swap the rows to make it), and denote by $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times n}$ the submatrix by keeping the first n rows of $\mathbf{A}$, i.e., $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix}$. Denote $\mathbf{A}_1$'s inverse by $\mathbf{A}_1^{-1} \in \mathbb{Z}_q^{n\times n}$, and let $\mathbf{A}' = \begin{pmatrix} \mathbf{I}_n \\ \mathbf{A}_2\mathbf{A}_1^{-1} \end{pmatrix}$. Then, we have

$$\Lambda = \{\mathbf{y} | \mathbf{y} = \mathbf{A}\mathbf{x} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\} = \{\mathbf{y} | \mathbf{y} = \mathbf{A}'\mathbf{x} \bmod q, \mathbf{x} \in \mathbb{Z}_q^n\} \subset \mathbb{Z}^m \ ,$$

and the columns of the following matrix

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & 0 \\ \mathbf{A}_2\mathbf{A}_1^{-1} & q\mathbf{I}_{m-n} \end{pmatrix} \in \mathbb{Z}^{m\times m}$$

constitutes a basis of Λ. It is easy to see $\det(\Lambda) = q^{m-n}$. We have by $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \bmod q$ that the distance between vector $\mathbf{b}$ and lattice Λ is $\|\mathbf{e}\|$. If we could find the closest

lattice point $\mathbf{u} \in \Lambda$ to $\mathbf{b}$, then we have $\mathbf{e} = \mathbf{b} - \mathbf{u}$ with high probability. In other words, the problem can be reduced to bounded distance decoding (BDD) problem on Λ, and then one applies the nearest planes algorithm [30] to solve. Furthermore, using Kannan's embedding [27, 3] one can reduce BDD to unique shortest vector problem (uSVP). In detail, consider lattice Λ' spanned by the column vectors of the following matrix:

$$\mathbf{B}' = \begin{pmatrix} \mathbf{I}_n & 0 & \\ \mathbf{A}_2\mathbf{A}_1^{-1} & q\mathbf{I}_{m-n} & \mathbf{b} \\ 0 & 0 & t \end{pmatrix} \in \mathbb{Z}^{(m+1)\times(m+1)},$$

where $t \in \mathbb{Z}$ is an ajustable parameter. Theoretically, the algorithm works best when $t = \|\mathbf{e}\|$. However, empirical experiments suggest that best performance is achieved when $t = 1$, which is the typical value we choose. In this case, we have that $\mathbf{v} = (\mathbf{e}^T, 1)^T \in \mathbb{Z}^{m+1}$ is the unique shortest vector in Λ' with high probability. To summarize, we work on uSVP problem in Λ' to fine out the error vector $\mathbf{e} \in \mathbb{Z}^m$, which is in turn used to solve the system of linear congruences to recover $\mathbf{s} \in \mathbb{Z}_q^n$, i.e, $\mathbf{As} = \mathbf{b} - \mathbf{e}$. Besides, since every element in vector $\mathbf{e}$ is sampled from subgaussian χ_{α_2}, we have $\|\mathbf{v}\| \approx \|\mathbf{e}\| \approx \alpha_2\sqrt{m}$, namely, $\mathbf{v}$ is of small norm.

Primal attack: variant 1. When $\alpha_1 = \alpha_2$, it gives the best known primal attack [7, 5]. Define lattice

$$\Lambda = \{\mathbf{v} = (\mathbf{x}^T, \mathbf{y}^T, z)^T \in \mathbb{Z}^{n+m+1} | (\mathbf{A}\|\mathbf{I}_m\| - \mathbf{b})\mathbf{v} = 0 \bmod q\},$$

It is easy to verify that the column vectors of

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & 0 & 0 \\ -\mathbf{A} & q\mathbf{I}_m & \mathbf{b} \\ 0 & 0 & 1 \end{pmatrix} \in \mathbb{Z}^{(m+n+1)\times(m+n+1)}$$

constitute a basis of Λ. Clearly, the dimension of lattice Λ is $d = m + n + 1$. Further, we have that $\det(\Lambda) = q^m$ and $\mathbf{v} = (\mathbf{s}^T, \mathbf{e}^T, 1)^T \in \mathbb{Z}^{n+m+1}$ is a short vector in Λ (i.e., $\|\mathbf{v}\| \approx \sqrt{\alpha_1^2 n + \alpha_2^2 m}$ as $\mathbf{s} \leftarrow \chi_{\alpha_1}^n$ and $\mathbf{e} \leftarrow \chi_{\alpha_2}^n$). Therefore, we can recover $\mathbf{s} \in \mathbb{Z}_q^m$ by working out the (u)SVP on Λ.

