

Practical Key Establishment from Module Lattice*

Abstract

Lattice-based cryptography is one of the promising mathematical approaches to achieving security resistant to quantum attacks. Module lattice is between general lattice and ideal lattice, and is desirable for its flexibility, versatility and competitive performance in designing cryptosystems. In this work, we present practical key establishment protocols based on the module learning-with-errors (MLWE) assumption. In particular, as a key building tool we present symmetric key consensus from noise (SKCN) that is designed to optimize performance for both key exchange and digital signature. Finally, we made efforts to test a large set of parameters in order to choose the concrete parameters with different goals or trade-offs among security, efficiency and failure rate.

1 Introduction

Most public-key cryptosystems currently in use, based on the hardness of solving (elliptic curve) discrete logarithm or factoring large integers, will be broken, if large-scale quantum computers are ever built. The arrival of such quantum computers is now believed by many scientists to be merely a significant engineering challenge, and is estimated to be within the next two decades or so. Historically, it has taken almost two decades to deploy the modern public key cryptography infrastructure. Therefore, regardless of whether we can estimate the exact time of the arrival of the quantum computing era, we should begin now to prepare our information security systems to be able to resist quantum computing. In addition, for the content we want to protect over a period of 15 years or longer, it becomes necessary to switch to post-quantum cryptography today. In the majority of contexts, *ephemeral* key establishment (KE), which plays a central role in modern cryptography, is among the most critical asymmetric primitives to upgrade to post-quantum security.

Lattice-based cryptography is one of the promising mathematical approaches to achieving security resistant to quantum attacks. For cryptographic usage, compared with the classic hard lattice problems such as SVP and CVP, the learning with errors (LWE) problem is proven to be much more versatile [Reg09]. Nevertheless, LWE-based cryptosystems are usually less efficient, which was then resolved by the introduction of the ring-LWE (RLWE) problem [LPR10] and the module-LWE (MLWE) [LS15]. In recent years, large numbers of impressive works are developed from LWE and its variant, with (ephemeral) key exchange and public-key encryption being the study focus of this work [DXL12, Pei14, BCNS15, ADPS16, BCD⁺16, Reg09, GPV08, LP10, LPR10, LPR13, PG13, BDK⁺17, NIST]. Module lattice is between general lattice and ideal lattice, and is desirable for its flexibility, versatility and competitive performance in designing cryptosystems.

*Preliminary version of this work, particularly SKCN and AKCN, appeared at arXiv: <https://arxiv.org/abs/1611.06150>.

In this work, we focus on the design and analysis of key establishment based on the MLWE problem.

One of the main technical contributions in recent works on achieving practical key establishment based on LWE and its variants (e.g., [ADPS16, BCD⁺16, PG13]) is the improvement and generalization of the key reconciliation mechanisms [DXL12, Pei14, LP10, NIST]. But the key reconciliation mechanisms were only previously used and analyzed, for both KE and PKE, in a *non-black-box* way. This means, for new key reconciliation mechanisms developed in the future to be used for constructing lattice-based cryptosystems, we need to analyze their security from scratch. Moreover, for the various parameters involved in key reconciliation, the bounds on what could or couldn't be achieved are unclear. As a consequence, we lack basic criteria to evaluate various reconciliation mechanisms and to indicate whether they can be further improved.

Abstraction/generalization is fundamental to natural science (mathematics, physics), and is particularly important to cryptography. For example, in the area of signature, Schnorr signature is generalized via Fiat-Shamir transformation [FS86], with abstraction of Σ -protocol [CDS94]. The similar abstraction and generalization also plays a fundamental role in CCA-secure PKE, and in many more areas of modern cryptography. Abstraction and generalization is particularly helpful and expected for lattice-based cryptography, as they are usually less easy to understand and evaluate, and are related to the ongoing NIST post-quantum cryptography standardization [NIST].

In this work, we abstract the key ingredients in previous key exchange and PKE schemes based on LWE and its variants, by introducing and formalizing the building tool, referred to as key consensus (KC) and its asymmetric variant AKC. KC and AKC allow two communicating parties to reach consensus from close values obtained by some secure information exchange, such as exchanging their LWE samples. We then discover upper bounds on parameters for any KC and AKC. As a conceptual contribution, this simplifies the design and analysis of these cryptosystems in the future. We then design and analyze both generic and highly practical KC and AKC schemes, which are referred to as *symmetric key consensus with noise* (SKCN) and *asymmetric key consensus with noise* (AKCN) respectively for presentation simplicity. Both SKCN and AKCN can be instantiated to tightly match the proved upper bounds. In particular, we show in [CGZ18] that a determined version of SKCN serves also as the key building tool for constructing lattice-based signature schemes. Then, based on SKCN and AKCN we present practical key establishment protocols from the MLWE problem. Finally, we made efforts to test a large set of parameters in order to choose the concrete parameters with different goals or trade-offs among security, efficiency and failure rate.

2 Preliminaries

A string or value α means a binary one, and $|\alpha|$ is its binary length. For any real number x , $\lfloor x \rfloor$ denotes the largest integer that less than or equal to x , and $\lfloor x \rfloor = \lfloor x + 1/2 \rfloor$. For any positive integers a and b , denote by $\text{lcm}(a, b)$ the least common multiple of them. For any $i, j \in \mathbb{Z}$ such that $i < j$, denote by $[i, j]$ the set of integers $\{i, i+1, \dots, j-1, j\}$. For any positive integer t , we let \mathbb{Z}_t denote $\mathbb{Z}/t\mathbb{Z}$. The elements of \mathbb{Z}_t are represented, by default, as $[0, t-1]$. Nevertheless, sometimes, \mathbb{Z}_t is explicitly specified to be represented as $[-\lfloor (t-1)/2 \rfloor, \lfloor t/2 \rfloor]$.

If S is a finite set then $|S|$ is its cardinality, and $x \leftarrow S$ is the operation of picking an element uniformly at random from S . For two sets $A, B \subseteq \mathbb{Z}_q$, define $A + B \triangleq \{a + b \mid a \in A, b \in B\}$. For an additive group $(G, +)$, an element $x \in G$ and a subset $S \subseteq G$, denote by $x + S$ the set

containing $x + s$ for all $s \in S$. For a set S , denote by $\mathcal{U}(S)$ the uniform distribution over S . For any discrete random variable X over \mathbb{R} , denote $\text{Supp}(X) = \{x \in \mathbb{R} \mid \Pr[X = x] > 0\}$.

We use standard notations and conventions below for writing probabilistic algorithms, experiments and interactive protocols. If \mathcal{D} denotes a probability distribution, $x \leftarrow \mathcal{D}$ is the operation of picking an element according to \mathcal{D} . If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement. If A is a probabilistic algorithm, then $A(x_1, x_2, \dots; r)$ is the result of running A on inputs x_1, x_2, \dots and coins r . We let $y \leftarrow A(x_1, x_2, \dots)$ denote the experiment of picking r at random and letting y be $A(x_1, x_2, \dots; r)$. By $\Pr[R_1; \dots; R_n : E]$ we denote the probability of event E , after the ordered execution of random processes R_1, \dots, R_n .

2.1 The LWE and LWR Problems

Given positive *continuous* $\alpha > 0$, define the real Gaussian function $\rho_\alpha(x) \triangleq \exp(-x^2/2\alpha^2)/\sqrt{2\pi\alpha^2}$ for $x \in \mathbb{R}$. Let $D_{\mathbb{Z},\alpha}$ denote the one-dimensional *discrete* Gaussian distribution over \mathbb{Z} , which is determined by its probability density function $D_{\mathbb{Z},\alpha}(x) \triangleq \rho_\alpha(x)/\rho_\alpha(\mathbb{Z}), x \in \mathbb{Z}$. Finally, let $D_{\mathbb{Z}^n,\alpha}$ denote the n -dimensional *spherical* discrete Gaussian distribution over \mathbb{Z}^n , where each coordinate is drawn *independently* from $D_{\mathbb{Z},\alpha}$.

Given positive integers n and q that are both polynomials in the security parameter λ , an integer vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution χ on \mathbb{Z}_q , let $A_{q,\mathbf{s},\chi}$ be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, and an error term $e \leftarrow \chi$, and outputting the pair $(\mathbf{a}, b = \mathbf{a}^T \mathbf{s} + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. The error distribution χ is typically taken to be the discrete Gaussian probability distribution $D_{\mathbb{Z},\alpha}$ defined previously; However, as suggested in [BCD⁺16], other alternative distributions of χ can be taken. Briefly speaking, the (decisional) *learning with errors* (LWE) assumption [Reg09] says that, for sufficiently large security parameter λ , no probabilistic polynomial-time (PT) algorithm can distinguish, with non-negligible probability, $A_{q,\mathbf{s},\chi}$ from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. This holds even if \mathcal{A} sees polynomially many samples, and even if the secret vector \mathbf{s} is drawn randomly from χ^n [ACPS09].

The LWR problem [BPR12] is a “decarbonized” variant of the LWE problem. Let \mathcal{D} be some distribution over \mathbb{Z}_q^n , and $\mathbf{s} \leftarrow \mathcal{D}$. For integers $q \geq p \geq 2$ and any $x \in \mathbb{Z}_q$, denote

$$\lfloor x \rfloor_p = \lfloor \frac{p}{q} x \rfloor. \quad (1)$$

Then, for positive integers n and $q \geq p \geq 2$, the LWR distribution $A_{n,q,p}(\mathbf{s})$ over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ is obtained by sampling \mathbf{a} from \mathbb{Z}_q^n uniformly at random, and outputting $(\mathbf{a}, \lfloor \mathbf{a}^T \mathbf{s} \rfloor_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p$. Briefly speaking, the (decisional) LWR assumption says that, for sufficiently large security parameter, no PT algorithm \mathcal{A} can distinguish, with non-negligible probability, the distribution $A_{n,q,p}(\mathbf{s})$ from the distribution $(\mathbf{a} \leftarrow \mathbb{Z}_q^n, \lfloor u \rfloor_p)$ where $u \leftarrow \mathbb{Z}_q$. This holds even if \mathcal{A} sees polynomially many samples. An efficient reduction from the LWE problem to the LWR problem, for super-polynomial large q , is provided in [BPR12]. Let B denote the bound for any component in the secret \mathbf{s} . It is recently shown that, when $q \geq 2mBp$ (equivalently, $m \leq q/2Bp$), the LWE problem can be reduced to the (decisional) LWR assumption with m independently random samples [BGM⁺16]. Moreover, the reduction from LWE to LWR is actually independent of the distribution of the secret \mathbf{s} .

2.2 Ring Choices

For positive integer m , let $\Phi_m(x) \in \mathbb{Z}[x]$ denote the m -th cyclotomic polynomial. Let $\varphi(\cdot)$ denote the Euler's phi function. Below, we explicitly list three kinds of rings.

- **Power-of-Two:** Let n be a power of two and let q be a prime such that $q \equiv 1 \pmod{2n}$. Let $m = 2n$, thus $n = \varphi(m)$ and $q \equiv 1 \pmod{m}$. In this case, $\Phi_m(X) = X^n + 1$. Define the rings as

$$\mathcal{R} = \mathbb{Z}[X]/(X^n + 1), \quad \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1).$$

This is a special case of ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$.

