One Sample Ring-LWE with Rounding and Its Application to Key Exchange

Jintai Ding[1], Xinwei Gao[2], Tsuyoshi Takagi[3,4], and Yuntao Wang[3(✉)]

[1] University of Cincinnati, Cincinnati, USA
jintai.ding@gmail.com
[2] Beijing Jiaotong University, Beijing, China
xinwei.gao.7@yandex.com
[3] The University of Tokyo, Tokyo, Japan
{takagi,y-wang}@mist.i.u-tokyo.ac.jp
[4] CREST, Japan Science and Technology Agency, Kawaguchi, Japan

Abstract. In this paper, we introduce a new provably secure ephemeral-only RLWE+Rounding-based key exchange protocol and a proper approach to more accurately estimate the security level of the RLWE problem with only one sample. Since our scheme is an ephemeral-only key exchange, it generates only one RLWE sample from protocol execution. We carefully analyze how to estimate the practical security of the RLWE problem with only one sample, which we call the ONE-sample RLWE problem. Our approach is different from existing approaches that are based on estimation with multiple RLWE samples. Though our analysis is based on some recently developed techniques in Darmstadt, our type of practical security estimate was never done before and it produces security estimates substantial different from the estimates before based on multiple RLWE samples. We show that the new design improves the security and reduce the communication cost of the protocol simultaneously by using one RLWE+Rounding sample technique. We also present two parameter choices ensuring 2^{-60} key exchange failure probability which cover security of AES-128/192/256 with concrete security analysis and implementation. We believe that our construction is secure, simple, efficient and elegant with wide application prospects.

Keywords: Key exchange · Post-quantum · Diffie-Hellman · RLWE · Lattice · One sample

1 Introduction

1.1 The Post-quantum World

Key exchange is a very important cryptographic primitive which allows communicating parties to agree on same keys over insecure network. In 1976, the first key exchange primitive – Diffie-Hellman key exchange protocol was proposed in [20]. This ground-breaking work is a key part of public key cryptography and it inspires cryptographers to build new public key cryptosystems and

R. H. Deng et al. (Eds.): ACNS 2019, LNCS 11464, pp. 323–343, 2019.
https://doi.org/10.1007/978-3-030-21568-2_16

key exchange protocols. With properly chosen parameters and implementations, Diffie-Hellman key exchange and its variants are hard to break with current computing resources.

However, such cryptosystems are no longer secure against sufficiently large quantum computers. In 1994, Shor proposed a quantum algorithm which can solve discrete logarithm problem (DLP) and integer factorization problem (IFP) on a quantum computer [32] in polynomial time. Therefore, if a sufficient large quantum computer is built, Shor's algorithm is expected to break cryptosystems which are constructed based on DLP, IFP and their elliptic curve variants etc., including RSA, DSA, ElGamal etc. It is vital to develop secure and practical post-quantum alternatives for the upcoming post-quantum world.

During recent years, various works are focusing on the lattice-based Ring Learning With Errors (RLWE) problem [25], which is the ring variant of Learning With Errors (LWE) problem [27]. They enjoy high efficiency as well as strong security, making them very promising towards the post-quantum world.

In 2015, NSA announced that it is planning the transition to quantum-resistant cryptography suites in near future. In 2016, NIST formally published calls for new post-quantum cryptography algorithms [19]. This stresses importance and urgency to develop post-quantum alternatives for near future. NIST focused on three primitives: public key encryption, digital signature and key establishment.

1.2 Quantum-Resistant RLWE+Rounding Key Exchange with One Sample

The first complete key exchange solution appeared in the LWE & RLWE-based key exchange protocols proposed by Ding et al. in 2012 [21]. There are various similar works that construct LWE/RLWE-based key exchange protocols, including BCNS [14], NewHope [6], Frodo [13], NewHope-Simple [5], HILA5 [28], Kyber [15] etc. Also there are various new protocols in NIST's round 1 submissions [19].

[11] proposed the Learning With Rounding (LWR) problem, which can reduce communication cost of LWE problem through rounding. Since rounding and recovering algorithms generate deterministic errors, [11] suggests that error term in LWE problem can be discarded with properly chosen parameters. Till now, concrete security of LWR and its ring variant – RLWR problem is not well understood. In LWR and RLWR, "error" on the term $\mathbf{a} \cdot \mathbf{s}$ is only generated by deterministic rounding and recovering algorithm, and this brings security concerns over LWR and RLWR problems. This is also the reason why we prefer the"RLWE+Rounding" approach, instead of using RLWR directly.

Inspired by the notion of RLWR and RLWE-based key exchange, we introduce a new rounding technique dedicated to our key exchange design to reduce the communication cost and increase the security simultaneously. Unlike LWR/RLWR-based cryptosystems, we keep the freshly generated and secret error term $2\mathbf{e}$ in our RLWE instance $\mathbf{a} \cdot \mathbf{s} + 2\mathbf{e}$, then we apply our new rounding technique. We call this a RLWE+Rounding sample. By designing new rounding

and recovering techniques, we reduce communication cost substantially and further improve the practical efficiency of our key exchange protocol. Moreover, it actually adds larger perturbation – "error" on $\mathbf{a} \cdot \mathbf{s}$ compared to standard RLWE instance, which helps to improve security of our protocol even further.

In addition, we give two flexible parameter choices and implementation that ensure low key exchange failure probability and cover security of AES-128/192/256 using our new security analysis technique for only one sample case.

1.3 Parameter Settings for ONE-Sample RLWE Case

It is very clear that for an RLWE-based ephemeral key exchange, an attack can only get one sample. And each RLWE sample can be expanded to n LWE samples by rotating elements in the convolution polynomial ring. Recently a work of [29] developed techniques in solving standard LWE instances with a restricted number of samples. However, it is not adapted in practical security analysis of RLWE key exchanges directly. Especially we can not adopt the LWE-estimator [4] directly because of the perturbations from the rounding/recovering functions in our key exchange scheme. We developed the security analysis of the dual embedding attack (we call "SIS attack" in this work) on solving ONE-sample RLWE case.

Further, in Table 1 we show the complexity of solving the standard LWE instance using SIS attack given n and $2n$ samples. We use Regev's parameter settings $(n, \alpha = \frac{1}{\sqrt{2\pi n \log^2 n}}, q \approx n^2)$ in the original LWE paper [27]. We estimate the hardness of standard LWE for $n = \{128, 512, 1024, 2048\}$ using the LWE-estimator [4] and restrict the number of given samples to n and $2n$. From the table we can see that the gap of complexities is distinctly larger with n increasing. Note that the n and $2n$ samples here can be seen as extracted from ONE-sample RLWE case and TWO-samples RLWE case respectively. Hence for the security analysis of RLWE instance, the available number of samples may lead a big gap for high dimensions.