Primal attack: variant 2. This variant is adapted from [7, 5], which is most effective for $\alpha_1 \neq \alpha_2$ but the values of α_1 and α_2 are close. Formally, let $c = \alpha_2/\alpha_1$ and consider lattice

$$\Lambda = \left\{ \mathbf{v} = \begin{pmatrix} c\mathbf{x} \\ \mathbf{y} \\ \alpha_2 z \end{pmatrix} \in \mathbb{R}^{n+m+1} \middle| (\mathbf{A}\|\mathbf{I}_m\| - \mathbf{b})\mathbf{u} = 0 \bmod q, \mathbf{u} = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \\ z \end{pmatrix} \in \mathbb{Z}^{n+m+1} \right\},$$

Then the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} c\mathbf{I}_n & 0 & 0 \\ -\mathbf{A} & q\mathbf{I}_m & \mathbf{b} \\ 0 & 0 & \alpha_2 \end{pmatrix} \in \mathbb{R}^{(n+m+1)\times(n+m+1)},$$

constitutes a basis of Λ. We observe that Λ is of dimension $m+n+1$, $\det(\Lambda) = \alpha_2 c^n q^m$, and that $\mathbf{v} = (c\mathbf{s}^T, \mathbf{e}^T, \alpha_2)^T \in \mathbb{R}^{m+n+1}$ is a short vector in Λ (i.e., $\|\mathbf{v}\| \approx \alpha_2\sqrt{n+m+1}$). Therefore, we recover $\mathbf{s}$ by solving (u)SVP on lattice Λ.

Intuitively, the basic idea of this variant is to scale the components of target vector to the same magnitude . Experiments show that solvers for (u)SVP are more likely to output vector with components of roughly the same magnitude.

Estimating the computational cost of primal attacks. The best known SVP solver is the BKZ-b lattice basis reduction algorithm and its variants. Given a d-dimensional basis as input, the algorithm translates the lattice basis reduction problem to the SVP problem on b-dimensional sublattice ($b < d$). BKZ-b repeats the SVP solving procedure $O(d)$ times (the actual number may depend on the specific algorithm used) on different b-dimensional sublattices to reduce the norm of the basis (of the original d-dimensional basis). Therefore, the cost of BKZ-b basis reduction algorithm is mainly decided by SVP solving algorithm for b-dimensional lattice.

Assume that the reduced basis obtained from running BKZ-b on d-dimensional lattice basis $\mathbf{B} \in \mathbb{Z}^{d\times d}$ satisfies the Geometry Series Assumption (GSA), which is the best scenario for adversary. Let $\hat{\mathbf{B}} = (\hat{\mathbf{b}}_1, \ldots, \hat{\mathbf{b}}_d)$ be a reduced basis, then we have

$$\|\hat{\mathbf{b}}_1\| = \delta^d \det(\Lambda)^{1/d}, \qquad \|\hat{\mathbf{b}}_i^*\| = \delta^{-2d(i-1)/(d-1)}\|\hat{\mathbf{b}}_1\|,$$

where $\delta = ((\pi b)^{1/b} \cdot b/2\pi e)^{\frac{1}{2(b-1)}}$ [17], $\hat{\mathbf{B}}^* = (\hat{\mathbf{b}}_1^*, \ldots, \hat{\mathbf{b}}_d^*)$ is the orthogonalized matrix of $\hat{\mathbf{B}}$ (i.e., $\hat{\mathbf{b}}_1 = \hat{\mathbf{b}}_1^*$). Specifically, [5, 3] show that when the norm of the projection of the unique shortest vector $\mathbf{v}$ into the space spanned by the last b orthogonal vectors $(\hat{\mathbf{b}}_{d-b+1}^*, \ldots, \hat{\mathbf{b}}_d^*)$ is less than $\|\hat{\mathbf{b}}_{d-b+1}^*\|$, then the BKZ-$b$ basis reduction algorithm recovers $\mathbf{v}$.