- **Safe-Prime-1:** Let m be a safe prime such that $m = 2m' + 1$ for another prime m' . Let e denote the smallest integer such that $2^e > 2m$ and let q be a prime such that $q \equiv 1 \pmod{2^e \cdot m}$. In this case, $\Phi_m(X) = X^{m-1} + X^{m-2} + \dots + 1 = X^n + X^{n-1} + \dots + 1$, where $n = m - 1 = 2m'$. Define the rings as

$$\mathcal{R} = \mathbb{Z}[X]/(X^n + X^{n-1} + \dots + 1), \quad \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + X^{n-1} + \dots + 1).$$

- **Safe-Prime-2:** Let $n + 1$ be a safe prime. In this case, $\Phi_{n+1}(X) = X^n + X^{n-1} + \dots + 1$. Modulus $q = \text{poly}(\lambda)$ is power of 2 or q is a positive prime such that $q \equiv 1 \pmod{n + 1}$. Define the rings as

$$\mathcal{R} = \mathbb{Z}[X]/(X^n + X^{n-1} + \dots + 1), \quad \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + X^{n-1} + \dots + 1).$$

In this work, we identify the polynomial $a = \sum_{i=0}^{n-1} a_i x^i \in \mathcal{R}$ (*resp.* \mathcal{R}_q) with the vector $\mathbf{a} = (a_0, a_1, \dots, a_{n-1}) \in \mathbb{Z}^n$ (*resp.* \mathbb{Z}_q^n). Therefore, for a polynomial $a \in \mathcal{R}$ (*resp.* \mathcal{R}_q) and any distribution \mathcal{D} over \mathbb{Z}^n (*resp.* \mathbb{Z}_q^n), $a \leftarrow \mathcal{D}$ means sampling the vector \mathbf{a} following the distribution \mathcal{D} and then identifying the vector with its corresponding polynomial a . Similarly, a k -dimensional vector of polynomials $\mathbf{a} \in \mathcal{R}^k$ (*resp.* \mathcal{R}_q^k) can be generated according to the distribution \mathcal{D}^k .

2.3 The MLWE Problem

Let $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ be any of the above three kinds of rings. In this work, for the sake of faster implementation, we use the ring of power-of-two in the actual implementation, where $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$, and $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$.

Let l be a positive integer parameter. Let S_η denote a distribution on all elements $w \in \mathcal{R}$ such that $\|w\|_\infty \leq \eta$.¹ The Module-LWE (MLWE) problem is introduced in [LS15], which is a generalization of the RLWE problem. We make use of the definitions described in [BDK⁺17].

- **MLWE distribution.** The MLWE distribution is defined on $\mathcal{R}_q^l \times \mathcal{R}_q$ induced by pairs $(\mathbf{a}_i, \mathbf{b}_i)$, where $\mathbf{a}_i \leftarrow \mathcal{R}_q^l$ is uniform and $\mathbf{b} = \mathbf{a}_i^T \mathbf{s} + \mathbf{e}_i$ with $\mathbf{s} \leftarrow S_\eta^l$ common to all samples and $\mathbf{e}_i \leftarrow S_\eta$ fresh for every sample.

¹A typical instantiation of S_η , as proposed in [BDK⁺17], is based on the following centered binomial distribution: sample $(a_1, \dots, a_\eta, b_1, \dots, b_\eta) \leftarrow \{0, 1\}^{2\eta}$, and output $\sum_{i=1}^\eta (a_i - b_i)$.

- **MLWE assumption.** The MLWE problem consists in recovering \mathbf{s} from polynomially many samples chosen from the MLWE distribution. More precisely, for an algorithm A , we define

$$\text{Adv}_{h,l,\eta}^{\text{mlwe}}(A) = \Pr \left[\mathbf{x} = \mathbf{s} : \begin{array}{l} \mathbf{A} \leftarrow \mathcal{R}_q^{h \times l}; (\mathbf{s}, \mathbf{e}) \leftarrow S_\eta^l \times S_\eta^h; \\ b \leftarrow \mathbf{A}\mathbf{s} + \mathbf{e}; \mathbf{x} \leftarrow A(\mathbf{A}, \mathbf{b}); \end{array} \right].$$

We say that the (t, ε) $\text{MLWE}_{h,l,\eta}$ hardness assumption holds if no algorithm A running in time at most t has an advantage greater than ε .

3 Key Consensus with Noise

Before presenting the definition of key consensus (KC) scheme, we first introduce a new function $|\cdot|_t$ relative to arbitrary positive integer $t \geq 1$: $|x|_t = \min\{x \bmod t, t - x \bmod t\}$, $\forall x \in \mathbb{Z}$, where the result of modular operation is represented in $\{0, \dots, (t-1)\}$. For instance, $|-1|_t = \min\{-1 \bmod t, (t+1) \bmod t\} = \min\{t-1, 1\} = 1$. In the following description, we use $|\sigma_1 - \sigma_2|_q$ to measure the distance between two elements $\sigma_1, \sigma_2 \in \mathbb{Z}_q$.

Definition 3.1. A KC scheme $KC = (\text{params}, \text{Con}, \text{Rec})$ is specified as follows.

- $\text{params} = (q, m, g, d, \text{aux})$ denotes the system parameters, where q, m, g, d are positive integers satisfying $2 \leq m, g \leq q, 0 \leq d \leq \lfloor \frac{q}{2} \rfloor$, and aux denotes some auxiliary values that are usually determined by (q, m, g, d) and could be set to be a special symbol \emptyset indicating “empty”.
- $(k_1, v) \leftarrow \text{Con}(\sigma_1, \text{params})$: On input of $(\sigma_1 \in \mathbb{Z}_q, \text{params})$, the probabilistic polynomial-time conciliation algorithm Con outputs (k_1, v) , where $k_1 \in \mathbb{Z}_m$ is the shared-key, and $v \in \mathbb{Z}_g$ is a hint signal that will be publicly delivered to the communicating peer to help the two parties reach consensus.
- $k_2 \leftarrow \text{Rec}(\sigma_2, v, \text{params})$: On input of $(\sigma_2 \in \mathbb{Z}_q, v, \text{params})$, the deterministic polynomial-time reconciliation algorithm Rec outputs $k_2 \in \mathbb{Z}_m$.

Correctness: A KC scheme is correct, if it holds $k_1 = k_2$ for any $\sigma_1, \sigma_2 \in \mathbb{Z}_q$ such that $|\sigma_1 - \sigma_2|_q \leq d$.

Security: A KC scheme is secure, if k_1 and v are independent, and k_1 is uniformly distributed over \mathbb{Z}_m , whenever $\sigma_1 \leftarrow \mathbb{Z}_q$. The probability is taken over the sampling of σ_1 and the random coins used by Con .

3.1 Efficiency Upper Bound of KC

The following theorem reveals an upper bound on the parameters q (dominating security and efficiency), m (parameterizing range of consensus key), g (parameterizing bandwidth), and d (parameterizing error rate), which allows us to take balance on these parameters according to different priorities. Due to space limitation, the proof is presented in Appendix A.

Theorem 3.1. If $KC = (\text{params}, \text{Con}, \text{Rec})$ is a correct and secure key consensus scheme, and $\text{params} = (q, m, g, d, \text{aux})$, then $2md \leq q \left(1 - \frac{1}{g}\right)$.

Algorithm 1 SKCN: Symmetric and Rounded KC with Noise

```
1: params =  $(q, m, g, d, aux)$ ,  $aux = \{q' = \text{lcm}(q, m), \alpha = q'/q, \beta = q'/m\}$ 
2: procedure CON( $(\sigma_1, \text{params})$ )  $\triangleright \sigma_1 \in [0, q-1]$ 
3:    $e \leftarrow [-\lfloor(\alpha-1)/2\rfloor, \lfloor\alpha/2\rfloor]$ 
4:    $\sigma_A = (\alpha\sigma_1 + e) \bmod q'$ 
5:    $k_1 = \lfloor\sigma_A/\beta\rfloor \bmod m \in \mathbb{Z}_m$ 
6:    $v' = \sigma_A \bmod \pm\beta$ 
7:    $v = \lfloor v'g/\beta\rfloor$   $\triangleright v \in \mathbb{Z}_g/\mathbb{Z}_{g+1}$ 
8:   return  $(k_1, v)$ 
9: end procedure
10: procedure REC( $\sigma_2, v, \text{params}$ )  $\triangleright \sigma_2 \in [0, q-1]$ 
11:    $k_2 = \lfloor\alpha\sigma_2/\beta - v/g\rfloor \bmod m$ 
12:   return  $k_2$ 
13: end procedure
```

3.2 Construction and Analysis of SKCN

Let $r \bmod \alpha$ denote the unique integer $r' \in [0, \alpha-1]$ such that $\alpha \mid (r' - r)$, and let $r \bmod \pm\alpha$ denote the unique integer $r'' \in [-\lfloor\frac{\alpha-1}{2}\rfloor, \lfloor\frac{\alpha}{2}\rfloor]$ such that $\alpha \mid (r'' - r)$. For a positive integer t and an element $x \in \mathbb{Z}_t$, define $|x|_t \triangleq |x \bmod \pm t|$.

The key consensus scheme, named SKCN, is presented in Algorithm 1.

Fact 3.1. *For any $x, y, t, l \in \mathbb{Z}$ where $t \geq 1$ and $l \geq 0$, if $|x - y|_t \leq l$, then there exists $\theta \in \mathbb{Z}$ and $\delta \in [-l, l]$ such that $x = y + \theta t + \delta$.*

Theorem 3.2. *Suppose that the system parameters satisfy $(2d+1)m < q \left(1 - \frac{1}{g}\right)$ where $m \geq 2$ and $g \geq 2$. Then, the SKCN scheme is correct.*

Proof. Suppose $|\sigma_1 - \sigma_2|_q \leq d$. By Fact 3.1, there exist $\theta \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. From line 4 and 6 in Algorithm 1, we know that there is a $\theta' \in \mathbb{Z}$, such that $\alpha\sigma_1 + e + \theta'q' = \sigma_A = k_1\beta + v'$. And from the definition of α, β , we have $\alpha/\beta = m/q$. Taking these into the formula of k_2 in Rec (line 11 in Algorithm 1), we have

$$k_2 = \lfloor\alpha\sigma_2/\beta - v/g\rfloor \bmod m \tag{2}$$

$$= \lfloor\alpha(\theta q + \sigma_1 + \delta)/\beta - v/g\rfloor \bmod m \tag{3}$$

$$= \left\lfloor m(\theta - \theta') + \frac{1}{\beta}(k_1\beta + v' - e) + \frac{\alpha\delta}{\beta} - \frac{v}{g} \right\rfloor \bmod m \tag{4}$$

$$= \left\lfloor k_1 + \left(\frac{v'}{\beta} - \frac{v}{g}\right) - \frac{e}{\beta} + \frac{\alpha\delta}{\beta} \right\rfloor \bmod m \tag{5}$$

Notice that $|v'/\beta - v/g| = |v'g - \beta v|/\beta g \leq 1/2g$. So

$$\left| \left(\frac{v'}{\beta} - \frac{v}{g}\right) - \frac{e}{\beta} + \frac{\alpha\delta}{\beta} \right| \leq \frac{1}{2g} + \frac{\alpha}{\beta}(d + 1/2).$$