Table 1. Hardness estimation for restricted number of LWE samples with Regev's parameter settings from LWE-estimator.

n	128		512		1024		2048	
#{given samples}	128	256	512	1024	1024	2048	2048	4096
#{used samples}	128	228	512	919	1024	1853	2048	3821
logarithmic complexity (clock cycles)	66.8	57.7	241.4	201.6	497.3	410.2	1043.8	851.5

1.4 Contribution

In this paper, we introduce an appropriate method to estimate the security of only one RLWE sample. Complexity of various practical attacks on having only one sample and multiple samples are very different. We discuss such differences carefully. We apply the one sample model to construct an ephemeral-only RLWE-based key exchange protocol. Our construction is an ephemeral

RLWE+Rounding variant of the classic Diffie-Hellman key exchange protocol, which can be regarded as a direct drop-in replacement for current widely-deployed Diffie-Hellman key exchange protocol and its variants. We use the new RLWE+Rounding technique instead of RLWR to improve the security of our scheme and reduce communication cost simultaneously. We note that multiple key reuse attacks targeting RLWE-based key exchange protocols do not work for our protocol. Moreover, we study the practical SIS attack on the only one-sample RLWE case. We give secure parameter settings for AES-128/192/256 security levels, which are based on the progressive BKZ simulator as a practical reference and using the sieving-BKZ estimation as a lower bound, taking the impact of exponential memory requirement of sieving subroutine into account. We present protocol specifications, parameter choices, security analysis and performance analysis of our protocol.

Advantages. Here we briefly summarize advantages of our construction as follows: (1) one RLWE sample and flexible parameter choices. Attackers can only use **one** RLWE sample for lattice attacks since our construction is an ephemeral-only key exchange; (2) reduced Communication Cost. Our rounding technique gives at least 10% smaller communication cost compared with similar RLWE-based ones at similar security level; (3) longer Final Shared Keys. Our protocol generates a 512 or 1024 bits key, while most similar works generate 256-bit key. We believe long shared key is extremely important for real-world applications, e.g. the master key in TLS protocol is 384 bits; (4) forward Secure. Our protocol is an ephemeral Diffie-Hellman-like schemes instead of KEM, where in practice, the latter approach reuses public key. If the secret key is leaked, then all previous captured traffic can be decrypted.

2 Ephemeral-Only RLWE+Rounding Key Exchange

2.1 Preliminaries

Let $R_q = \mathbb{Z}_q[x]/f(x)$ be the quotient ring of integer polynomials with $f(x) = x^n + 1$, q a prime number, and n a number as a power of 2. A polynomial $\mathbf{a}$ in R_q is represented as $\mathbf{a} = a_1 + a_2 x + \cdots + a_n x^{n-1}$. Coefficients of a polynomial $\mathbf{a}$ can also denoted by a vector $\mathbf{a} = (a_1, ..., a_n)$.

Let Λ be a discrete subset of $\mathbb{Z}^n$. For any vector $\mathbf{c} \in \mathbb{R}^n$ and any positive parameter $\sigma > 0$, let $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = e^{-\pi\|\mathbf{x}-\mathbf{c}\|^2/\sigma^2}$ be the Gaussian function on $\mathbb{R}^n$ with the center $\mathbf{c}$ and the parameter σ. Denote $\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{x\in\Lambda} \rho_{\sigma,\mathbf{c}(x)}$ be the discrete integral of $\rho_{\sigma,\mathbf{c}}$ over Λ, and $D_{\Lambda,\sigma,\mathbf{c}}$ be the discrete Gaussian distribution over Λ with the center $\mathbf{c}$ and the parameter σ. For all $\mathbf{y} \in \Lambda$, we have $D_{\Lambda,\sigma,\mathbf{c}}(\mathbf{y}) = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(\Lambda)}$. In this paper, we fix Λ to be $\mathbb{Z}^n$ and $\mathbf{c}$ to be zero vector. For ease of notation, we denote $D_{\mathbb{Z}^n,\sigma,0}$ as $D_{\mathbb{Z}^n,\sigma}$. Let $U[a, b]$ be the uniform distribution over discrete set $\{a, a+1, \cdots, b-1, b\}$ over integers. Let $\xleftarrow{\$} \chi$ denote a random sampling according to the distribution χ. Here we represent $\mathbb{Z}_q$ as $\{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$. However, on occasion, we treat elements in $\mathbb{Z}_q$ as elements in $\{0, \cdots, q-1\}$ for convenience, but we will remark the switch clearly.

Let $||\cdot||_1$ be the l_1-norm, $||\cdot||_2$ be the l_2-norm, $||\cdot||_\infty$ be the l_∞-norm. Let $\lfloor x \rfloor$ be the floor function which outputs the greatest integer that is less than or equal to x, $\lceil x \rceil$ be ceiling function which outputs the least integer that is greater than or equal to x, $\lfloor x \rceil$ be the rounding function which rounds x to nearest integer. Let "$a \| b$" denotes the concatenation of a and b. Function log denotes the natural logarithm, $\log_2$ denotes logarithm with base 2.

First we recall and introduce useful lemmas.

Lemma 1 ([34], Lemma 2.5). For $\sigma > 0$, $r \geq 1/\sqrt{2\pi}$, $\Pr[\|\mathbf{x}\|_2 > r\sigma\sqrt{n}; \mathbf{x} \xleftarrow{\$} D_{\mathbb{Z}^n,\sigma}] < (\sqrt{2\pi e r^2} \cdot e^{-\pi r^2})^n$. □

Lemma 2. *For* $\mathbf{a}, \mathbf{b} \in R_q$, $\|\mathbf{a} \cdot \mathbf{b}\|_\infty \leq \|\mathbf{a}\|_2 \cdot \|\mathbf{b}\|_2$.

Proof. Denote the coefficient vector of polynomial $\mathbf{a}(x) = a_1 + a_2 x + a_3 x^2 + \cdots + a_{n-1} x^{n-2} + a_n x^{n-1} \in R_q$ as $(a_1, a_2, a_3, \cdots, a_{n-1}, a_n)$.

For $\mathbf{c} = \mathbf{a} \cdot \mathbf{b} \in R_q$, c_n equals the inner product of $(a_1, a_2, \cdots, a_{n-1}, a_n)$ and $(b_n, b_{n-1}, \cdots, b_2, b_1)$. Similar computations can be applied to coefficients $c_{n-1}, \cdots, c_2, c_1$ as well. By applying Cauchy-Schwarz inequality and property of norm (i.e. for any vector $\mathbf{x}$, $\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_1$), we have $\|\mathbf{c}\|_\infty \leq \|\mathbf{a}\|_2 \cdot \|\mathbf{b}\|_2$. □

2.2 Core Functions

In this section, we define several functions which are crucial to construct our RLWE-based key exchange protocols.

Hint Function. Hint functions $\sigma_0(x)$, $\sigma_1(x)$ from $\mathbb{Z}_q$ to $\{0, 1\}$ are defined as:

$$\sigma_0(x) = \begin{cases} 0, x \in [-\lfloor \frac{q}{4} \rfloor, \lfloor \frac{q}{4} \rfloor] \\ 1, otherwise \end{cases}, \sigma_1(x) = \begin{cases} 0, x \in [-\lfloor \frac{q}{4} \rfloor + 1, \lfloor \frac{q}{4} \rfloor + 1] \\ 1, otherwise \end{cases}$$

Signal Function. A signal function Sig() is defined as:

For any $y \in \mathbb{Z}_q$, $\mathrm{Sig}(y) = \sigma_b(y)$, where $b \xleftarrow{\$} \{0, 1\}$. If $\mathrm{Sig}(y) = 1$, we say y is in the outer region, otherwise y is in the inner region.

Signal function is defined for an integer $x \in \mathbb{Z}_q$. Signal function for $\mathbf{a} \in R_q$ is computed by applying Sig() for each coefficient $a_i \in \mathbb{Z}_q$. In this document, we use the same notation "Sig()" for both signal functions over $\mathbb{Z}_q$ and R_q.