The norm of $\mathbf{v}$'s projection into the space spanned by b orthogonal vectors $(\hat{\mathbf{b}}_{d-b+1}^*, \ldots, \hat{\mathbf{b}}_d^*)$ approximately equals $\ell\sqrt{b/d}$, where $\ell = \|\mathbf{v}\|$. That is, we assume every component of vector $\mathbf{v}$'s projection is of roughly the same size. In this case, the computational cost of primal attack and its variant is mainly decided by the minimal value of b that satisfies the constraint

$$\ell\sqrt{b/d} \leq \delta^{(-d^2+2db-d)/(d-1)} \det(\Lambda)^{1/d} \ . \tag{7}$$

Following [5], our (conservative) estimation of the complexity of solving uSVP on d-dimensional lattice is based on that of solving SVP on b-dimensional lattice, where b satisfy inequality (7) (recall that the SVP solving algorithm is invoked on b-dimensional lattice several times). Furthermore, same as previous works (e.g., [5, 19, 12]), we use $\text{cost}_b = 2^{0.292b}$ and $\text{cost}_b = 2^{0.265b}$ to distinguish between the complexities of classical algorithm and quantum algorithm (for solving SVP on b-dimensional lattice) respectively. In conclusion, the complexity of primal attack and its variants in the above conservative model is mainly decided by the b satisfying (7).

C.2 Dual Attack and Its Variants

Dual attack translates the LWE problem to a SIS problem, whose solution is used to solve the decisional LWE by distinguishing between subgaussians (for certain parameters) and uniform distribution over $\mathbb{Z}_q^n$. We mainly consider the following dual attack and its variants aiming at solving the $\text{ALWE}_{n,q,m,\alpha_1,\alpha_2}$ problem.

Traditional Dual Attack. The first dual attack was given in [36]. The attack only cares about the error distribution of the LWE-like problem, and thus does not distinguish between LWE and ALWE. In other words, the traditional dual attack solves ALWE the same way as it does to LWE, and it appears more efficient when $\alpha_1 \gg \alpha_2$. Formally, let

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^m | \mathbf{A}^T\mathbf{x} = \mathbf{0} \bmod q\},$$

be a lattice of dimension $d = m \gg n$. Note that $\mathbf{A}$ has full rank with high probability and we assume WLOG the submatrix $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times n}$ made up by the first n columns of $\mathbf{A}^T$ has rank n and $\mathbf{A}^T = (\mathbf{A}_1 \| \mathbf{A}_2) \in \mathbb{Z}_q^{n\times m}$. Let $\mathbf{A}' = (\mathbf{I}_n \| \mathbf{A}_1^{-1}\mathbf{A}_2)$. Then, we have that the column vectors of

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^m | \mathbf{A}^T\mathbf{x} = \mathbf{0} \bmod q\} = \{\mathbf{x} \in \mathbb{Z}^m | \mathbf{A}'\mathbf{x} = \mathbf{0} \bmod q\},$$

and matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1}\mathbf{A}_2 \\ 0 & \mathbf{I}_{m-n} \end{pmatrix} \in \mathbb{Z}^{m\times m}$$

constitute a basis of lattice Λ and $\det(\Lambda) = q^n$. Traditional dual attack first solves the SIS problem by finding out a short vector $\mathbf{v} \in \mathbb{Z}^m$ with $\mathbf{A}^T\mathbf{v} = \mathbf{0} \bmod q$ in lattice Λ with basis $\mathbf{B}$. It then computes $u = \langle \mathbf{v}, \mathbf{b} \rangle = \langle \mathbf{v}, \mathbf{e} \rangle \bmod q$ in order to distinguish between subgaussian and uniform distributions. Specifically, if the norm $\ell = \|\mathbf{v}\|$ is relatively small, then the value of $\langle \mathbf{v}, \mathbf{e} \rangle$ is relatively small too and u can be seen as random variable drawn from subgaussian distribution with standard variance $\ell\alpha_2$, which can be distinguished from uniform distribution on $\mathbb{Z}_q$ with probability $4\exp(-2\pi^2\tau^2)$, where $\tau = \ell\alpha_2/q$. Therefore, intuitively using large value for α_2 improves the hardness of ALWE against traditional dual attacks.