From the assumed condition $(2d+1)m < q(1 - \frac{1}{g})$, we get that the right-hand side is strictly smaller than $1/2$; Consequently, after the rounding, $k_2 = k_1$. \square

Algorithm 2 SKCN simple

```
1: params :  $q = 2^{\bar{q}}, g = 2^{\bar{g}}, m = 2^{\bar{m}}, d$ , where  $\bar{g} + \bar{m} = \bar{q}$ 
2: procedure CON( $\sigma_1$ , params)
3:    $k_1 = \lfloor \frac{\sigma_1}{g} \rfloor$ 
4:    $v = \sigma_1 \bmod g$ 
5:   return ( $k_1, v$ )
6: end procedure
7: procedure REC( $\sigma_2, v$ , params)
8:    $k_2 = \lfloor \frac{\sigma_2 - v}{g} \rfloor \bmod m$ 
9:   return  $k_2$ 
10: end procedure
```

Theorem 3.3. *SKCN is secure. Specifically, when $\sigma_1 \leftarrow \mathbb{Z}_q$, k_1 and v are independent, and k_1 is uniform over \mathbb{Z}_m , where the probability is taken over the sampling of σ_1 and the random coins used by Con.*

Proof. Recall that $q' = \text{lcm}(q, m)$, $\alpha = q'/q$, $\beta = q'/m$. We first demonstrate that σ_A is subject to uniform distribution over $\mathbb{Z}_{q'}$. Consider the map $f : \mathbb{Z}_q \times \mathbb{Z}_\alpha \rightarrow \mathbb{Z}_{q'}$; $f(\sigma, e) = (\alpha\sigma + e) \bmod q'$, where the elements in \mathbb{Z}_q and \mathbb{Z}_α are represented in the same way as specified in Algorithm 1. It is easy to check that f is an one-to-one map. Since $\sigma_1 \leftarrow \mathbb{Z}_q$ and $e \leftarrow \mathbb{Z}_\alpha$ are subject to uniform distributions, and they are independent, $\sigma_A = (\alpha\sigma_1 + e) \bmod q' = f(\sigma_1, e)$ is also subject to uniform distribution over $\mathbb{Z}_{q'}$.

In the similar way, defining $f' : \mathbb{Z}_m \times \mathbb{Z}_\beta \rightarrow \mathbb{Z}_{q'}$ such that $f'(k_1, v') = \beta k_1 + v'$, then f' is obviously a one-to-one map. From line 6 of Algorithm 1, $f'(k_1, v') = \sigma_A$. As σ_A is distributed uniformly over $\mathbb{Z}_{q'}$, (k_1, v') is uniformly distributed over $\mathbb{Z}_m \times \mathbb{Z}_\beta$, and so k_1 and v' are independent. As v only depends on v' , k_1 and v are independent. \square

Corollary 3.1. *If m, g are power of 2, $q = m \cdot g$, and $2md < q$, then the KC scheme described in Algorithm 2 is correct and secure. Notice that the constraint on parameters is further simplified to $2md < q$ in this case.*

Proof. For correctness, supposing $|\sigma_1 - \sigma_2|_q \leq d$, by Fact 3.1, there exist $\theta \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. Taking this into line 8 of Algorithm 2, i.e., the formula computing k_2 , we have

$$\begin{aligned} k_2 &= \lfloor (\sigma_1 - v + \theta q + \delta)/g \rfloor \bmod m \\ &= (k_1 + \theta m + \lfloor \delta/g \rfloor) \bmod m. \end{aligned}$$

If $2md < q$, then $|\delta/g| \leq d/g < 1/2$, so that $k_2 = k_1 \bmod m = k_1$.

For security, as a special case of generic scheme described in Algorithm 1, the security of Algorithm 2 follows directly from that of Algorithm 1. \square \square

4 Asymmetric Key Consensus with Noise

Definition 4.1. *An asymmetric key consensus scheme $AKC = (\text{params}, \text{Con}, \text{Rec})$ is specified as follows:*

- $\text{params} = (q, m, g, d, \text{aux})$ denotes the system parameters, where $q, 2 \leq m, g \leq q, 1 \leq d \leq \lfloor \frac{q}{2} \rfloor$ are positive integers, and aux denotes some auxiliary values that are usually determined by (q, m, g, d) and could be set to be empty.
- $v \leftarrow \text{Con}(\sigma_1, k_1, \text{params})$: On input of $(\sigma_1 \in \mathbb{Z}_q, k_1 \in \mathbb{Z}_m, \text{params})$, the probabilistic polynomial-time conciliation algorithm Con outputs the public hint signal $v \in \mathbb{Z}_g$.
- $k_2 \leftarrow \text{Rec}(\sigma_2, v, \text{params})$: On input of $(\sigma_2, v, \text{params})$, the deterministic polynomial-time algorithm Rec outputs $k_2 \in \mathbb{Z}_m$.

Correctness: An AKC scheme is correct, if it holds $k_1 = k_2$ for any $\sigma_1, \sigma_2 \in \mathbb{Z}_q$ such that $|\sigma_1 - \sigma_2|_q \leq d$.

Security: An AKC scheme is secure, if v is independent of k_1 whenever σ_1 is uniformly distributed over \mathbb{Z}_q . Specifically, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, it holds that $\Pr[v = \tilde{v} | k_1 = \tilde{k}_1] = \Pr[v = \tilde{v} | k_1 = \tilde{k}'_1]$, where the probability is taken over $\sigma_1 \leftarrow \mathbb{Z}_q$ and the random coins used by Con .

Theorem 4.1. Let AKC be an asymmetric key consensus scheme with $\text{params} = (q, m, d, g, \text{aux})$. If AKC is correct and secure, then $2md \leq q \left(1 - \frac{m}{g}\right)$.

The proof of Theorem 4.1 is given in Appendix B. Comparing the formula $2md \leq q(1 - m/g)$ in Theorem 4.1 with the formula $2md \leq q(1 - 1/g)$ in Theorem 3.1, we see that the only difference is a factor m in g . This indicates that, on the same values of (q, m, d) , an AKC scheme has to use a bigger bandwidth parameter g compared to KC. Some discussions on the relationship between KC/AKC and fuzzy extractor [DORS08] are presented in Appendix C.

4.1 Construction and Analysis of AKCN

The AKCN scheme, referred to as *asymmetric key consensus with noise*, is depicted in Algorithm 3. For AKCN, we can offline compute and store k_1 and $g \lfloor k_1 q / m \rfloor$ in order to accelerate online performance.

Algorithm 3 AKCN: Asymmetric KC with Noise

```

1:  $\text{params} = (q, m, g, d, \text{aux})$ , where  $\text{aux} = \emptyset$ .
2: procedure  $\text{CON}(\sigma_1, k_1, \text{params})$   $\triangleright \sigma_1 \in [0, q - 1]$ 
3:    $v = \lfloor g(\sigma_1 + \lfloor k_1 q / m \rfloor) / q \rfloor \bmod g$ 
4:   return  $v$ 
5: end procedure
6: procedure  $\text{REC}(\sigma_2, v, \text{params})$   $\triangleright \sigma_2 \in [0, q - 1]$ 
7:    $k_2 = \lfloor m(v/g - \sigma_2/q) \rfloor \bmod m$ 
8:   return  $k_2$ 
9: end procedure

```

The design of AKCN was guided by, and motivated for, the upper-bound for AKC proved in this work. In designing AKCN, we combine all the existing optimizations in the literature in order to almost meet the upperbound proved in Theorem 4.1. AKCN is a generalization of the basic reconciliation mechanisms proposed in [LPR10, LP10], and its design was also inspired

by the design of our SKCN and the works [BPR12, PG13].² In particular, the reconciliation mechanisms proposed in [LPR10, LP10] correspond to the special case of AKCN when $g = q$ and $m = 2$. Note that, with AKCN, we use Equation 1 described in the definition of LWR [BPR12], which may also be derived implicitly from [Pei09].

Briefly speaking, the novelty of AKCN lies in two aspects: (1) the combination of the basic reconciliation mechanism from [LPR10, LP10] and the rounding technique from [BPR12, Pei09] in the **Con** procedure is first explicitly presented in this work, where the **Rec** procedure is less straightforward in this case; (2) multi-bit reconciliation with generalized m . To the best of our knowledge, we are unaware of any existing single work from which AKCN is directly instantiated.

Theorem 4.2. *Suppose the parameters of AKCN satisfy $(2d + 1)m < q \left(1 - \frac{m}{g}\right)$. Then, the AKCN scheme described in Algorithm 3 is correct.*

Proof. From the formula generating v , we know that there exist $\varepsilon_1, \varepsilon_2 \in \mathbb{R}$ and $\theta \in \mathbb{Z}$, where $|\varepsilon_1| \leq 1/2$ and $|\varepsilon_2| \leq 1/2$, such that

$$v = \frac{g}{q} \left(\sigma_1 + \left(\frac{k_1 q}{m} + \varepsilon_1 \right) \right) + \varepsilon_2 + \theta g$$

Taking this into the formula computing k_2 in **Rec**, we have

$$\begin{aligned} k_2 &= \lfloor m(v/g - \sigma_2/q) \rfloor \bmod m \\ &= \left\lfloor m \left(\frac{1}{q} (\sigma_1 + k_1 q/m + \varepsilon_1) + \frac{\varepsilon_2}{g} + \theta - \frac{\sigma_2}{q} \right) \right\rfloor \bmod m \\ &= \left\lfloor k_1 + \frac{m}{q} (\sigma_1 - \sigma_2) + \frac{m}{q} \varepsilon_1 + \frac{m}{g} \varepsilon_2 \right\rfloor \bmod m \end{aligned}$$

By Fact 3.1 (page 6), there exist $\theta' \in \mathbb{Z}$ and $\delta \in [-d, d]$ such that $\sigma_1 = \sigma_2 + \theta' q + \delta$. Hence,

$$k_2 = \left\lfloor k_1 + \frac{m}{q} \delta + \frac{m}{q} \varepsilon_1 + \frac{m}{g} \varepsilon_2 \right\rfloor \bmod m$$

Since $|m\delta/q + m\varepsilon_1/q + m\varepsilon_2/g| \leq md/q + m/2q + m/2g < 1/2$, $k_1 = k_2$. □ □

Theorem 4.3. *The AKCN scheme is secure. Specifically, v is independent of k_1 when $\sigma_1 \leftarrow \mathbb{Z}_q$.*

Proof. For arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, we prove that $\Pr[v = \tilde{v} | k_1 = \tilde{k}_1] = \Pr[v = \tilde{v} | k_1 = \tilde{k}'_1]$ when $\sigma_1 \leftarrow \mathbb{Z}_q$.