Reconciliation Function. $\mathrm{Mod}_2()$ is a deterministic function with error tolerance δ. $\mathrm{Mod}_2()$ is defined as: for any x in $\mathbb{Z}_q$ and $w = \mathrm{Sig}(x)$, $\mathrm{Mod}_2(x, w) = (x + w \cdot \frac{q-1}{2} \mod q) \mod 2$. Here we treat elements in $\mathbb{Z}_q$ as elements in $\mathbb{Z}$ before we perform the modulo 2 operation.

We define the error tolerance δ, as the largest integer such that for any $x, y \in \mathbb{Z}_q$, if $\|x - y\|_\infty \leq \delta$, then $\mathrm{Mod}_2(x, w) = \mathrm{Mod}_2(y, w)$, where $w = \mathrm{Sig}(y)$. Error tolerance δ is $\frac{q}{4} - 2$, which is the key to ensure correctness of key exchange over RLWE with overwhelming probability.

Reconciliation function is defined for an integer $x \in \mathbb{Z}_q$. The function for $\mathbf{a} \in R_q$ is computed by applying $\mathrm{Mod}_2()$ for each coefficient $a_i \in \mathbb{Z}_q$. We use the same notation "$\mathrm{Mod}_2()$" for reconciliation functions over $\mathbb{Z}_q$ and R_q.

Lemma 3. *Let $q > 8$ be an odd integer. Function $Mod_2()$ as defined above is a robust extractor with respect to signal function $Sig()$ with error tolerance $\delta = \frac{q}{4} - 2$.*

For concrete proofs of Lemma 3, please refer to [21].

Rounding Function. For $x \in \mathbb{Z}_q$, $q > p > 0$ be integers. x is a coefficient of polynomial in R_q, q, p are parameters of our protocol.

For the convenience of notation, we change the representation of $x \in \{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$ to $x \in \{0, \cdots, q-1\}$ before Round() runs. Function Round(x, p, q) is defined in Algorithm 1.

Algorithm 1. Round(x, p, q)

Input: $x \in \mathbb{Z}_q, p, q$
Output: Rounded value x' of x
$t \leftarrow \lfloor 2q/p \rfloor, k \leftarrow \lfloor x/t \rfloor$
if x is odd number **then**
    $x' \leftarrow 2k + 1$
else if x is even number **then**
    $x' \leftarrow 2k$
end if
if $x' = p$ **then**
    $rnd \xleftarrow{\$} U[0, 1]$
    if $rnd = 1$ **then**
        $x' \leftarrow x' - 2$
    else
        $x' \leftarrow (x' + 2) \mod (p + 1)$
    end if
end if

Rounding function is defined for an integer $x \in \mathbb{Z}_q$. Rounding function for $\mathbf{a} \in R_q$ is computed by applying Round() for each coefficient $a_i \in \mathbb{Z}_q$ of $\mathbf{a} \in R_q$. In this document, we use the same notation Round() for both rounding functions over $\mathbb{Z}_q$ and R_q.

Recovering Function. Recover() is a deterministic function. $q > p > 0$ be integers. x' is one coefficient of rounded polynomial, q, p are parameters of our protocol. Function Recover(x', p, q) is defined in Algorithm 2.

Algorithm 2. Recover(x', p, q)

Input: x', p, q
Output: Recovered value x'' of x'
$t \leftarrow \lfloor q/p \rfloor$
if x' is odd number **then**
    $x'' \leftarrow x' \cdot t + 1$
else if x' is even number **then**
    $x'' \leftarrow (x' + 1) \cdot t$
end if

In order to be consistent with theoretical analysis, we change representation of $x'' \in \{0, \cdots, q-1\}$ to $x'' \in \{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$ after Recover() runs.

Recovering function is defined for an integer x'. Recovering function for vector $\mathbf{a}$ is computed by applying Recover() for each coefficient a_i in vector $\mathbf{a}$. In this document, we use the same notation "Recover()" for both recovering functions over integer x' and vector $\mathbf{a}$.

Lemma 4. *For parameter p and q, let $t = \lceil \log_2 q \rceil - \lceil \log_2 p \rceil$, $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ be a vector whose each coefficient is uniformly random sampled integer in $\mathbb{Z}_q$, $\boldsymbol{x}$' be a vector whose each coefficient x'_i = Recover(Round(x_i, p, q),p, q). Let $\boldsymbol{d} = \boldsymbol{x}$-$\boldsymbol{x}$' be a vector whose each coefficient $d_i = x_i - x'_i$ ($i \in [1, n]$). Then d_i is an even number with possible values in set $\{-2^t, -2^t + 2, \cdots, 2^t - 2\}$. $\Pr[d_i = -2^t] = \Pr[d_i = -2^t + 2] = \cdots = \Pr[d_i = 2^t - 2] = \frac{1}{2^t}$* □

Note that our rounding and recovering algorithm is very different from Kyber [15]. Our algorithms round and recover integers with same parity in order to meet the need of our reconciliation mechanism, while Kyber directly rounds and recovers to nearest integer with same or different parity.

A Derivation Function. In each key exchange execution, we use a 128-bit *seed* to generate fresh $\mathbf{a}$. Set *seed* to pseudorandom number generator. Each coefficient $a_i \in \mathbb{Z}_q$ ($i \in [1, n]$) of $\mathbf{a} \in R_q$ is derived as follows:

Algorithm 3. Derive_a(*seed*)

Output: Coefficient a_i of polynomial $\mathbf{a} \in R_q$

1: $a_i \xleftarrow{\$} U[0, q-1]$

2.3 Protocol Specification

In this section, we present our RLWE-based key exchange protocol.

2.3.1 Specification. We give the description of key exchange between party i and party j. In our protocol, users share following parameters: n, σ, q, p. The protocol is illustrated in Fig. 1.

Initiate. Party i instantiates key exchange by generating 128-bit random *seed*, computes fresh $\mathbf{a}$ = Derive_a(*seed*) and public key $\mathbf{p_i} = \mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i} \in R_q$, where $\mathbf{s_i}$ and $\mathbf{e_i}$ are sampled from $D_{\mathbb{Z}^n,\sigma}$. Round $\mathbf{p_i}$ as $\mathbf{p'_i} = \text{Round}(\mathbf{p_i}, p, q)$, send $\mathbf{p'_i}$ and *seed* to party j.

Response. Party j computes fresh $\mathbf{a}$ = Derive_a(*seed*), public key $\mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j} \in R_q$, where $\mathbf{s_j}$ and $\mathbf{e_j}$ are sampled from $D_{\mathbb{Z}^n,\sigma}$. Round $\mathbf{p_j}$ as $\mathbf{p'_j} = \text{Round}(\mathbf{p_j}, p, q)$. Recover public key received from party i as $\mathbf{p''_i} = \text{Recover}(\mathbf{p'_i}, p, q)$. Computes key exchange material $\mathbf{k_j} = \mathbf{p''_i} \cdot \mathbf{s_j} \in R_q$, signal value $w_j = \text{Sig}(\mathbf{k_j})$ and final shared key $sk_j = \text{Mod}_2(\mathbf{k_j}, w_j)$. Send $\mathbf{p'_j}$ and w_j to party i.

Finish. Party i recovers public key received from party j as $\mathbf{p''_j} = \text{Recover}(\mathbf{p'_j}, p, q)$. Compute key exchange material $\mathbf{k_i} = \mathbf{p''_j} \cdot \mathbf{s_i} \in R_q$ and final shared key $sk_i = \text{Mod}_2(\mathbf{k_i}, w_j)$.