Dual Attack: Variant 1. We generalize [5] so that the attack [5] falls into a special case (of our generalized attack) for $\alpha_1 = \alpha_2$. Formally, define the following lattice of dimension $d = m + n$ ($m \gg n$)

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T)^T \in \mathbb{Z}^{m+n} | \mathbf{A}^T\mathbf{x} = \mathbf{y} \bmod q\},$$

Likewise, we assume WLOG that $\mathbf{A}^T = (\mathbf{A}_1 \| \mathbf{A}_2) \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times n}$ has full rank. Let $\mathbf{A}' = (\mathbf{I}_n \| \mathbf{A}_1^{-1}\mathbf{A}_2)$, then we have

$$\begin{aligned} \Lambda &= \{(\mathbf{x}^T, \mathbf{y}^T)^T | \mathbf{A}^T\mathbf{x} = \mathbf{y} \bmod q\} \\ &= \{(\mathbf{x}^T, \mathbf{y}^T)^T | \mathbf{A}'\mathbf{x} = \mathbf{A}_1^{-1}\mathbf{y} \bmod q\} \ , \end{aligned}$$

and the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1}\mathbf{A}_2 & \mathbf{A}_1^{-1} \\ 0 & \mathbf{I}_{m-n} & 0 \\ 0 & 0 & \mathbf{I}_n \end{pmatrix} \in \mathbb{Z}^{(m+n)\times(m+n)}$$

constitutes a basis of lattice Λ, where $\det(\Lambda) = q^n$. We first find out a short vector $\mathbf{v} = (\mathbf{x}^T, \mathbf{y}^T)^T \in \mathbb{Z}^{m+n}$ with $\mathbf{A}^T\mathbf{x} = \mathbf{y} \bmod q$ on $(m+n)$-dimensional lattice Λ with basis $\mathbf{B}$. Then, we compute

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = \langle \mathbf{y}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

and thus distinguish subgaussian and uniform distributions. In particular, when the norm $\ell = \|\mathbf{v}\|$ is relatively small, then the values of $\langle \mathbf{y}, \mathbf{s} \rangle$ and $\langle \mathbf{x}, \mathbf{e} \rangle$ are relatively small too. Furthermore, assume the components of $\mathbf{v}$ are of approximately the same magnitude, then $u = \langle \mathbf{y}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ can be regarded as following subgaussian distribution with parameter $\ell\sqrt{\frac{\alpha_1^2 m + \alpha_2^2 n}{m+n}}$. In this case, one can distinguish subgaussian-distributed u from uniform element with probability $4\exp(-2\pi^2\tau^2)$, where $\tau = \ell\sqrt{\frac{\alpha_1^2 m + \alpha_2^2 n}{m+n}}/q$.

Dual Attack: Variant 2. The variant is the most effective dual attack [5] on standard LWE (ALWE for $\alpha_1 = \alpha_2$) and we generalize to the ALWE problem where α_1 and α_2 are (not necessarily equal but) of approximately the same size. Formally, define lattice

$$\Lambda = \{(\mathbf{x}^T, \mathbf{y}^T/c)^T \in \mathbb{R}^{m+n} | \mathbf{A}^T\mathbf{x} = \mathbf{y} \bmod q, \mathbf{x} \in \mathbb{Z}^m, \mathbf{y} \in \mathbb{Z}^n\} \ ,$$

where Λ has dimension $d = m + n$, $m \gg n$ and $c = \frac{\alpha_2}{\alpha_1}$. Assume WLOG that $\mathbf{A}^T = (\mathbf{A}_1 \| \mathbf{A}_2) \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ has full rank, and let $\mathbf{A}' = (\mathbf{I}_n \| \mathbf{A}_1^{-1}\mathbf{A}_2)$. Then, we have

$$\begin{aligned}\Lambda &= \{(\mathbf{x}^T, \mathbf{y}^T/c)^T | \mathbf{A}^T\mathbf{x} = \mathbf{y} \bmod q\} \\ &= \{(\mathbf{x}^T, \mathbf{y}^T/c)^T | \mathbf{A}'\mathbf{x} = \mathbf{A}_1^{-1}\mathbf{y} \bmod q\} \ ,\end{aligned}$$

and the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A}_1^{-1}\mathbf{A}_2 & \mathbf{A}_1^{-1} \\ 0 & \mathbf{I}_{m-n} & 0 \\ 0 & 0 & \frac{1}{c}\mathbf{I}_n \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$$

constitutes a basis of lattice Λ with $\det(\Lambda) = (q/c)^n$. Likewise, we find out a short vector $\mathbf{v} = (\mathbf{x}^T, \hat{\mathbf{y}}^T)^T \in \mathbb{R}^{m+n}$ with $\mathbf{A}^T\mathbf{x} = c\hat{\mathbf{y}} \bmod q$ from $(m+n)$-dimensional lattice Λ with basis $\mathbf{B}$, and then compute computes

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

and therefore distinguish the above u from uniform. Specifically, the values of $c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle$ and $\langle \mathbf{x}, \mathbf{e} \rangle$ are relatively small for small $\ell = \|\mathbf{v}\|$. Thus, $u = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ can be almost regarded as following subgaussian distribution with parameter $\ell\alpha_2$. This enable efficient algorithms that distinguish the subgaussian-distributed u from uniform with probability $4\exp(-2\pi^2\tau^2)$, where $\tau = \ell\alpha_2/q$.