For any (\tilde{k}, \tilde{v}) in $\mathbb{Z}_m \times \mathbb{Z}_g$, the event $(v = \tilde{v} \mid k_1 = \tilde{k})$ is equivalent to the event that there exists $\sigma_1 \in \mathbb{Z}_q$ such that $\tilde{v} = \lfloor g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q \rfloor \bmod g$. Note that $\sigma_1 \in \mathbb{Z}_q$ satisfies $\tilde{v} = \lfloor g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q \rfloor \bmod g$, if and only if there exist $\varepsilon \in (-1/2, 1/2]$ and $\theta \in \mathbb{Z}$ such that $\tilde{v} = g(\sigma_1 + \lfloor \tilde{k}q/m \rfloor)/q + \varepsilon - \theta g$. That is, $\sigma_1 = (q(\tilde{v} - \varepsilon)/g - \lfloor \tilde{k}q/m \rfloor) \bmod q$, for some $\varepsilon \in (-1/2, 1/2]$. Let $\Sigma(\tilde{v}, \tilde{k}) = \{\sigma_1 \in \mathbb{Z}_q \mid \exists \varepsilon \in (-1/2, 1/2] \text{ s.t. } \sigma_1 = (q(\tilde{v} - \varepsilon)/g - \lfloor \tilde{k}q/m \rfloor) \bmod q\}$. Defining the map $\phi : \Sigma(\tilde{v}, 0) \rightarrow \Sigma(\tilde{v}, \tilde{k})$, by setting $\phi(x) = (x - \lfloor \tilde{k}q/m \rfloor) \bmod q$. Then ϕ is obviously a one-to-one map. Hence, the cardinality of $\Sigma(\tilde{v}, \tilde{k})$ is irrelevant to \tilde{k} . Specifically, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k}_1, \tilde{k}'_1 \in \mathbb{Z}_m$, it holds that $|\Sigma(\tilde{v}, \tilde{k}_1)| = |\Sigma(\tilde{v}, \tilde{k}'_1)| = |\Sigma(\tilde{v}, 0)|$

²But AKCN and the underlying reconciliation mechanism of [PG13] could be viewed as incomparable in general.

Algorithm 4 AKCN simple

```
1: params =  $(q, m, g, d, aux)$ , where  $q = 2^{\bar{q}}$ ,  $g = 2^{\bar{g}}$ ,  $m = 2^{\bar{m}}$ , and  $q = gm$  (i.e.,  $\bar{g} + \bar{m} = \bar{q}$ )  
2: procedure CON( $\sigma_1, k_1, \text{params}$ )  $\triangleright \sigma_1 \in [0, q - 1]$   
3:    $v = \lfloor (k_1 g + \sigma_1) / m \rfloor \bmod g$   $\triangleright k_1 g / m$  can be offline computed  
4:   return  $v$   
5: end procedure  
6: procedure REC( $\sigma_2, v, \text{params}$ )  $\triangleright \sigma_2 \in [0, q - 1]$   
7:    $k_2 = \lfloor (mv - \sigma_2) / g \rfloor \bmod m$   
8:   return  $k_2$   
9: end procedure
```

Now, for arbitrary $\tilde{v} \in \mathbb{Z}_g$ and arbitrary $\tilde{k} \in \mathbb{Z}_m$, when $\sigma_1 \leftarrow \mathbb{Z}_q$ we have that $\Pr[v = \tilde{v} \mid k_1 = \tilde{k}] = \Pr[\sigma_1 \in \Sigma(\tilde{v}, \tilde{k}) \mid k_1 = \tilde{k}] = |\Sigma(\tilde{v}, \tilde{k})|/q = |\Sigma(\tilde{v}, 0)|/q$. The right-hand side only depends on \tilde{v} , and so v is independent of k_1 . \square

Corollary 4.1. *If q , m and g all are power of 2 satisfying $q = mg$, and d , m and g satisfy $m + 2d < g$, then the AKCN-simple described in Algorithm 4 is correct and secure.*

Proof. For correctness, suppose $|\sigma_1 - \sigma_2|_q \leq d$, then there exist $\delta \in [-d, d]$ and $\theta \in \mathbb{Z}$ such that $\sigma_2 = \sigma_1 + \theta q + \delta$. From the formula calculating v , there exist $\theta' \in \mathbb{Z}$ and $\varepsilon \in (-1/2, 1/2]$ such that $v = \sigma_1 2^{-\bar{m}} + k_1 2^{\bar{g}-\bar{m}} + \varepsilon + \theta' g$. Taking these into the formula computing k_2 , we have

$$k_2 = \lfloor k_1 + (m\varepsilon - \delta)/g \rfloor \bmod m$$

If $m + 2d < g$, then $|k_1 + (m\varepsilon - \delta)/g| < 1/2$, so that $k_1 = k_2$.

As a special case of the AKCN scheme, the security of the AKCN-simple scheme in Algorithm 4 directly follows from that of Algorithm 3. \square

4.2 Discussions on KC vs. AKC

Key establishment (KE) schemes based upon KC and AKC have different performances and features.

- KC-based KE corresponds to Diffie-Hellman key exchange in the lattice world, while AKC-based to El Gamal key transport.
- When deploying AKC-based KE in practice, if the randomness used by the responder (e.g., a low-power device like smart card) is poor, it will significantly ruin the session-key security. Or, if the responder is just lazy (or for economic reasons), who may re-use session-keys across multiple sessions, as demonstrated with some deployed TLS implementations in reality. In comparison, with KC-based KE, the two players play a symmetric role in generating the session-key, and thus the damage caused by poor randomness can be alleviated. In addition, symmetry is usually a desirable feature for cryptographic schemes in practice.
- On the same parameters (q, m, g) (which imply the same bandwidth), SKCN-based KE has lower error probability than AKCN-based. Or, on the same parameters (q, m, d) (which

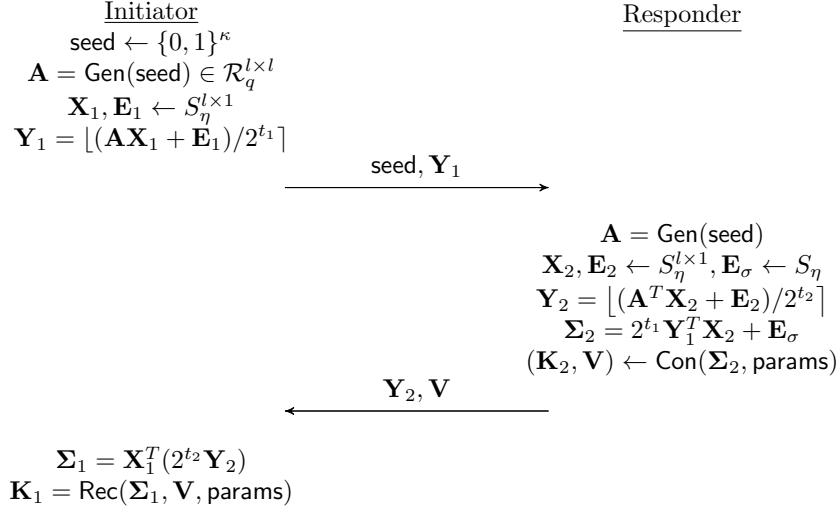


Figure 1: Generic construction of SKCN-MLWE

imply the same error probability), SKCN-based KE has smaller bandwidth than AKCN-based. This comparison is enabled by the upper-bounds on these parameters proved in Theorem 3.1 and 4.1.

- KC-based KE is more versatile, in the sense that it can also be straightforwardly adapted into a key transport protocol or a CPA-secure PKE scheme. Moreover, in another work submitted to the same conference [CGZ18], we show that the deterministic version of SKCN is also a fundamental building tool for lattice-based signature.
- KC-based KE is more appropriate for incorporating into the existing standards like IKE and TLS that are based on Diffie-Hellman via the SIGMA mechanism [Kra03]. We note that key transport is explicitly abandoned with TLS1.3 [Res].
- For the parameters proposed in this work, SKCN is actually (slightly) more efficient than AKCN.

For the above reasons, we focus more on KC-based key establishment (specifically, key exchange) than AKC-based in this work. Still, we aim for a unified protocol structure that can be instantiated with either KC or AKC, in order to simplify system complexity.

5 Constructions of MLWE-Based Key Establishment

Let $KC = (\text{Con}, \text{Rec}, \text{params})$ be a *correct* and *secure* KC or AKC scheme with parameters $\text{params} = (q, m, g, d)$, where $m = 2$ in this section. When Con and Rec are applied to a polynomial in \mathcal{R}_q , they are applied to each coefficients of the polynomial respectively. SKCN-based key exchange from MLWE is depicted in Figure 1, and AKCN-based key transport from MLWE is presented in Figure 2. When being cased into the public-key setting, its specification is given in Section 5.1. For provable security, t_1 is set to be 0. Here, for simplicity and symmetry, we assume the same number of tail bits are chopped off from both Y_1 and Y_2 by setting $t = t_1 = t_2 \geq 0$.

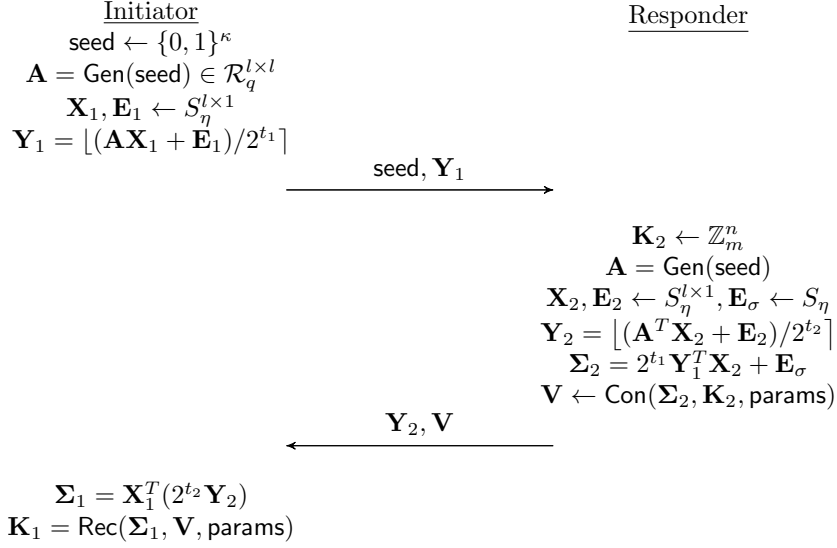


Figure 2: Generic construction of AKCN-MLWE, where $m = 2$ in this work

5.1 KEM Specification of SKCN-MLWE in Public-Key Setting

When being casted into the public-key setting, the specification of SKCN-MLWE is given below. The adaption to the specification of AKCN-MLWE is straightforward, and is specified in Section 5.2. For presentation simplicity, we simply set the resultant shared-key to be $\mathbf{K} = \mathbf{K}_1 = \mathbf{K}_2$. In actual implementation of KEM, the shared-key is derived from \mathbf{K} and the interaction transcript via some key derivation function (e.g., a cryptographic hash function or HMAC, etc).