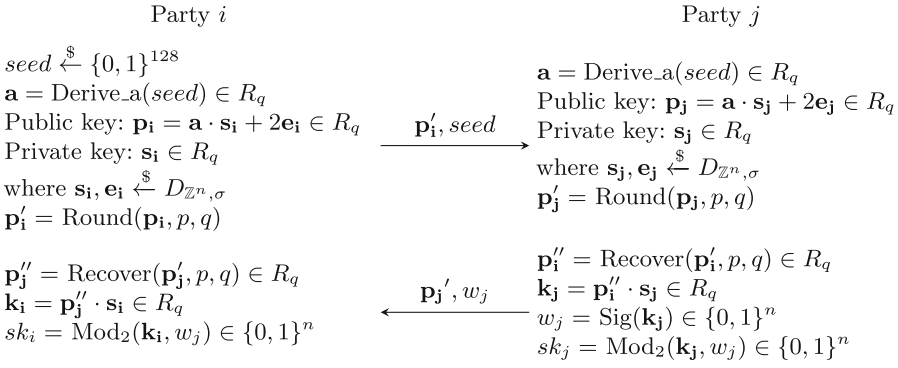

Fig. 1. The proposed RLWE key exchange protocol

2.3.2 Correctness. With above protocol, we have

$$\begin{aligned} \mathbf{k_i} = \mathbf{p_j''}\mathbf{s_i} &= (\mathbf{a}\mathbf{s_j} + \mathbf{2e_j} + \mathbf{d_j})\mathbf{s_i} \\ &= \mathbf{a}\mathbf{s_j}\mathbf{s_i} + 2\mathbf{e_j}\mathbf{s_i} + \mathbf{d_j}\mathbf{s_i} \end{aligned} \tag{1}$$

$$\begin{aligned} \mathbf{k_j} = \mathbf{p_i''}\mathbf{s_j} &= (\mathbf{a}\mathbf{s_i} + 2\mathbf{e_i} + \mathbf{d_i})\mathbf{s_j} \\ &= \mathbf{a}\mathbf{s_i}\mathbf{s_j} + 2\mathbf{e_i}\mathbf{s_j} + \mathbf{d_i}\mathbf{s_j} \end{aligned} \tag{2}$$

$\mathbf{k_i} - \mathbf{k_j} = 2(\mathbf{e_j}\mathbf{s_i} - \mathbf{e_i}\mathbf{s_j}) + (\mathbf{d_j}\mathbf{s_i} - \mathbf{d_i}\mathbf{s_j})$. In order to achieve key exchange with overwhelming success probability, $\|\mathbf{k_i} - \mathbf{k_j}\|_\infty \leq$ error tolerance δ of error reconciliation mechanism, i.e. $\|\mathbf{k_i} - \mathbf{k_j}\|_\infty \leq \frac{q}{4} - 2$. Since the elements in $\mathbf{d}_i$ and $\mathbf{d}_j$ are all even, we have

$$\begin{aligned} \|\mathbf{k_i} - \mathbf{k_j}\|_\infty &= \|2(\mathbf{e_j}\mathbf{s_i} - \mathbf{e_i}\mathbf{s_j}) + (\mathbf{d_j}\mathbf{s_i} - \mathbf{d_i}\mathbf{s_j})\|_\infty \\ &\leq 4\|\mathbf{se}\|_\infty + 2\|\mathbf{d's}\|_\infty \end{aligned} \tag{3}$$

where $\mathbf{s}, \mathbf{e} \in R_q \xleftarrow{\$} D_{\mathbb{Z}^n,\sigma}$. Definition of $\mathbf{d}'$ is consistent with Lemma 4.

With Lemmas 1 and 2, we have $4\|\mathbf{se}\|_\infty \leq 4\|\mathbf{s}\|_2 \cdot \|\mathbf{e}\|_2 \leq 4(r\sigma\sqrt{n})^2 = 4r^2\sigma^2 n$, where $r \geq 1/\sqrt{2\pi}$ is defined in Lemma 1 and n is the degree of polynomial. With Lemma 4, we have $2\|\mathbf{d's}\|_\infty \leq 2\|\mathbf{d}'\|_2 \cdot \|\mathbf{s}\|_2 = 2\|\mathbf{d}'\|_2 \cdot r\sigma\sqrt{n}$. Recall that error tolerance $\delta = \frac{q}{4} - 2$. Therefore as long as $q \geq 4 \cdot [2 + (4r^2\sigma^2 n) + (2\|\mathbf{d}'\|_2 \cdot r\sigma\sqrt{n})]$, key exchange failure probability is estimated to be $(\sqrt{2\pi e r^2} \cdot e^{-\pi r^2})^n$.

2.3.3 Parameter Choice. Parameter choices covering security of AES-128/192/256 are given in Table 2.

Note that for parameter choice $(n, \sigma, q, p) = (1024, 2.6, 120833, 7552)$, it is enough to cover security of AES-128/192/256. We will elaborate this in Sect. 3.4. Modulus $q = 120833$ can instantiate NTT efficiently as $q \equiv 1 \mod 2n$. A failed key exchange implies that at least one bit in sk_i and sk_j mismatches.

Table 2. Our parameter choice

n	σ	q	p	Claimed security level	Failure probability
512	4.19	120833	7552	AES-128	2^{-60}
1024	2.6	120833	7552	AES-192/256	2^{-60}

For Lemma 4 and above parameter choices, let $t = \lceil \log_2 q \rceil - \lceil \log_2 p \rceil$. We have $\Pr[d_i = -2^t] = \Pr[d_i = -2^t + 2] = \cdots = \Pr[d_i = 2^t - 2] = \frac{1}{2^t}$. Therefore, $n = 512, t = 4,\ \|d\|_2 = 32\sqrt{43}$, $n = 1024, t = 4,\ \|d\|_2 = 32\sqrt{86}$.

2.4 Passive Security

We define the passive security of our Diffie-Hellman-like ephemeral-only RLWE-based key exchange protocol in Sect. 2.3. Notations are consistent with Sect. 2.3. We start with the security of our key exchange protocol without rounding and recovering public key. Our proof refers to the methodology in [13]. Then we discuss the hardness of our protocol.

Definition 1. *We say a key exchange protocol is secure under passive adversary, if for any PPT adversary the advantage is negligible.*

Note that even if the secret information is involved in signal function computation, intuitively it is infeasible for adversary $\mathcal{A}$ to recover secret from the binary signal w_j. Thus the signal w_j can not be seen as a RLWE sample from the perspective of both security proof and real attacks (on key exchange itself and RLWE problem) in our setting, i.e. keys from key exchange execution are not reused.

Intuitively, any probabilistic polynomial time (PPT) adversary should not distinguish a real shared key ($sk \in \{0,1\}^n$) from a random one ($rand \xleftarrow{\$} \{0,1\}^n$) even if he gets the transcripts (public key and signal value) of the protocol. We define the advantage of an passive adversary $\mathcal{A}$ as:

$$\mathbf{Adv}_{\mathcal{A}} = \left|\Pr(\mathcal{A}(\mathbf{a}, \mathbf{p_i}, \mathbf{p_j}, w_j, sk) = 1) - \Pr(\mathcal{A}(\mathbf{a}, \mathbf{p_i}, \mathbf{p_j}, w_j, rand) = 1)\right|.$$

Then we want the adversary to distinguish the final shared key $sk \in \{0,1\}^n$ from uniformly random one ($rand \xleftarrow{\$} \{0,1\}^n$) within negligible probability.