At last, we stress that the variant is symmetric about parameters α_1 and α_2. We can use $c' = \alpha_1/\alpha_2$ to launch the dual attack, which takes the same complexity as that using $c = \alpha_2/\alpha_1$ under our estimation model.

Dual Attack: Variant 3. This variant is derived from [2], and is similar to variant 2, where the main difference is the choice of c. First, we compute a short vector $\mathbf{v} = (\mathbf{x}^T, \hat{\mathbf{y}}^T)^T \in \mathbb{R}^{m+n}$ with $\mathbf{A}^T\mathbf{x} = c\hat{\mathbf{y}} \bmod q$, where $c = \frac{\alpha_2\sqrt{m}}{\alpha_1\sqrt{n}}$. Then, we compute

$$u = \langle \mathbf{x}, \mathbf{b} \rangle = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle \bmod q,$$

and thus distinguish the above u from uniformly random element from $\mathbb{Z}_q$. That is, we consider $u = c \cdot \langle \hat{\mathbf{y}}, \mathbf{s} \rangle + \langle \mathbf{x}, \mathbf{e} \rangle$ as being distributed to subgaussian with parameter

$\ell\alpha_2\sqrt{\frac{2m}{m+n}}$, where $\ell = \|\mathbf{v}\|$, and thus efficiently distinguish it from uniformly random element over $\mathbb{Z}_q$ with probability $4\exp(-2\pi^2\tau^2)$ probability, where $\tau = \ell\alpha_2\sqrt{\frac{2m}{m+n}}/q$. Likewise, the attack is also symmetric between α_1 and α_2, i.e., the attack using coefficient $c' = \frac{\alpha_1\sqrt{n}}{\alpha_2\sqrt{m}}$ takes the same complexity as that using $c = \frac{\alpha_2\sqrt{m}}{\alpha_1\sqrt{n}}$ under our estimation model.

Estimating Computational Cost of Dual Attack. As shown in the previous sections, the computational costs of dual attack and its variants is mainly dominated by the cost of solving SIS problem and transforming distinguishing attacks into secret recovery attacks (with success probability more than half) for the ALWE problem. We use the model of primal attack to estimate the cost of solving SIS problem. That is, applying BKZ-b to reduce the basis of d-dimensional lattice yields a short vector $\mathbf{v}$ of norm $\ell = \|\mathbf{v}\| = \delta^d \det(\Lambda)^{1/d}$. Substituting this into the previous analysis, we distinguish ALWE problem with advantage $\epsilon = 4\exp(-2\pi^2\tau^2)$, where τ is fully decided by ℓ and the actual dual attack algorithm. To obtain the final secret recovery attack with probability greater than $1/2$, we need $1/\epsilon^2$ short vectors. Taking into account that each invocation of the sieving yields $2^{0.2075b}$ short vectors, we need to repeat it $R = 1/\max(1, 1/(2^{0.2075b}\epsilon^2))$ times.

Following the cost estimation model for primal attacks, we use the complexity of SVP solving algorithm on b-dimensional lattice to (conservatively) estimate the cost of solving SVP on d-dimensional lattice (since the latter will invoke the former many times). Furthermore, we use $\mathrm{cost}_b = 2^{0.292b}$ and $\mathrm{cost}_b = 2^{0.265b}$ as estimated complexities of classical and quantum algorithms (for solving SVP problem on b-dimensional lattices) respectively. Under this model, the complexity of dual attack is estimated to

$$1/\max(1, 1/(2^{0.2075b} \cdot 16\exp(-4\pi^2\tau^2))) \cdot \mathrm{cost}_b, \tag{8}$$

where τ is decided by b and other parameters of the dual attack.

D Concrete Attacks against AMSIS

In this section, we consider the best known attacks and their variants against AMSIS. 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 $\mathbf{x} \in \mathbb{Z}^{m_1+m_2}$ to the ℓ_∞-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 [19] 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 $\mathrm{ASIS}^\infty_{n,q,m_1,m_2,\beta_1,\beta_2}$.