Algorithm 5 $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\kappa)$

- 1: $\text{seed} \leftarrow \{0, 1\}^\kappa$
 - 2: $\mathbf{A} := \text{Gen}(\text{seed})$
 - 3: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow S_\eta^{l \times 1}$
 - 4: $\mathbf{Y}_1 := \lfloor (\mathbf{A}\mathbf{X}_1 + \mathbf{E}_1)/2^{t_1} \rfloor$
 - 5: **return** $(\text{pk} := (\text{seed}, \mathbf{Y}_1), \text{sk} := \mathbf{X}_1)$
-

Algorithm 6 $(\text{ct}, \text{key}) \leftarrow \text{Encaps}(\text{pk})$

- 1: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow S_\eta^{l \times 1}, \mathbf{E}_\sigma \leftarrow S_\eta$
 - 2: $\mathbf{A} := \text{Gen}(\text{seed})$
 - 3: $\mathbf{Y}_2 := \lfloor (\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2)/2^{t_2} \rfloor$
 - 4: $\boldsymbol{\Sigma}_2 := 2^{t_1} \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$
 - 5: $(\mathbf{K}_2, \mathbf{V}) \leftarrow \text{Con}(\boldsymbol{\Sigma}_2, \text{params})$
 - 6: **return** $(\text{ct} := (\mathbf{Y}_2, \mathbf{V}), \text{key} := \mathbf{K}_2)$
-

Algorithm 7 $\text{key}' \leftarrow \text{Decaps}(\text{sk}, \text{ct})$

- 1: $\boldsymbol{\Sigma}_1 := \mathbf{X}_1^T (2^{t_2} \mathbf{Y}_2)$
 - 2: $\mathbf{K}_1 := \text{Rec}(\boldsymbol{\Sigma}_1, \mathbf{V}, \text{params})$
 - 3: **return** $\text{key}' := \mathbf{K}_1$
-

5.2 KEM Specification of AKCN-MLWE in Public-Key Setting

Algorithm 8 $(pk, sk) \leftarrow \text{KeyGen}()$

```

1: seed  $\leftarrow \{0, 1\}^\kappa$ 
2:  $\mathbf{A} := \text{Gen}(\text{seed})$ 
3:  $\mathbf{X}_1, \mathbf{E}_1 \leftarrow S_\eta^{l \times 1}$ 
4:  $\mathbf{Y}_1 := \lfloor (\mathbf{A}\mathbf{X}_1 + \mathbf{E}_1)/2^{t_1} \rfloor$ 
5: return  $(pk := (\text{seed}, \mathbf{Y}_1), sk := \mathbf{X}_1)$ 

```

Algorithm 9 $(ct, key) \leftarrow \text{Encaps}(pk)$

```

1:  $\mathbf{K}_2 \leftarrow \mathbb{Z}_m^n$ 
2:  $\mathbf{X}_2, \mathbf{E}_2 \leftarrow S_\eta^{l \times 1}, \mathbf{E}_\sigma \leftarrow S_\eta$ 
3:  $\mathbf{A} := \text{Gen}(\text{seed})$ 
4:  $\mathbf{Y}_2 := \lfloor (\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2)/2^{t_2} \rfloor$ 
5:  $\Sigma_2 := 2^{t_1} \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$ 
6:  $\mathbf{V} \leftarrow \text{Con}(\Sigma_2, \mathbf{K}_2, \text{params})$ 
7: return  $(ct := (\mathbf{Y}_2, \mathbf{V}), key := \mathbf{K}_2)$ 

```

Algorithm 10 $key' \leftarrow \text{Decaps}(sk, ct)$

```

1:  $\Sigma_1 := \mathbf{X}_1^T (2^{t_2} \mathbf{Y}_2)$ 
2:  $\mathbf{K}_1 := \text{Rec}(\Sigma_1, \mathbf{V}, \text{params})$ 
3: return  $key' := \mathbf{K}_1$ 

```

6 Analysis of MLWE-Based Key Establishment

6.1 Security Analysis

Theorem 6.1. *If $(\text{params}, \text{Con}, \text{Rec})$ is a correct and secure KC or AKC scheme, the key exchange protocol described in Figure 1 and 2 is secure under the MLWE assumption.*

Definition 6.1. *A KC or AKC based key exchange protocol from LWE is secure, if for any sufficiently large security parameter λ and any PT adversary \mathcal{A} , $|\Pr[b' = b] - \frac{1}{2}|$ is negligible, as defined w.r.t. game G_0 specified in Algorithm 11.*

Algorithm 11 Game G_0

```

1:  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ 
2:  $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$ 
3:  $\mathbf{Y}_1 = \mathbf{A}\mathbf{X}_1 + \mathbf{E}_1$ 
4:  $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ 
5:  $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$ 
6:  $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$ 
7:  $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$ 
8:  $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$ 
9:  $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
10:  $b \leftarrow \{0, 1\}$ 
11:  $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 

```

Before starting to prove the security, we first recall some basic properties of the LWE assumption. The following lemma is derived by a direct hybrid argument [PVW08, BCD⁺16].

Lemma 6.1 (LWE in the matrix form). *For positive integer parameters $(\lambda, n, q \geq 2, l, t)$, where n, q, l, t all are polynomial in λ , and a distribution χ over \mathbb{Z}_q , denote by $L_\chi^{(l,t)}$ the distribution over $\mathbb{Z}_q^{t \times n} \times \mathbb{Z}_q^{t \times l}$ generated by taking $\mathbf{A} \leftarrow \mathbb{Z}_q^{t \times n}, \mathbf{S} \leftarrow \chi^{n \times l}, \mathbf{E} \leftarrow \chi^{t \times l}$ and outputting $(\mathbf{A}, \mathbf{AS} + \mathbf{E})$. Then, under the standard LWE assumption on indistinguishability between $A_{q,s,\chi}$ (with $\mathbf{s} \leftarrow \chi^n$) and $\mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$, no PT distinguisher \mathcal{D} can distinguish, with non-negligible probability, between the distribution $L_\chi^{(l,t)}$ and $\mathcal{U}(\mathbb{Z}_q^{t \times n} \times \mathbb{Z}_q^{t \times l})$ for sufficiently large λ .*

Theorem 6.2. *If $(\text{params}, \text{Con}, \text{Rec})$ is a correct and secure KC or AKC scheme, the key exchange protocol described in Figure ?? is secure under the (matrix form of) LWE assumption.*

Proof. The proof is similar to, but actually simpler than, that in [Pei14, BCD⁺16]. The general idea is that we construct a sequence of games: G_0, G_1 and G_2 , where G_0 is the original game for defining security. In every move from game G_i to G_{i+1} , $0 \leq i \leq 1$, we change a little. All games G_i 's share the same PT adversary \mathcal{A} , whose goal is to distinguish between the matrices chosen uniformly at random and the matrices generated in the actual key exchange protocol. Denote by T_i , $0 \leq i \leq 2$, the event that $b = b'$ in Game G_i . Our goal is to prove that $\Pr[T_0] < 1/2 + \text{negl}$, where negl is a negligible function in λ . For ease of readability, we re-produce game G_0 below. For presentation simplicity, in the subsequent analysis, we always assume the underlying KC or AKC is correct. The proof can be trivially extended to the case that correctness holds with overwhelming probability (i.e., failure occurs with negligible probability).

Algorithm 12 Game G_0	Algorithm 13 Game G_1
1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$	1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$
2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$	2: $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$
3: $\mathbf{Y}_1 = \mathbf{A}\mathbf{X}_1 + \mathbf{E}_1$	3: $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$
4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$	4: $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$
5: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$	5: $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$
6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$	6: $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$
7: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$	7: $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$
8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$	8: $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$
9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$	9: $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$
10: $b \leftarrow \{0, 1\}$	10: $b \leftarrow \{0, 1\}$
11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$	11: $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$

Lemma 6.2. $|\Pr[T_0] - \Pr[T_1]| < \text{negl}$, under the indistinguishability between $L_\chi^{(l_A, n)}$ and $\mathcal{U}(\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A})$.

Proof. Construct a distinguisher \mathcal{D} , in Algorithm 14, who tries to distinguish $L_\chi^{(l_A, n)}$ from $\mathcal{U}(\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A})$.

Algorithm 14 Distinguisher \mathcal{D}

```

1: procedure  $\mathcal{D}(\mathbf{A}, \mathbf{B})$   $\triangleright \mathbf{A} \in \mathbb{Z}_q^{n \times n}, \mathbf{B} \in \mathbb{Z}_q^{n \times l_A}$ 
2:    $\mathbf{Y}_1 = \mathbf{B}$ 
3:    $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ 
4:    $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$ 
5:    $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$ 
6:    $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$ 
7:    $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$ 
8:    $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
9:    $b \leftarrow \{0, 1\}$ 
10:   $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 
11:  if  $b' = b$  then
12:    return 1
13:  else
14:    return 0
15:  end if
16: end procedure

```

If (\mathbf{A}, \mathbf{B}) is subject to $L_\chi^{(l_A, n)}$, then \mathcal{D} perfectly simulates G_0 . Hence, $\Pr \left[\mathcal{D} \left(L_\chi^{(l_A, n)} \right) = 1 \right] = \Pr[T_0]$. On the other hand, if (\mathbf{A}, \mathbf{B}) is chosen uniformly at random from $\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^{n \times l_A}$, which are denoted as $(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U})$, then \mathcal{D} perfectly simulates G_1 . So, $\Pr[\mathcal{D}(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U}) = 1] = \Pr[T_1]$. Hence, $|\Pr[T_0] - \Pr[T_1]| = \left| \Pr[\mathcal{D}(L_\chi^{(l_A, n)}) = 1] - \Pr[\mathcal{D}(\mathbf{A}^\mathcal{U}, \mathbf{B}^\mathcal{U}) = 1] \right| < \text{negl}$. \square \square

Algorithm 15 Game G_1

```

1:  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ 
2:  $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$ 
3:  $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$ 
4:  $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ 
5:  $\mathbf{Y}_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2$ 
6:  $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$ 
7:  $\Sigma_2 = \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma$ 
8:  $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$ 
9:  $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
10:  $b \leftarrow \{0, 1\}$ 
11:  $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 

```

Algorithm 16 Game G_2

```

1:  $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ 
2:  $\mathbf{X}_1, \mathbf{E}_1 \leftarrow \chi^{n \times l_A}$ 
3:  $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$ 
4:  $\mathbf{X}_2, \mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ 
5:  $\mathbf{Y}_2 \leftarrow \mathbb{Z}_q^{n \times l_B}$ 
6:  $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$ 
7:  $\Sigma_2 \leftarrow \mathbb{Z}_q^{l_A \times l_B}$ 
8:  $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\Sigma_2, \text{params})$ 
9:  $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
10:  $b \leftarrow \{0, 1\}$ 
11:  $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 

```

Lemma 6.3. $|\Pr[T_1] - \Pr[T_2]| < \text{negl}$, under the indistinguishability between $L_\chi^{(l_B, n+l_A)}$ and $\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})$.

Proof. As \mathbf{Y}_1 is subject to uniform distribution in G_1 , $(\mathbf{Y}_1^T, \Sigma_2)$ can be regarded as an $L_\chi^{(l_B, l_A)}$ sample of secret \mathbf{X}_2 and noise \mathbf{E}_σ . Based on this observation, we construct the following distinguisher \mathcal{D}' .