Lemma 5. *For any odd $q > 2$, if x is uniformly random in $\mathbb{Z}_q$, then $Mod_2(x, w)$ is uniformly random conditioned on signal $w \in \{0,1\}$.*

Please refer to [21] for concrete proofs of Lemma 5. In addition, we give the following lemma for the security proof of our protocol.

Lemma 6. $\mathbf{k_j}$ *can be seen as a RLWE sample in the security proof, when* $\mathbf{p_i}$ *is computed as RLWE instance.*

Proof. Due to the proposition of multiplication distribution, the multiplication of two Gaussians is itself a Gaussian [16]. In our protocol, both $\mathbf{s}$ and $\mathbf{e}$ are sampled from Gaussian distribution $D_{\mathbb{Z}^n,\sigma}$ with small standard deviation. Hence the secret polynomial can be computed as $\mathbf{k_j} = \mathbf{p_i} \cdot \mathbf{s_j} = \mathbf{a} \cdot \mathbf{s_i} \cdot \mathbf{s_j} + 2\mathbf{s_j} \cdot \mathbf{e_i}$. Since $\mathbf{s_j}$ and $\mathbf{e_i}$ are sampled from $D_{\mathbb{Z}^n,\sigma}$, due to the proposition that the product of two Gaussian PDFs is proportional to Gaussian PDF with a standard deviation that is the square root of half of the denominator, i.e. $\sigma_{\mathbf{s_j}\cdot\mathbf{e_i}} = \sqrt{\frac{\sigma^2_{\mathbf{s_j}} \cdot \sigma^2_{\mathbf{e_i}}}{\sigma^2_{\mathbf{s_j}} + \sigma^2_{\mathbf{e_i}}}} = \frac{\sqrt{2}}{2}\sigma$. If we denote by $\mathbf{a'} = \mathbf{a} \cdot \mathbf{s_i}$, $\mathbf{e'_i} = \mathbf{s_j} \cdot \mathbf{e_i}$, we can get a RLWE instance $(\mathbf{a'}, \mathbf{e'_i})$ with parameters $(n, q, \sigma_{\mathbf{e'_i}})$. Namely, $\mathbf{k_j}$ can be seen as an RLWE instance with parameters $(n, q, \frac{\sqrt{2}}{2}\sigma)$ if $\mathbf{p_i}$ is RLWE itself.

In the following Theorem and its proof, we rewrite $\mathbf{k_j}$ as $\mathbf{k_j} = \mathbf{a'} \cdot \mathbf{s_j} + 2\mathbf{e'_i}$ for the sake of convenience. As discussed above that essentially $\mathbf{k_j}$ can not be used as an RLWE instance in real attack since the secret key can not be recovered from the published signal ω_j. □

Theorem 1. *The construction above is secure against passive PPT adversaries, if the pseudorandom function Derive_a() is secure and the decision RLWE hardness assumption holds.*

Proof. Theorem 1 can be stated in this way: Let n, q, σ be parameters in our proposed key exchange protocol and let $D_{\mathbb{Z}^n,\sigma}$ be the Gaussian distribution defined in Sect. 2.1. If the pseudorandom function $Derive\_a()$ is secure against PPT adversary $\mathcal{B}_0$ and the decision RLWE problem is hard for (n, q, σ), then the key exchange protocol in Fig. 1 guarantees keys indistinguishable from uniform random. Namely,

$$\mathbf{Adv}_{\mathcal{A}} \leq \mathbf{Adv}_{Derive\_a}(\mathcal{B}_0) + \mathbf{Adv}_{n,q,D_{\mathbb{Z}^n,\frac{\sqrt{2}}{2}\sigma}}(\mathcal{A} \circ \mathcal{B}_1) + \mathbf{Adv}_{n,q,D_{\mathbb{Z}^n,\sigma}}(\mathcal{A} \circ \mathcal{B}_2). \quad (4)$$

holds where $\mathcal{B}_1$ and $\mathcal{B}_2$ are assumed PPT adversaries who can distinguish the RLWE samples from uniform random.

We prove the security by a sequence of games and lemmas. Let S_i be the challenge where the adversary guesses b in game i. The first game $\mathbf{Game}_0$ is the real game which the adversary gets all of the original published information, while in the last game $\mathbf{Game}_4$ the adversary gets uniformly random parameters without RLWE information. We show that the views of $\mathbf{Game}_0$ and $\mathbf{Game}_4$ are computational indistinguishable for any PPT adversaries, under the decision RLWE hardness assumption.

$\mathbf{Game}_0$. This is the original game between the protocol challenger and the passive adversary $\mathcal{A}$. That is, the adversary obtains $\mathbf{a}, \mathbf{p_i}, \mathbf{p_j}, w_j, \mathbf{k}_b$, where $\mathbf{p_i} = \mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i}$ and $\mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j}$. Then $\mathcal{A}$ outputs a guess b'. Note that there are three RLWE pairs in $\mathbf{Game}_0$: $(\mathbf{a}, \mathbf{p_i})$ with secret vector $\mathbf{s_i}$, and $(\mathbf{a}, \mathbf{p_j})$ and $(\mathbf{a'}, \mathbf{k_j})$ both with secret vector $\mathbf{s_j}$. Here we can rewrite $\mathbf{Adv}_{\mathcal{A}}$ as

$$\mathbf{Adv}_{\mathcal{A}} = |\Pr(S_0) - 1/2|\,. \quad (5)$$

Game$_1$. This game is identical to **Game**$_0$ except that instead of generating $\mathbf{a}$ pseudorandomly from *seed* using function Derive_a(), the challenger samples $\mathbf{a}$ uniformly at random.

Lemma 7. *Any PPT passive adversary cannot distinguish* ***Game***$_0$ *and* ***Game***$_1$*, if the assumption holds that the pseudorandom function Derive_a() is secure.*

Proof. It is obvious that the two games are indistinguishable under our assumption that the pseudorandom function Derive_a() is secure, i.e.

$$|\Pr(S_0) - \Pr(S_1)| \leq \mathbf{Adv}_{Derive\_a}(\mathcal{B}_0). \tag{6}$$

□

Game$_2$. This game is identical to **Game**$_1$ except that instead of setting $\mathbf{p_i} = \mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i}$, the challenger sets $\mathbf{p_i} = \mathbf{r_i}$, where $\mathbf{r_i} \xleftarrow{\$} R_q$.

Lemma 8. *Any PPT passive adversary cannot distinguish* ***Game***$_1$ *and* ***Game***$_2$*, if the decision RLWE assumption holds.*

That is to say, in **Game**$_1$, the challenge $(\mathbf{a}, \mathbf{p_i})$ is sampled honestly from RLWE oracle. In **Game**$_2$, $(\mathbf{a}, \mathbf{p_i})$ is uniformly sampled from $R_q \times R_q$ at random. These two distributions are computationally indistinguishable under the assumption that the decision RLWE problem is hard for parameter set (n, q, σ).