D.1 Two-Norm Attack and Its Variants

Traditional ℓ_2-Norm Attack. This attack was originally used to solve SVP. Formally, define

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^{m_1+m_2} | (\mathbf{I}_n\|\mathbf{A})\mathbf{x} = 0 \bmod q\}.$$

Then, the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A} \\ 0 & \mathbf{I}_{m_1+m_2-n} \end{pmatrix} \in \mathbb{Z}^{(m_1+m_2)\times(m_1+m_2)}$$

constitute a basis of lattice Λ of dimension $d = m_1 + m_2$ and $\det(\Lambda) = q^n$. The complexity of this algorithm is that of applying BKZ-b, where the resulting ℓ_2-norm solution satisfying $\|\mathbf{x}\| \leq \beta = \min(\beta_1, \beta_2)$ constitutes a solution to the ℓ_∞-norm ASIS problem.

ℓ_2-Norm Attack: Variant 1. This attack hopes to optimize by balancing the components of target solution. Formally, let $c = \beta_2/\beta_1$ and define lattice

$$\Lambda = \left\{ \mathbf{x} = \begin{pmatrix} c\mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \in \mathbb{R}^{m_1+m_2} \;\middle|\; (\mathbf{I}_n \| \mathbf{A})\mathbf{v} = 0 \bmod q, \mathbf{v} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \in \mathbb{Z}^{m_1+m_2} \right\},$$

Then, the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} c\mathbf{I}_{m_1} & 0 \\ 0 & \mathbf{I}_{m_2} \end{pmatrix} \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A} \\ 0 & \mathbf{I}_{m_1+m_2-n} \end{pmatrix} \in \mathbb{R}^{(m_1+m_2)\times(m_1+m_2)}$$

constitute a basis of lattice Λ of dimension $d = m_1 + m_2$ and $\det(\Lambda) = c^{m_1} q^n$. Clearly, we obtain the solution $(\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{Z}^{m_1+m_2}$ of the ASIS problem, by finding out $\mathbf{x} = (c\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{R}^{m_1+m_2}$ with $\|\mathbf{x}\| \leq \beta_2$ in lattice Λ. Therefore, the complexity of this attack is essentaily that of running BKZ-b in search for $\mathbf{x}$ with $\|\mathbf{x}\| \leq \beta_2$.

Estimating the Complexity Cost of ℓ_2-Norm Attack. Let

$$\delta = ((\pi b)^{1/b} \cdot b/2\pi e)^{\frac{1}{2(b-1)}}$$

, and apply the BKZ-b to reduce the basis $\mathbf{B} \in \mathbb{Z}^{d\times d}$ of d-dimensional lattice to obtain a lattice vector $\mathbf{x}$ with $\|\mathbf{x}\| \approx \delta^d \det(\Lambda)^{1/d}$. Thus, the ℓ_2-Norm attack on ASIS is successful if we have

$$\delta^d \det(\Lambda)^{1/d} \leq \beta \ . \tag{9}$$

Same as the estimation for the ALWE problem, we use the complexity of solving SVP algorithm on b-dimensional lattice to (conservatively) estimate that of solving SVP on d-dimensional lattice (since the later will invoke the former several times). Likewise, we use $\text{cost}_b = 2^{0.292b}$ and $\text{cost}_b = 2^{0.265b}$ to distinguish between the complexities of classical and quantum algorithms respectively for solving SVP on b-dimensional lattice. Under this model, the complexity of ℓ_2-Norm attack is decided by the b satisfying (9).

D.2 Infinity-Norm Attack and Its Variants

Traditional ℓ_∞-Norm Attack. This attack is derived from the deterministic method in [19]. Formally, define

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^{m_1+m_2} | (\mathbf{I}_n \| \mathbf{A})\mathbf{x} = 0 \bmod q\}.$$

Then, column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A} \\ 0 & \mathbf{I}_{m_1+m_2} \end{pmatrix} \in \mathbb{Z}^{(m_1+m_2)\times(m_1+m_2)}$$

constitute a basis of lattice Λ of dimension $d = m_1 + m_2$ and $\det(\Lambda) = q^n$. Applying the BKZ-b lattice basis reduction algorithm to $\mathbf{B}$ yields a reduced basis $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$. Let $\hat{\mathbf{B}}^* = (\hat{\mathbf{b}}_1^*, \ldots, \mathbf{b}_d^*)$ be the orthogonalized matrix of $\hat{\mathbf{B}}$ by means of the Gram-Schmidt process, and $\ell_i = \log_2(\|\hat{\mathbf{b}}_i^*\|)$, then there exist integers $1 \le i \le j \le d$ such that