Algorithm 17 Distinguisher \mathcal{D}'

```

1: procedure  $\mathcal{D}'(\mathbf{A}', \mathbf{B})$  where  $\mathbf{A}' \in \mathbb{Z}_q^{(n+l_A) \times n}$ ,  $\mathbf{B} \in \mathbb{Z}_q^{(n+l_A) \times l_B}$ 
2:   Denote  $\mathbf{A}' = \begin{pmatrix} \mathbf{A}^T \\ \mathbf{Y}_1^T \end{pmatrix}$   $\triangleright \mathbf{A} \in \mathbb{Z}_q^{n \times n}, \mathbf{Y}_1^T \in \mathbb{Z}_q^{l_A \times n}$ 
3:   Denote  $\mathbf{B} = \begin{pmatrix} \mathbf{Y}_2 \\ \boldsymbol{\Sigma}_2 \end{pmatrix}$   $\triangleright \mathbf{Y}_2 \in \mathbb{Z}_q^{n \times l_B}, \boldsymbol{\Sigma}_2 \in \mathbb{Z}_q^{l_A \times l_B}$ 
4:    $(\mathbf{K}_2^0, \mathbf{V}) \leftarrow \text{Con}(\boldsymbol{\Sigma}_2, \text{params})$ 
5:    $\mathbf{K}_2^1 \leftarrow \mathbb{Z}_m^{l_A \times l_B}$ 
6:    $b \leftarrow \{0, 1\}$ 
7:    $b' \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{Y}_1, \lfloor \mathbf{Y}_2/2^t \rfloor, \mathbf{K}_2^b, \mathbf{V})$ 
8:   if  $b' = b$  then
9:     return 1
10:  else
11:    return 0
12:  end if
13: end procedure

```

If $(\mathbf{A}', \mathbf{B})$ is subject to $L_\chi^{(l_B, n+l_A)}$, $\mathbf{A}' \leftarrow \mathbb{Z}_q^{(n+l_A) \times n}$ corresponds to $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$ and $\mathbf{Y}_1 \leftarrow \mathbb{Z}_q^{n \times l_A}$ in G_1 ; and $\mathbf{S} \leftarrow \chi^{n \times l_B}$ (resp., $\mathbf{E} \leftarrow \chi^{(n+l_A) \times l_B}$) in generating $(\mathbf{A}', \mathbf{B})$ corresponds to $\mathbf{X}_2 \leftarrow \chi^{n \times l_B}$ (resp., $\mathbf{E}_2 \leftarrow \chi^{n \times l_B}$ and $\mathbf{E}_\sigma \leftarrow \chi^{l_A \times l_B}$) in G_1 . In this case, we have

$$\begin{aligned}
\mathbf{B} &= \mathbf{A}'\mathbf{S} + \mathbf{E} = \begin{pmatrix} \mathbf{A}^T \\ \mathbf{Y}_1^T \end{pmatrix} \mathbf{X}_2 + \begin{pmatrix} \mathbf{E}_2 \\ \mathbf{E}_\sigma \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2 \\ \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma \end{pmatrix} = \begin{pmatrix} \mathbf{Y}_2 \\ \boldsymbol{\Sigma}_2 \end{pmatrix}
\end{aligned}$$

Hence $\Pr \left[\mathcal{D}' \left(L_\chi^{(l_B, n+l_A)} \right) = 1 \right] = \Pr[T_1]$.

On the other hand, if $(\mathbf{A}', \mathbf{B})$ is subject to uniform distribution $\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})$, then $\mathbf{A}, \mathbf{Y}_1, \mathbf{Y}_2, \boldsymbol{\Sigma}_2$ all are also uniformly random; So, the view of \mathcal{D}' in this case is the same as that in game G_2 . Hence, $\Pr[\mathcal{D}'(\mathbf{A}', \mathbf{B}) = 1] = \Pr[T_2]$ in this case. Then $|\Pr[T_1] - \Pr[T_2]| = |\Pr[\mathcal{D}'(L_\chi^{(l_B, n+l_A)}) = 1] - \Pr[\mathcal{D}'(\mathcal{U}(\mathbb{Z}_q^{(n+l_A) \times n} \times \mathbb{Z}_q^{(n+l_A) \times l_B})) = 1]| < \text{negl}$. \square \square

Lemma 6.4. *If the underlying KC or AKC is secure, $\Pr[T_2] = \frac{1}{2}$.*

Proof. Note that, in Game G_2 , for any $1 \leq i \leq l_A$ and $1 \leq j \leq l_B$, $(\mathbf{K}_2^0[i, j], \mathbf{V}[i, j])$ only depends on $\boldsymbol{\Sigma}_2[i, j]$, and $\boldsymbol{\Sigma}_2$ is subject to uniform distribution. By the *security* of KC, we have that, for each pair (i, j) , $\mathbf{K}_2^0[i, j]$ and $\mathbf{V}[i, j]$ are independent, and $\mathbf{K}_2^0[i, j]$ is uniform distributed. Hence, \mathbf{K}_2^0 and \mathbf{V} are independent, and \mathbf{K}_2^0 is uniformly distributed, which implies that $\Pr[T_2] = 1/2$. \square \square

This finishes the proof of Theorem 6.1. \square \square

6.2 Error Probability Analysis

Denote $\varepsilon_2 = \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2 - 2^t \lfloor (\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2) / 2^t \rfloor$, and $\varepsilon_1 = \mathbf{A} \mathbf{X}_1 + \mathbf{E}_1 - 2^t \lfloor (\mathbf{A} \mathbf{X}_1 + \mathbf{E}_1) / 2^t \rfloor$. Then we have

$$\boldsymbol{\Sigma}_1 - \boldsymbol{\Sigma}_2 = \mathbf{X}_1^T (2^t \mathbf{Y}_2) - (2^t \mathbf{Y}_1^T \mathbf{X}_2 + \mathbf{E}_\sigma) \quad (6)$$

$$= 2^t \mathbf{X}_1^T [(\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2)/2^t] - ((2^t \lfloor (\mathbf{A} \mathbf{X}_1 + \mathbf{E}_1)/2^t \rfloor)^T \mathbf{X}_2 + \mathbf{E}_\sigma) \quad (7)$$

$$= \mathbf{X}_1^T (\mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2 - \boldsymbol{\varepsilon}_2) - ((\mathbf{A} \mathbf{X}_1 + \mathbf{E}_1 - \boldsymbol{\varepsilon}_1)^T \mathbf{X}_2 + \mathbf{E}_\sigma) \quad (8)$$

$$= \mathbf{X}_1^T (\mathbf{E}_2 - \boldsymbol{\varepsilon}_2) - (\mathbf{E}_1 - \boldsymbol{\varepsilon}_1)^T \mathbf{X}_2 - \mathbf{E}_\sigma \quad (9)$$

From MLWE assumption, $(\mathbf{A}, \mathbf{A}^T \mathbf{X}_2 + \mathbf{E}_2)$ is indistinguishable with (\mathbf{A}, \mathbf{U}) , where \mathbf{U} is subjected to the uniform distribution, $\boldsymbol{\varepsilon}_i (i = 1, 2)$ should be closed to $\mathbf{U} - 2^t \lfloor \mathbf{U}/2^t \rfloor$. We can roughly regard each coefficients of polynomials in $\mathbf{U} - 2^t \lfloor \mathbf{U}/2^t \rfloor$ as uniform distribution over $[-2^{t-1}, 2^{t-1}]^n$.

Then we can calculate the standard deviation, denoted by s , of each coefficients of polynomials in $\boldsymbol{\Sigma}_2 - \boldsymbol{\Sigma}_1$. We have

$$s^2 = 2nl\sigma^2 \left(\sigma^2 + \frac{(1 + 2^t)^2 - 1}{12} \right) + \sigma^2 \quad (10)$$

The actual error probability is then gotten by running program scripts.

7 Parameter Selection and Comparison

The parameters and performance of SKCN-MLWE and AKCN-MLWE are presented in Table 1 and Table 2 respectively. There, “pq-sec” refers to the best known quantum attack against the underlying lattice problem w.r.t. $t = 0$. The concrete security values are gotten by running the scripts provided in [BDK⁺17]. We believe that chopping off t least significant bits from \mathbf{Y}_1 and \mathbf{Y}_2 can essentially strengthen the security in reality (particularly for ephemeral key exchange or transport).

Note that the security is proved with $t_1 = 0$, but in the actual implementation we set $t_1 = t_2 = t$ as in Kyber [BDK⁺17]. We made this choice based on the following considerations:

- Setting $t_1 = t_2 = t > 0$ minimizes the bandwidth, without sacrificing the actual security in any meaningful way (a detailed explanation is given in [BDK⁺17]).
- Symmetry in protocol structure is always a desirable feature in practice.

Of course, if one strictly concerns about provable security, we can always implement the protocol with $t_1 = 0$. On the same parameters, with $t_1 = 0$ the error probability is further lowered while the bandwidth (specifically, the size of \mathbf{Y}_1) is accordingly increased.

	$ K $	n	q	η	g	t	l	pq-sec	err	pk (B)	cipher (B)	bw. (B)
SKCN-MLWE-KE-light	256	256	7681	5	2^3	4	2	102	$2^{-36.2}$	608	672	1280
SKCN-MLWE-PKE-light	256	256	7681	5	2^3	3	2	102	$2^{-105.5}$	672	736	1408
SKCN-MLWE-KE-Recommended	256	256	7681	2	2^4	4	3	147	$2^{-76.1}$	896	992	1888
SKCN-MLWE-PKE-Recommended	256	256	7681	2	2^2	3	3	147	$2^{-166.4}$	992	1024	2016
SKCN-MLWE-KE-Paranoid	512	512	7681	8	2^4	1	2	248	$2^{-60.8}$	1568	1792	3360

Table 1: Parameters for SKCN-MLWE. KE refers to parameters aimed for ephemeral key exchange, and PKE to parameters aimed for CCA-secure public-key encryption.

	$ K $	n	q	η	g	t	l	pq-sec	err	pk (B)	cipher (B)	bw. (B)
AKCN-MLWE-KT-light	256	256	7681	5	2^3	4	2	102	$2^{-36.2}$	608	704	1312
AKCN-MLWE-PKE-light	256	256	7681	5	2^3	3	2	102	$2^{-105.5}$	672	768	1440
Kyber-light	256	256	7681	5	2^3	2	2	102	2^{-145}	736	832	1568
AKCN-MLWE-KT-Recommended	256	256	7681	2	2^4	4	3	147	$2^{-67.1}$	896	992	1888
AKCN-MLWE-PKE-Recommended	256	256	7681	2	2^3	3	3	147	$2^{-166.4}$	992	1056	2048
Kyber-Recommended	256	256	7681	4	2^3	2	3	161	$2^{-142.7}$	1088	1152	2240
AKCN-MLWE-KT-Paranoid	512	512	7681	8	2^6	1	2	248	$2^{-64.1}$	1568	1920	3488
Kyber-Paranoid	256	256	7681	3	2^3	2	4	218	2^{-169}	1440	1536	2976

Table 2: Parameters for AKCN-MLWE, and comparisons with Kyber. KT refers to parameters aimed for ephemeral key transport, and PKE to parameters aimed for CCA-secure public-key encryption.