Proof. We prove the lemma by showing that if there exists an adversary $\mathcal{A}$ who can distinguish **Game**$_1$ and **Game**$_2$, then we can construct another adversary $\mathcal{B}_1$ to distinguish the RLWE samples from uniform random. $\mathcal{B}_1$ works as follows. Once obtaining challenges $(\mathbf{a}, \mathbf{b_i}) \in R_q \times R_q$ from the RLWE oracle, where $\mathbf{b_i}$ is either $\mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i}$ or random $\mathbf{r_i}$ in R_q, $\mathcal{B}_1$ samples $\mathbf{s_j} \xleftarrow{\$} D_{\mathbb{Z}^n,\sigma}$ and sets $\mathbf{k_j} = \mathbf{b_i} \cdot \mathbf{s_j}$. $\mathcal{B}_1$ also computes $\mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j}$. Finally $\mathcal{B}_1$ sends $(\mathbf{a}, \mathbf{p_i} = \mathbf{b_i}, \mathbf{p_j}, w_j, \mathbf{k_j})$ to $\mathcal{A}$. $\mathcal{B}_1$ outputs whatever $\mathcal{A}$ outputs. We note that $\mathcal{B}_1$ can compute w_j by himself. If $\mathbf{b_i}$ is an RLWE sample, then what $\mathcal{A}$ obtains are exactly the same as in **Game**$_1$, if $\mathbf{b_i}$ is uniformly random in R_q, then what $\mathcal{A}$ obtains are exactly the same as in **Game**$_2$. This implies that if $\mathcal{A}$ can distinguish **Game**$_1$ and **Game**$_2$ with noticeable advantage, then $\mathcal{B}$ can distinguish RLWE samples from uniformly random with the same advantage. Simultaneously, the adversary $\mathcal{B}_1$ sets $\mathbf{k_j} = \mathbf{u_j}$ by $\mathbf{u_j} = \mathbf{b_i} \cdot \mathbf{s_j}$, where $\mathbf{b_i}$ is either sampled from RLWE or uniformly random $\mathbf{r_i}$ in R_q.

Intuitively it leads to $\mathbf{u_j} = \mathbf{b_i} \cdot \mathbf{s_j}$ is RLWE or uniformly random $\mathbf{r_j}$, according to the analysis under Lemma 6. Hence indeed what $\mathcal{B}_1$ sends to $\mathcal{A}$ can be rewritten as $(\mathbf{a}, \mathbf{p_i} = \mathbf{b_i}, \mathbf{p_j}, w_j, \mathbf{k_j} = \mathbf{u_j})$ to $\mathcal{A}$. Thus we have two RLWE samples $(\mathbf{a}, \mathbf{p_i} = \mathbf{b_i})$ and $(\mathbf{a}', \mathbf{k_j} = \mathbf{u_j})$ in **Game**$_2$ where under the RLWE assumption, RLWE sample $(\mathbf{a}, \mathbf{p_i})$ with $D_{\mathbb{Z}^n,\sigma}$ is indistinguishable with random sample $(\mathbf{a}, \mathbf{r_i})$, and RLWE sample $(\mathbf{a}', \mathbf{k_j})$ with $D_{\mathbb{Z}^n,\frac{\sqrt{2}}{2}\sigma}$ is indistinguishable with $(\mathbf{p_i}, \mathbf{r_j})$ respectively. This finishes the proof and simultaneously we can get the following inequation:

$$|\Pr(S_1) - \Pr(S_2)| \leq \mathbf{Adv}_{n,q,D_{\mathbb{Z}^n,\frac{\sqrt{2}}{2}\sigma}}(\mathcal{A} \circ \mathcal{B}_1). \tag{7}$$

□

Game$_3$. This game is identical to **Game**$_2$ except that instead of setting $\mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j}$, the challenger sets $\mathbf{p_j} = \mathbf{r_j}$, where $\mathbf{r_j} \xleftarrow{\$} R_q$.

Lemma 9. *Any PPT passive adversary cannot distinguish* ***Game***$_2$ *and* ***Game***$_3$*, if the decision RLWE assumption holds.*

Proof. The proof for Lemma 9 is analogous to the proof for Lemma 8, i.e. we should show if there exists an adversary $\mathcal{A}$ who can distinguish **Game**$_2$ and **Game**$_3$, then we can construct another adversary $\mathcal{B}_2$ to distinguish the RLWE samples from uniform random. $\mathcal{B}_2$ works as follows. Once obtaining challenges $(\mathbf{a}, \mathbf{b_j})$ where $\mathbf{b_j}$ is either RLWE instance or random $\mathbf{r_j}$ in R_q, $\mathcal{B}_2$ sets $\mathbf{p_j} = \mathbf{b_j}$. Finally $\mathcal{B}_2$ sends $(\mathbf{a}, \mathbf{p_i}, \mathbf{p_j} = \mathbf{b_j}, w_j, \mathbf{k_j})$ to $\mathcal{A}$. $\mathcal{B}_2$ outputs whatever $\mathcal{A}$ outputs. Hence we have

$$|\Pr(S_2) - \Pr(S_3)| \leq \mathbf{Adv}_{n,q,D_{\mathbb{Z}^n,\sigma}}(\mathcal{A} \circ \mathcal{B}_2). \tag{8}$$

□

Furthermore, in **Game**$_3$, the adversary is given $(\mathbf{a}, \mathbf{b_j})$ which is either sampled uniformly at random. However, when $\mathbf{p_i}$ is uniformly sampled, elements of $\mathbf{k_j} = \mathbf{p_i} \cdot \mathbf{s_j} \in R_q$ is also uniformly distributed. Thus the elements in w_j can be seen uniformly distributed owing to the construction of signal function. Due to Lemma 5, sk_j computed by the reconciliation function from $\mathbf{k_j}$ and w_j is also distributed uniformly at random. Namely, there is no RLWE information in **Game**$_3$ so the adversary can not distinguish the key is generated from the key exchange protocol or just uniformly sampled. Hence, we have the following equation in **Game**$_3$.

$$|\Pr(S_3)| = 1/2. \tag{9}$$

Consequently, we can get in Eq. (4) (and finish the proof for Theorem 1) by combining the formulas from (5) to (9). □

Now we deal with the security regarding to rounding and recovering $\mathbf{a} \cdot \mathbf{s} + 2\mathbf{e}$ in the next lemma.

Lemma 10. *For following two key exchange protocols:*

1. $\mathbf{p_i} = \mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i}, \mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j}, \mathbf{k_i} = \mathbf{p_j} \cdot \mathbf{s_i}, \mathbf{k_j} = \mathbf{p_i} \cdot \mathbf{s_j}, w_j = \mathit{Sig}(\mathbf{k_j}), sk_i = \mathit{Mod}_2(\mathbf{k_i}, w_j), sk_j = \mathit{Mod}_2(\mathbf{k_j}, w_j)$
2. $\mathbf{p_i} = \mathbf{a} \cdot \mathbf{s_i} + 2\mathbf{e_i}, \mathbf{p_j} = \mathbf{a} \cdot \mathbf{s_j} + 2\mathbf{e_j}, \mathbf{p'_i} = \mathit{Round}(\mathbf{p_i}, p, q), \mathbf{p''_i} = \mathit{Recover}(\mathbf{p'_i}, p, q), \mathbf{p'_j} = \mathit{Round}(\mathbf{p_j}, p, q), \mathbf{p''_j} = \mathit{Recover}(\mathbf{p'_j}, p, q), \mathbf{k_i} = \mathbf{p''_j} \cdot \mathbf{s_i}, \mathbf{k_j} = \mathbf{p''_i} \cdot \mathbf{s_j}, w_j = \mathit{Sig}(\mathbf{k_j}), sk_i = \mathit{Mod}_2(\mathbf{k_i}, w_j), sk_j = \mathit{Mod}_2(\mathbf{k_j}, w_j)$

The hardness of computing final shared key of second protocol is at least as hard as computing final shared key of first protocol.