- For the first i vectors, we have $\ell_1 = \cdots = \ell_i = \log_2 q$;
- For the middle $j-i$ vectors, we have $\ell_{j'} = \log_2 q + s\cdot(j'-i)$ for each $j' \in \{i+1, \ldots, j\}$;
- For the last $d-j$ vectors, we have $\ell_{j+1} = \ell_{j+2} = \cdots = \ell_d = 0$.

where $s = \frac{1}{b-1}\log_2(\frac{b}{2\pi e}(\pi b)^{1/b}) < 0$ and $(j-i)(i+j+1)s = -2(n-i)\log_2 q$.

Although BKZ-b algorithm has no guarantee of reaching a valid solution to the ASIS problem, it takes as input $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$ and obtains $(\sqrt{4/3})^b$ vectors at the computational cost of solving the SVP problem on b-dimensional lattice[19], where the orthogonal projections of these vectors onto the space spanned by the first i vectors of $\hat{\mathbf{B}}^*$ have ℓ_2-norm of roughly $2^{\ell_{i+1}}$. By properly modelling the distribution of these $(\sqrt{4/3})^b$ vectors, we can estimate the probability of getting a solution of ASIS problem from these vectors. The complexity of BKZ-b divided by this success probability gives the overall complexity of this attack.

ℓ_∞-Norm Attack: Variant 1. This attack is derived from the randomized algorithm in [19]. Formally, define

$$\Lambda = \{\mathbf{x} \in \mathbb{Z}^{m_1+m_2} | (\mathbf{I}_n \| \mathbf{A})\mathbf{x} = 0 \bmod q\}.$$

Then, the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A} \\ 0 & \mathbf{I}_{m_1+m_2} \end{pmatrix} \in \mathbb{Z}^{(m_1+m_2)\times(m_1+m_2)}$$

constitute a basis of lattice Λ of dimension $d = m_1 + m_2$ and $\det(\Lambda) = q^n$. Randomize matrix $\mathbf{B}$ to get $\mathbf{B}'$ such that $\mathbf{B}'$ has no trivial vectors like $q\mathbf{e}_i$, where $\mathbf{e}_i$ is the i-th unit vector. Then, run BKZ-b on $\mathbf{B}'$ to get a reduced basis $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$. Let matrix $\hat{\mathbf{B}}^* = (\hat{\mathbf{b}}_1^*, \ldots, \mathbf{b}_d^*)$ be the Gram-Schmidt orthogonalized matrix of $\hat{\mathbf{B}}$ and $\ell_i = \log_2(\|\hat{\mathbf{b}}_i^*\|)$. We have that there exists integer $j \in \{1, 2, \cdots, d\}$ such that:

- For the first j vectors, we have $\ell_{j'} = -s\cdot(j-j'+1)$ for each $j' \in \{1, \ldots, j\}$;
- For the last $d-j$ vectors, we have $\ell_{j+1} = \ell_{j+2} = \cdots = \ell_d = 0$,

where $s = \frac{1}{b-1}\log_2(\frac{b}{2\pi e}(\pi b)^{1/b}) < 0$ and $j(j+1) = -\frac{2n\log_2 q}{s}$. Likewise, apply BKZ-$b$ to $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$ such that the resulting $(\sqrt{4/3})^b$ vectors have ℓ_2-norm roughly equal to 2^{ℓ_1} [19]. By properly modeling the distribution of these $(\sqrt{4/3})^b$ vectors, we can estimate the probability of finding a solution of the ASIS problem from these $(\sqrt{4/3})^b$ vectors. The overall complexity of the attack is the complexity of BKZ-b algorithm divided by this probability.