7.1 Comparison with Kyber

Kyber is based on MLWE, and is AKC-based key transport protocol. In Kyber, $\mathbf{Y}_1 = \lfloor 2^{d_t}(\mathbf{A}\mathbf{X}_1 + \mathbf{E}_1)/q \rfloor$, $\mathbf{Y}_2 = \lfloor 2^{d_u}(\mathbf{A}^T\mathbf{X}_2 + \mathbf{E}_2)/q \rfloor$, $\mathbf{\Sigma}_2 = \lfloor q\mathbf{Y}_1/2^{d_t} \rfloor^T \mathbf{X}_2 + \mathbf{E}_\sigma$, $\mathbf{\Sigma}_1 = \mathbf{X}_1^T \lfloor q\mathbf{Y}_2/2^{d_u} \rfloor$, where d_t, d_u are non-negative integers. The underlying AKC mechanism of KYBER is essentially AKCN (Algorithm 3). Note that $\text{Rec}(\sigma_2, v, \text{params}) = \lfloor m \cdot (\lfloor qv/g \rfloor / q - \sigma_2/q) \rfloor \bmod m$ in KYBER, compared to $\text{Rec}(\sigma_2, v, \text{params}) = \lfloor m \cdot (v/g - \sigma_2/q) \rfloor \bmod m$ in AKCN.

In comparison, we present both SKCN-based key exchange and AKCN-based key transport. Moreover, the underlying key building tools: SKCN and AKCN, appeared in the literature since November 2016 <https://arxiv.org/abs/1611.06150>. In fact, the SKCN-MLWE and AKCN-MLWE protocols are the MLWE-based instantiations of SKCN-LWE and AKCN-LWE presented there. On all the chosen parameters, we can see that SKCN-MLWE protocols have better performance than their AKCN-MLWE counterparts. Below, we briefly compare AKCN-MLWE and Kyber.

- For AKCN-MLWE-PKE-light and Kyber-light, they both achieve 102-bit post-quantum security, but AKCN-MLWE-PKE-light (resp., Kyber-light) has bandwidth of 1440 bytes (resp., 1568 bytes) at the failure rate 2^{-105} (resp., 2^{-145}). As the post-quantum security level is for 102 bits and $105 > 102$, we suggest an error probability of 2^{-105} already suffices for 102-bit post-quantum security. Also, three (resp., two) least significant bits are chopped off with AKCN-MLWE-PKE-light (resp., Kyber-light), which means that AKCN-MLWE-PKE-light can have stronger security than Kyber-light in reality. In addition, we also present parameters for ephemeral key establishment with remarkably lower bandwidth: 1280 (resp., 1312) bytes for SKCN-MLWE-KE-light (resp., AKCN-MLWE-KT-light).
- For AKCN-MLWE-PKE-Recommended and Kyber-Recommended, they both have enough margins for 128-bit post-quantum security, where AKCN-MLWE-PKE (resp., Kyber-Recommended) has 147 (resp., 161) bit post-quantum security w.r.t. $t = 0$ at failure rate of $2^{-166.4} << 2^{-147}$ (resp., $2^{-142.7} >> 2^{-161}$). Notice that three (resp., two) least significant bits are chopped off with AKCN-MLWE-PKE-Recommended (resp., Kyber-Recommended). AKCN-MLWE-PKE-Recommended has bandwidth of 2048 bytes, compared with 2240 bytes of Kyber-Recommended. In addition, we also present recommended parameters for ephemeral key establishment: 1888 bytes at failure rate $2^{-76.1}$ (resp., $2^{-67.1}$) for SKCN-MLWE-KE-Recommended (resp., AKCN-MLWE-KT-Recommended).
- AKCN-MLWE-KE-Paranoid and Kyber-Paranoid are somewhat incomparable. AKCN-MLWE-KE-Paranoid is for ephemeral key establishment, which achieves 512-bit shared

key at 248-bit post-quantum security and failure rate of 2^{-64} . In comparison, Kyber-Paranoid is for achieving 256-bit shared key at post-quantum security of 218 and failure rate of 2^{-169} . We remark that 256-bit shared-key may only ensure about 128-bit security in the quantum world, in view of the quadratic speedup by Grover’s search algorithm and the recent advances of more sophisticated quantum attacks against symmetric-key cryptography.

References

- [AGKS05] M. Abe, R. Gennaro, K. Kurosawa and V. Shoup. Tag-KEM/DEM: A New Framework for Hybrid Encryption and A New Analysis of Kurosawa-DESmedt KEM. *EUROCRYPT 2005*: 128-146.
- [A17] M. R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HELib and SEAL. *EUROCRYPT 2017*: 103-129.
- [APS15] M. R. Albrecht, R. Player and S. Scott. On the Concrete Hardness of Learning with Errors. *Journal of Mathematical Cryptology*, Volume 9, Issue 3, pages 169-203, 2015.
- [ADPS16] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum Key Exchange — A New Hope. *25th USENIX Security Symposium (USENIX Security 16)*, pages 327–343.
- [ADPS16b] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. NewHope without Reconciliation. *Cryptology ePrint Archive*, Report 2016/1157, 2016.
- [AJS16] E. Alkim, P. Jakubeit, and P. Schwabe. A New Hope on ARM Cortex-M. *Cryptology ePrint Archive*, Report 2016/758, 2016.
- [ACPS09] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. *CRYPTO 2009*: 595-618.
- [BLL⁺15] S. Bai, A. Langlois, T. Lepoint, D. Stehlé, and R. Steinfeld. Improved Security Proofs in Lattice-Based Cryptography: Using the Rényi Divergence rather than the Statistical Distance. *ASIACRYPT 2015*: 3-24.
- [BPR12] A. Banerjee and C. Peikert and A. Rosen. Pseudorandom Functions and Lattices. *EUROCRYPT 2012*: 719-737.
- [BGL+18] S. Bhattacharya, O. Garcia-Morchon, T. Laarhoven, R. Rietman, M. Saarinen, L. Tolhuizen, and Z. Zhang. Round5: Compact and Fast Post-Quantum Public-Key Encryption. *Cryptology ePrint Archive*, 2018/725.
- [BBG+17] H. Baan, S. Bhattacharya, O. Garcia-Morchon, R. Rietman, L. Tolhuizen, J.L. Torre-Arce, and Z. Zhang. Round2: KEM and PKE based on GLWR. *Cryptology ePrint Archive*, 2017/1183.
- [BGM⁺16] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen. On the Hardness of Learning with Rounding over Small Modulus. *TCC 2016*: 209-224.

- [BCD⁺16] J. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan, and D. Stebila. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. *ACM CCS 2016*: 1006-1018.
- [BCNS15] J.W. Bos, C. Costello, M. Naehrig, and D. Stebila. Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors Problem. *IEEE Symposium on Security and Privacy 2015*, 553-570.
- [BDK⁺17] J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, D. Stehlé. CRYSTALS-Kyber: a CCA-Secure Module-lattice-based KEM. Available from: <https://pq-crystals.org/> Preliminary version appears at Euro S&P 2018.
- [CZZ18] R. Chen, Z. Zhang and Z. Zhang. On the Hardness of the Computational Ring-LWR Problem and its Applications. *ASIACRYPT 2018*. Available from Cryptology ePrint Archive, 2018/536.
- [CN11] Y. Chen and P.Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. *ASIACRYPT 2011*: 1-20.
- [CGZ18] L. Cheng, B. Gong, and Y. Zhao. Lattice-Based Signature from Key Consensus. Cryptology ePrint Archive, Report 2018/1180. <https://eprint.iacr.org/2018/1180>
- [CKLS16] J.H. Cheon, D. Kim, J. Lee, and Y. Song. Lizard: Cut Off the Tail! Practical Post-Quantum Public-Key Encryption from LWE and LWR. *Cryptology ePrint Archive*, Report 2016/1126, 2016.
- [CW90] D. Coppersmith and S. Winograd. Matrix Multiplication via Arithmetic Progressions. *Journal of Symbolic Computation*, volume 9, issue 3, pages 251-280, 1990.
- [CDS94] R. Cramer, I. Damgrd and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. *CRYPTO 1994*: 174187.
- [DKRV17] J. D’Anvers, A. Karmakar, S.S. Roy, and F. Vercauteren. SABER: Mod-LWR based KEM. Proposal to NIST PQC Standardization.
- [DXL12] J. Ding, X. Xie and X. Lin. A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. *Cryptology ePrint Archive*, Report 2012/688, 2012.
- [DORS08] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM Journal on Computing*, volume 38, issue 1, pages 97-139, 2008.
- [DD12] L. Ducas and A. Durmus. Ring-LWE in Polynomial Rings. *PKC 2012*: 34-51.
- [DTV15] A. Duc, F. Tramèr, and S. Vaudenay. Better Algorithms for LWE and LWR. *EUROCRYPT 2015*: 173-202.
- [FS86] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. *CRYPTO 1986*: 186194.

- [FO99] E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* Volume 83, Issue 1, pages 24-32, 1999.
- [FO13] E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. *Journal of Cryptology*, Volume 26, Issue 1, pages 80-101, 2013.
- [GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. *ACM STOC 2008*: 197-206.
- [GS16] S. Gueron and F. Schlieker. Speeding Up R-LWE Post-Quantum Key Exchange. *Cryptology ePrint Archive*, Report 2016/467, 2016.
- [KLL15] M. Kaplan, G. Leurent, A. Leverrier and M. Naya-Plasencia. Quantum Differential and Linear Cryptanalysis. *ArXiv Preprint*: 1510.05836, 2015.
- [Kra03] H. Krawczyk. *SIGMA: The ‘SIGN-and-MAC’ Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols CRYPTO 2003*: 400-425.
- [KM10] H. Kuwakado and M. Morii. Quantum Distinguisher between the 3-round Feistel Cipher and the Random Permutation. *IEEE ISIT 2010*: 2682-2685.
- [LP10] R. Lindner and C. Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. *CT-RSA 2011*: 319-339. Also available from <http://eprint.iacr.org/2010/613>.
- [LS15] A. Langlois and D. Stehlé. Worst-case to Average-case Reductions for Module Lattices. *Des. Codes Cryptography*, 75(3): 565-599, 2015.
- [LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On Ideal Lattices and Learning with Errors over Rings. *EUROCRYPT 2010*: 1-23.
- [LPR13] V. Lyubashevsky, C. Peikert, and O. Regev. A Toolkit for Ring-LWE Cryptography. *EUROCRYPT 2013*: 35-54.
- [FrodoKEM] Michael Naehrig, Erdem Alkim, Joppe Bos, Leo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. Supporting documentation: Frodokem. Technical report, National Institute of Standards and Technology, 2017. <https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/Round-1-Submissions>.
- [NIST] NIST. Post-Quantum Cryptography Standardization. <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization>
- [Pei09] C. Peikert. Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem. *STOC 2009*: 333-342.
- [Pei14] C. Peikert. Lattice Cryptography for the Internet. *PQCrypto 2014*: 197-219.