Proof. With publicly known algorithm Round() and Recover(), publicly known parameters and public terms $\mathbf{p'_i}, \mathbf{p'_j}$, any adversary can compute $\mathbf{p''_i} \approx \mathbf{p_i}$ and $\mathbf{p''_j} \approx \mathbf{p_j}$. However, $\mathbf{p''_i} \neq \mathbf{p_i}$, $\mathbf{p''_j} \neq \mathbf{p_j}$, Round() and Recover() function generate additional errors, which makes recovering private key $\mathbf{s_i}$ or $\mathbf{s_j}$ using transcripts from our key exchange at least no easier than using $\mathbf{p_i}, \mathbf{p_j}$ or $\mathbf{k_i}, \mathbf{k_j}$ to solve RLWE problem. □

3 Estimating Security of One RLWE Sample

Before showing the analysis in this section, we stress that only ONE sample can be used in the real attack for our ephemeral protocol settings. It is infeasible for attacker to recover secret key or apply lattice attacks given the binary signal vector from our ephemeral key exchange. Hence the signal w can not be used as another RLWE sample in the real attack (and also the security proof in Sect. 2.4).

3.1 Prerequisites

Lattice Theory. A *lattice* L is defined as an infinite space expanded by *basis* $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, where $\mathbf{b}_i$ $(i = 1, \ldots, n)$ are a set of linearly independent vectors in $\mathbb{R}^m$. Here n is the dimension of L. The n-dimensional *volume* of L is denoted by $\mathrm{Vol}(L)$, which is computed by the *determinant* of basis B, i.e. $\mathrm{Vol}(L) = \det(B)$. We denote $V_n(R) = R^n \cdot \frac{\pi^{n/2}}{\Gamma(n/2+1)}$ as the volume of n-dimensional Euclidean ball of radius R.

Ring LWE (RLWE) Problem. Let $m \geq 1$ be a power of 2 and $q \geq 2$ be an integer, let $R_q = \mathbb{Z}_q[x]/\Phi_m(x)$, where $\Phi_m(x) = x^n + 1$ is the m-th cyclotomic polynomial with $n = m/2$. Let χ be a β-bounded distribution. For secret polynomial $\mathbf{s} \xleftarrow{\$} \chi$ and error polynomial $\mathbf{e} \xleftarrow{\$} \chi$, choosing $\mathbf{a} \in R_q$ uniformly random, output $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e} \in R_q)$. Search version of RLWE problem is: for $\mathbf{s} \xleftarrow{\$} \chi$, given $poly(n)$ number of samples of $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}) \in (R_q, R_q)$, find $\mathbf{s}$ (and $\mathbf{e}$ simultaneously).

Proposition. Let $\mathbf{z} = \mathrm{Recover}(\mathrm{Round}(\mathbf{a} \cdot \mathbf{s} + \mathbf{2e}, p, q), p, q) = \mathbf{as} + \mathbf{2e} + \mathbf{d} = \mathbf{as} + \mathbf{2f} \in R_q$, where $\mathbf{s}, \mathbf{e} \xleftarrow{\$} D_{\mathbb{Z}^n, \sigma}$ and $2\mathbf{f} = 2\mathbf{e} + \mathbf{d}$ (elements in $\mathbf{d}$ are even). Hence we can regard $\mathbf{f}$ as error term $\mathbf{e}$ in the definition of RLWE above. The attack on our protocol is given $\mathbf{z}$ and $\mathbf{a}$, output private key $\mathbf{s}$. This problem is equivalent to:

$$\begin{aligned} & \mathbf{z} = \mathbf{a} \cdot \mathbf{s} + 2\mathbf{f} \mod q \\ \Leftrightarrow\ & 2^{-1}\mathbf{z} = 2^{-1}\mathbf{a} \cdot \mathbf{s} + \mathbf{f} \mod q \\ \Leftrightarrow\ & \mathbf{z}'' = \mathbf{a}'' \cdot \mathbf{s} + \mathbf{f} \mod q \end{aligned}$$

Standard deviation of term $\mathbf{f}$ is denoted as σ_f. Note that σ_f is different from σ notation in Sect. 2.1 as f no longer follows discrete Gaussian distribution (histogram shows similar shape as Gaussian distribution), therefore σ_f is computed as the square root of variance.

Shortest Vector Problem. Given an input basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ of a lattice L, *Shortest Vector Problem* (SVP) is to find a non-zero shortest vector in L. We introduce the following two variants of the SVP to be used in this section.

Short Integer Solution Problem. Given an integer q and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, *Short Integer Solution problem* (SIS) is to compute a short vector $\mathbf{y} \in \mathcal{B}$ s.t. $\mathbf{Ay} \equiv \mathbf{0} \mod q$, where $\mathcal{B}$ is a set of short vectors with some norm bound.

Unique Shortest Vector Problem. *Unique SVP problem* (uSVP) is for a given lattice L which satisfies $\lambda_1(L) \ll \lambda_2(L)$, find the shortest vector in L. Here $\lambda_i(L)$ means the length of i-th linear independent shortest vector for $i = 1, 2$.

Root Hermite Factor. To evaluate the performance of lattice algorithms for solving SVP, we use the *root Hermite Factor* (rHF) defined in [22] as:

$$\delta = \mathrm{rHF}(\mathbf{b}_1, \ldots, \mathbf{b}_n) = (\|\mathbf{b}_1\|_2 / \mathrm{Vol}(L)^{1/n})^{1/n}.$$

Geometric Series Assumption (GSA). The *Geometric Series Assumption* [31] indicates the quality of an LLL-type reduced basis. It says l_2 norm of GSO vectors $\|\mathbf{b}_i^*\|$ in the reduced basis decrease geometrically with a constant r as $\|\mathbf{b}_i^*\|_2{}^2 / \|\mathbf{b}_1\|_2{}^2 = r^{i-1}$ $(i = 1, \ldots, n$ and $r \in [3/4, 1))$.

Lattice Algorithms. There are some lattice algorithms such as BKZ and sieving to solve SVP and its variants. *BKZ algorithm* was originally proposed in [30], which computes basis that are almost β-reduced, namely the projected lengths of each basis vectors are the shortest ones in the relative β-sized local blocks. BKZ algorithm runs in exponential time and there are some efficient improvements for BKZ algorithms [18,38]. In 2016, Aono et al. proposed a precise simulator to estimate runtime of progressive BKZ algorithm (pBKZ), which processes given basis by increasing block size with some strategy [8]. When dimension n is large $(n \geq 100)$, runtime $\mathrm{TimeBKZ}(n, \beta_t)$ of pBKZ with target blocksize β_t is estimated by Eq. (18) in [8]. Further details may be found in [8] and a reference implementation is freely available at [9].