ℓ_∞-Norm Attack: Variant 2. This method intends to optimize the attack by balancing the components of the target vector. Formally, let $c = \beta_2/\beta_1$ and define lattice

$$\Lambda = \left\{ \mathbf{x} = \begin{pmatrix} c\mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \in \mathbb{R}^{m_1+m_2} \;\middle|\; (\mathbf{I}_n \| \mathbf{A})\mathbf{v} = 0 \bmod q, \mathbf{v} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \in \mathbb{Z}^{m_1+m_2} \right\}.$$

Then, the column vectors of matrix

$$\mathbf{B} = \begin{pmatrix} c\mathbf{I}_{m_1} & 0 \\ 0 & \mathbf{I}_{m_2} \end{pmatrix} \begin{pmatrix} q\mathbf{I}_n & -\mathbf{A} \\ 0 & \mathbf{I}_{m_1+m_2-n} \end{pmatrix} \in \mathbb{R}^{(m_1+m_2)\times(m_1+m_2)}$$

constitute a basis of lattice Λ of dimension $d = m_1 + m_2$ and $\det(\Lambda) = c^{m_1} q^n$. Clearly, we can get the solution $(\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{Z}^{m_1+m_2}$ to the ASIS problem by finding out $\mathbf{x} = (c\mathbf{x}_1^T, \mathbf{x}_2^T)^T \in \mathbb{R}^{m_1+m_2}$ with $\|\mathbf{x}\|_\infty \leq \beta_2$ on lattice Λ.

We further randomize matrix $\mathbf{B}$ to get $\mathbf{B}'$, where $\mathbf{B}'$ has no trivial vector such as $q\mathbf{e}_i$ and $\mathbf{e}_i$ is the i-th unit vector. Then run BKZ-b on $\mathbf{B}'$ to get a reduced basis $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$ an let $\hat{\mathbf{B}}^* = (\hat{\mathbf{b}}_1^*, \ldots, \mathbf{b}_d^*)$ be the Gram-Schmidt orthogonalized matrix of $\hat{\mathbf{B}}$. Denote $\ell_i = \log_2(\|\hat{\mathbf{b}}_i^*\|)$. We have that there exists $j \in \{1, \cdots, d\}$ such that:

- For the first j vectors, we have $\ell_{j'} = -s \cdot (j - j' + 1)$ for each $j' \in \{1, \ldots, j\}$;
- For the last $d - j$ vectors, we have $\ell_{j+1} = \ell_{j+2} = \cdots = \ell_d = 0$,

where $s = \frac{1}{b-1} \log_2(\frac{b}{2\pi e}(\pi b)^{1/b}) < 0$ and $j(j+1) = -\frac{2n \log_2 q}{s}$. Similary, by paying the cost of solving SVP problem on b-dimensional lattice, we obtain from $\hat{\mathbf{B}} \in \mathbb{Z}^{d\times d}$ a sequence of $(\sqrt{4/3})^b$ vectors with ℓ_2-norm roughly equal to 2^{ℓ_1} [19]. By reasonably modeling the distribution of the $(\sqrt{4/3})^b$ vectors, we can estimate the probability of finding a solution of ASIS problem from these vectors, and the overall complexity of this attack is that of BKZ-b divided by the successful probability. Finally, we add that the attack is also symmetric about β_1 and β_2, i.e., setting $c' = \beta_1/\beta_2$ is equivalent to that using $c = \beta_2/\beta_1$ under this model.

Estimating the Computational Cost of ℓ_∞-Norm Attack. Let integers $i < j \leq m_1 + m_2$ be the integers in the aforementioned infinity-norm attack and its variants (for variant 1 and 2, set $i = 0$). Following [19], we assume that the $(\sqrt{4/3})^b$ vectors follow the distribution below: the coefficients for the first i dimensions follow the uniform distribution on $\mathbb{Z}_q$, those for the $(i+1)$-th to j-th dimension follow the Gaussian distribution with standard variance $2^{\ell_{i+1}}/\sqrt{j-i}$, and the last $d - j$ one are zero. Clearly, the probability that any of the first i coefficients is less than β_1 is relatively small (approximately β_1/q). To increase the success probability of traditional infinity-norm attack and variant 1, we further assume $\beta_1 > \beta_2$ (namely, we assume that there exist efficient algorithms that obtain vectors whose leading coefficients are less-than-$\max(\beta_1, \beta_2)$ values, although we do not know whether the assumption is realistic or not, making such an assumption can only make the estimation conservative). Under this model, we compute the probabilities of the three algorithms in finding a vector $\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T)^T$ satisfy $\|\mathbf{x}_1\|_\infty \leq \beta_1$ and $\|\mathbf{x}_2\|_\infty \leq \beta_2$ from $(\sqrt{4/3})^b$ vectors. Further, we use the complexity of solving SVP on b-dimensional sublattice as the estimate for that of solving SVP on d-dimensional lattice, which is conservative (since the latter invoke the former as a sub-routine many times). Under this conservative model, the overall complexity of ℓ_∞-norm attack is cost_b/p.