- [Pei16] C. Peikert. A Decade of Lattice Cryptography. In *Foundations and Trends in Theoretical Computer Science*, Volume 10, Issue 4, pages 283-424, 2016.
- [PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A Framework for Efficient and Composable Oblivious Transfer. *CRYPTO 2008*: 554-571.
- [Pop16] A.V. Poppel, Cryptographic Decoding of the Leech Lattice. *Cryptology ePrint Archive*, Report 2016/1050, 2016.
- [PG13] T. Pöppelmann and T. Güneysu. Towards Practical Lattice-Based Public-Key Encryption on Reconfigurable Hardware. *SAC 2013*: 68-85.
- [Reg09] O. Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *Journal of the ACM (JACM)*, Volume 56, Issue 6, pages 34, 2009.
- [Res] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. IETF RFC 8846. <https://datatracker.ietf.org/doc/rfc8446/>.
- [SBG+18] M. Saarinen, S. Bhattacharya, O. Garcia-Morchon, R. Rietman, L. Tolhuizen, and Z. Zhang. Shorter Messages and Faster Post-Quantum Encryption with Round5 on Cortex M. *Cryptology ePrint Archive*, 2018/723.
- [SE94] C. P. Schnorr and M. Euchner. Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems. *Mathematical Programming*, Volume 66, Issue 2, pages 181-199, Springer, 1994.
- [Sim02] M.K. Simon. Probability Distributions Involving Gaussian Random Variables : A Handbook for Engineers and Scientists. Springer, 2012.
- [SM16] D. Stebila and M. Mosca. Post-Quantum Key Exchange for the Internet and the Open Quantum Safe Project. *Cryptology ePrint Archive*, Report 2016/1017, 2016.
- [Str69] V. Strassen. Gaussian Elimination is not Optimal. *Numerische Mathematik*, Volume 13, Issue 4, pages 354-356, Springer, 1969.
Cryptology ePrint Archive: Report 2018/309
- [ZWXZ18] Z. Zheng, X. Wang, G. Xu and C. Zhao. Error Estimation of Practical Convolution Discrete Gaussian Sampling with Rejection Sampling. Cryptology ePrint Archive: Report 2018/309.

A Proof of Theorem 3.1

Before proceeding to prove Theorem 3.1, we first prove the following propositions.

Proposition A.1. *Given $\mathbf{params} = (q, m, g, d, aux)$ for a correct and secure KC scheme. For any arbitrary fixed $\sigma_1 \in \mathbb{Z}_q$, if $\text{Con}(\sigma_1, \mathbf{params})$ outputs (k_1, v) with positive probability, then the value k_1 is fixed w.r.t. the (v, σ_1) . That is, for any random coins (r, r') , if $\text{Con}(\sigma_1, \mathbf{params}, r) = (k_1, v)$ and $\text{Con}(\sigma_1, \mathbf{params}, r') = (k'_1, v)$, then $k_1 = k'_1$.*

Proof. Let $\sigma_2 = \sigma_1$, then $|\sigma_1 - \sigma_2|_q = 0 \leq d$. Then, according to the *correctness* of KC, we have that $k_1 = k_2 = \text{Rec}(\sigma_2, v) = \text{Rec}(\sigma_1, v)$. However, as Rec is a deterministic algorithm, k_2 is fixed w.r.t. (σ_1, v) . As a consequence, k_1 is also fixed w.r.t. (σ_1, v) , no matter what randomness is used by Con . \square

Proposition A.2. *Given $\text{params} = (q, m, g, d, \text{aux})$ for a KC scheme, for any $v \in \mathbb{Z}_g$, let S_v be the set containing all σ_1 such that $\text{Con}(\sigma_1, \text{params})$ outputs v with positive probability. Specifically,*

$$S_v = \{\sigma_1 \in \mathbb{Z}_q \mid \Pr[(k_1, v') \leftarrow \text{Con}(\sigma_1, \text{params}) : v' = v] > 0\}.$$

Then, there exists $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq q/g$.

Proof. For each $\sigma_1 \in \mathbb{Z}_q$, we run $\text{Con}(\sigma_1, \text{params})$ and get a pair $(k_1, v) \in \mathbb{Z}_m \times \mathbb{Z}_g$ satisfying $\sigma_1 \in S_v$. Then, the proposition is clear by the pigeonhole principle. \square

of Theorem 3.1. From Proposition A.2, there exists a $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq q/g$. Note that, for any $\sigma_1 \in S_{v_0}$, $\text{Con}(\sigma_1, \text{params})$ outputs v_0 with positive probability.

For each $i \in \mathbb{Z}_m$, let K_i denote the set containing all σ_1 such that $\text{Con}(\sigma_1, \text{params})$ outputs $(k_1 = i, v = v_0)$ with positive probability. From Proposition A.1, K_i 's form a disjoint partition of S_{v_0} . From the independence between k_1 and v , and the uniform distribution of k_1 , (as we assume the underlying KC is *secure*), we know $\Pr[k_1 = i \mid v = v_0] = \Pr[k_1 = i] > 0$, and so K_i is non-empty for each $i \in \mathbb{Z}_m$. Now, for each $i \in \mathbb{Z}_m$, denote by K'_i the set containing all $\sigma_2 \in \mathbb{Z}_q$ such that $\text{Rec}(\sigma_2, v_0, \text{params}) = i$. As Rec is deterministic, K'_i 's are well-defined and are disjoint.

From the *correctness* of KC, for every $\sigma_1 \in K_i$, $|\sigma_2 - \sigma_1|_q \leq d$, we have $\sigma_2 \in K'_i$. That is, $K_i + [-d, d] \subseteq K'_i$.

We shall prove that $K_i + [-d, d]$ contains at least $|K_i| + 2d$ elements. If $K_i + [-d, d] = \mathbb{Z}_m$, then $m = 1$, which is a contradiction (we exclude the case of $m = 1$ in the definition of KC as it is a trivial case). If there exists an $x \in \mathbb{Z}_m$ such that $x \notin K_i + [-d, d]$, we can see \mathbb{Z}_m as a segment starting from the point x by arranging its elements as $x, (x+1) \bmod m, (x+2) \bmod m, \dots, (x+m-1) \bmod m$. Let l be the left most element in $K_i + [-d, d]$ on the segment, and r be the right most such element. Then $K_i + [-d, d]$ contains at least $|K_i|$ elements between l and r inclusively on the segment. Since $l + [-d, 0]$ and $r + [0, d]$ are subset of $K_i + [-d, d]$, and are not overlap (because $x \notin K_i + [-d, d]$), the set $K_i + [-d, d]$ contains at least $|K_i| + 2d$ elements.

Now we have $|K_i| + 2d \leq |K'_i|$. When we add up on both sides for all $i \in \mathbb{Z}_m$, then we derive $|S_{v_0}| + 2md \leq q$. By noticing that $|S_{v_0}| \geq q/g$, the theorem is established. \square

B Proof of Theorem 4.1

Before proving Theorem 4.1, we first adjust Proposition A.2 to the AKC setting, as following.

Proposition B.1. *Given $\text{params} = (q, m, g, d, \text{aux})$ for an correct and secure AKC scheme, then there exists $v_0 \in \mathbb{Z}_g$ such that $|S_{v_0}| \geq mq/g$.*

Proof. If k_1 is taken uniformly at random from \mathbb{Z}_m , AKC can be considered as a special KC scheme by treating $k_1 \leftarrow \mathbb{Z}_m; v \leftarrow \text{Con}(\sigma_1, k_1, \text{params})$ as $(k_1, v) \leftarrow \text{Con}(\sigma_1, \text{params})$. Consequently, Proposition A.1 holds for this case.

Denote $S'_v \triangleq \{(\sigma_1, k_1) \in \mathbb{Z}_q \times \mathbb{Z}_m \mid \Pr[v' \leftarrow \text{Con}(\sigma_1, k_1, \text{params}) : v' = v] > 0\}$. Then, S_v defined in Proposition A.2 equals to the set containing all the values of σ_1 appeared in $(\sigma_1, \cdot) \in S'_v$.

We run $\text{Con}(\sigma_1, k_1, \text{params})$ for each pair of $(\sigma_1, k_1) \in \mathbb{Z}_q \times \mathbb{Z}_m$. By the pigeonhole principle, there must exist a $v_0 \in \mathbb{Z}_g$ such that $|S'_{v_0}| \geq qm/g$. For any two pairs (σ_1, k_1) and (σ'_1, k'_1) in S'_{v_0} , if $\sigma_1 = \sigma'_1$, from Proposition A.1 we derive that $k_1 = k'_1$, and then $(\sigma_1, k_1) = (\sigma'_1, k'_1)$. Hence, if (σ_1, k_1) and (σ'_1, k'_1) are different, then $\sigma_1 \neq \sigma'_1$, and so $|S_{v_0}| = |S'_{v_0}| \geq mq/g$. \square \square

of Theorem 4.1. By viewing AKC, with $k_1 \leftarrow \mathbb{Z}_q$, as a special KC scheme, all the reasoning in the proof of Theorem 3.1 holds true now. At the end of the proof of Theorem 3.1, we derive $|S_{v_0}| + 2md \leq q$. By taking $|S_{v_0}| \geq mq/g$ according to Proposition B.1, the proof is finished. \square

C On KC/AKC vs. Fuzzy Extractor

Our formulations of KC and AKC are abstractions of the core ingredients of previous constructions of KE and PKE from LWE and its variants. We also note that KC and AKC are similar to fuzzy extractor proposed in [DORS08], which extracts shared-keys from biometrics and noisy data. In this section, we make some discussions on the relationship between KC/AKC and fuzzy extractor.

The differences between the definitions of KC/AKC and that of fuzzy extractor lie mainly in the following ways. Firstly, AKC was not considered within the definitional framework of fuzzy extractor. Secondly, the metric $|\cdot|_q$ we use in defining KC and AKC was not considered for fuzzy extractor. Thirdly, in the definitions of KC and AKC, the algorithm Rec (corresponding Rep for fuzzy extractor) is mandated to be *deterministic*, while in the formulation of fuzzy extractor it is probabilistic. Fourthly, in the formulation of fuzzy extractor [DORS08], w , R and P (corresponding σ_1 , k and v in KC/AKC) are binary strings; while in the definitions of KC/AKC, the corresponding values $\sigma_1 \in \mathbb{Z}_q$, $k \in \mathbb{Z}_m$ and $v \in \mathbb{Z}_g$ have more structured ranges, which are helpful in deriving the exact upper bound. Finally, for the security of KC and AKC, we require that the signal value v be independent of the shared-key k_1 (that can be subject to arbitrary distribution for AKC); roughly speaking, in the definition of fuzzy extractor [DORS08], it is required that the joint distribution (R, P) be statistically close to (U_l, P) where $R \in \{0, 1\}^l$ and U_l is the uniform distribution over $\{0, 1\}^l$.

A generic upper bound on the length of key extracted by fuzzy extractor is proposed in [DORS08, Appendix C]. In comparison, the upper bounds for KC and AKC proved in this work are more versatile and precise w.r.t. the metric $|\cdot|_q$. For example, the effect of the length of the signal v , i.e., the bandwidth parameter g , is not considered in the upper bound for fuzzy extractor, but is taken into account in the upper bounds for KC and AKC.

A generic construction of fuzzy extractor from *secure sketch*, together with a generic construction of *secure sketch* for *transitive metric spaces*, is proposed in [DORS08]. We note that $(\mathbb{Z}_q, |\cdot|_q)$ can be naturally seen as a *transitive metric space*. Compared to the secure sketch based generic constructions of fuzzy extractor, our constructions of KC and AKC are direct and more efficient.

In spite of some similarities between KC/AKC and fuzzy extractors, we remark that before our this work the relation between fuzzy extractor and KE from LWE and its variants is actually opaque. Explicitly identifying and formalizing KC/AKC and reducing lattice-based cryptosystems to KC/AKC in a *black-box* modular way, with inherent bounds on what could or couldn't be done, cut the complexity of future design and analysis of these cryptosystems.