In 2001, Ajtai et al. proposed a sieving algorithm to solve SVP, which requires a runtime of $2^{0.52n+o(n)}$ in n dimension lattice and simultaneously requires exponential storage of $2^{0.2n+o(n)}$ [1]. According to recent research results, for a n-dimensional lattice L and fixed blocksize β in BKZ, the runtime of sieving algorithm can be estimated in $2^{0.292\beta+o(\beta)}$ clock cycles for a β-dimensional subroutine [4], and totally BKZ-β costs $8n \cdot 2^{0.292\beta+16.4}$ operations [12]. The phase transition of time cost and memory cost is considered in our work. Namely, we assume that practically the exponential large memory $(\beta \cdot 2^{0.292*\beta+o(\beta)})$ cost of sieve will increase the computation cost by at least one magnitude (x10).

3.2 Algorithms for Solving RLWE

In this work we use the adapted SIS attack algorithm on solving RLWE, which is an adaptation of the dual-embedding method mentioned in [29] and [10]. Note that the SIS attack with "rescaling technique" is also called as "Bai-Galbraith's" embedding attack in [3], which is developed by Bai and Galbraith to improve the attack on binary LWE in [10]. In the adapted SIS algorithm, we do not introduce the rescaling technique but just enlarge the lattice dimension. There are also some analysis to the embedding attack and its variants in previous articles [35–37,39].

3.3 Significance of Number of Samples in Practical Attack

At first we claim that because of the setting of our key exchange protocol: only one RLWE instance $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e} \bmod q) \in (R_q, R_q)$ is given, Kannan's embedding technique [23] and Liu-Nguyen's decoding attack [24] cannot be adopted since the lattice $L_{(\mathbf{A},q)} = \{\mathbf{v} \in \mathbb{Z}_q^m \mid \mathbf{v} \equiv \mathbf{A}\mathbf{x} \pmod q, \mathbf{x} \in D_\sigma^n\}$ is trivial when $m \leq n$. Therefore our estimator should be different from some other key exchange schemes as NewHope [6] and Albrecht's estimator [3] etc. which regard RLWE and normal LWE problem as having the same difficulty without considering the number of available RLWE samples. From the discussion in Sect. 1.3 and the estimations in Table 1, we observe that there is a big gap of hardness estimations between ONE-sample RLWE and multiple-samples RLWE. Note that indeed the lowest number m of required LWE samples from the 2016 estimate (Section 3.4) are as follows: m is 576 for $(n, q, \sigma_f) = (512, 120833, 4.92)$ and m is 1097 for $(n, q, \sigma_f) = (1024, 120833, 4.72)$. Therefore, the optimal number in the 2016 estimate can be obtained only from "more than one RLWE samples" ($m = 576 > n = 512$ and $m = 1097 > n = 1024$). Hence in practical attack, we can get only one n-dimensional RLWE instance, which can be amplified to $2n + 1$ without changing the distribution of error vectors. Therefore the lattice dimension of solving RLWE in our case is $d = 2n + 1$.

3.4 Our Simulator

At AsiaCrypt 2017 [3], Albrecht et al. re-estimated the hardness of LWE problem using Kannan's embedding and Bai-Gal's embedding respectively under estimation in NewHope [6] (denoted as "2016 estimate"). 2016 estimate states that if the Gaussian Heuristic and the GSA [31] hold for BKZ-β reduced basis and

$$\sqrt{\beta/d} \cdot \|(\mathbf{e}|1)\|_2 \approx \sqrt{\beta}\sigma \leq \delta^{2\beta-d} \cdot \mathrm{Vol}(L_{(\mathbf{A},q)})^{1/d}. \tag{10}$$

Then error $\mathbf{e}$ can be found by BKZ-β with root Hermite Factor δ. In our case, we assume $\mathbf{f}$ is the Gaussian distributed error vector plus the uniformly distributed perturbation sampled from a bounded set due to Rounding-Recovering functions and $(\mathbf{s}|\mathbf{f}|1)$ is the target vector in our attack. So there is a gap between the distribution of $\mathbf{f}$ and the Gaussian distribution. However, given a same standard deviation σ_f, the expected length of vectors sampled from the hybrid distribution is bigger than the one sampled from Gaussian distribution on average, by a simple computation using the center limit theorem. Hence in our estimation we assume $\mathbf{f}$ is Gaussian distributed. We adapt the left side of the inequality (10) as $\sqrt{\beta/d} \cdot \|(\mathbf{s}|\mathbf{f}|1)\|_2 \approx \sqrt{\beta \cdot (\sigma_e{}^2 + \sigma_f{}^2)}$. For BKZ reduction runtime estimation, we will give the result of progressive BKZ and Albrecht's BKZ with sieving estimator.

Step 1. A short vector $\|\mathbf{b}_1\|_2 = \delta^d \cdot \det(\mathbf{B})$ is assumed to be inside of the BKZ-β reduced basis $\mathbf{B}$ of dimension d [17], where the root Hermite Factor is

$$\delta = (((\pi\beta)^{1/\beta}\beta)/(2\pi e))^{\frac{1}{2(\beta-1)}}. \tag{11}$$

Since σ_f can be experimentally derived from σ_e, we can compute lower bound of σ_f in RLWE(n, q, σ_f) which covers security of AES-128/192/256 using Eqs. (11) and (12). Note that f no longer follows discrete Gaussian distribution (histogram shows similar shape as Gaussian distribution). Therefore we take a heuristic approach to estimate σ_f.

In our case, $d = 2n + 1$ is the dimension of lattice and also $\text{Vol}(L_{(\mathbf{A},q)}) = q^n$. Therefore we can pre-compute the expected root Hermite factor δ for $\beta = 10, \cdots, n$ and adapt inequality (10) to

$$\sqrt{\beta \cdot ({\sigma_e}^2 + {\sigma_f}^2)} \leq \delta^{2\beta - 2n - 1} \cdot q^{n/(2n+1)}. \tag{12}$$

To compute the target β in the progressive BKZ simulator, we use the correspondence between δ and the GSA constant r: Given a d-dimensional basis, in order to use progressive BKZ simulator, we need target β_t for our parameter choice. At this stage, we can get the target GSA constant $r_t = \delta^{-4d/(d-1)}$. Therefore we can compute the terminating blocksize β_t in progressive BKZ corresponding to r_t by equations (10) and (11) given in [8].

Step 2. We compute the complexity of BKZ-β with sieving SVP oracle estimated as $8d \cdot 2^{0.292\beta + 16.4}$ double precision floating point operations [2,12]. we translate this to complexity of bit unit by $T_{sieving-BKZ} = 8d \cdot 2^{0.292\beta+16.4} \cdot 64$ (bits).

Simultaneously, T_{BKZ} can also be replaced by progressive BKZ simulator explained in Sect. 3.1. We run the progressive BKZ simulator for both $n = 512$ and $n = 1024$ cases. Considering the number of iterations for each fixed blocksize in BKZ, we get following two fitting functions to estimate the runtime of two cases respectively.

$$\log_2(Time_{pBKZ}(secs)) = \begin{cases} 0.003924 \cdot \beta^2 - 0.568 \cdot \beta + 41.93 & (n = 512) \\ 0.004212 \cdot \beta^2 - 0.6886 \cdot \beta + 55.49 & (n = 1024) \end{cases} \tag{13}$$

Then we compute the complexity of bit unit by $T_{pBKZ} = Time_{pBKZ} \times 2.7 \times 10^9 \times 64$ (bits). on our Intel(R) Xeon(R) CPU E5-2697 v2 @ 2.70GHz server.

We generate 1,000 and 2,000 $\mathbf{as} + 2\mathbf{e}$ samples for parameter choice $(n, \sigma, q, p) = (1024, 2.6, 120833, 7552)$ and $(n, \sigma, q, p) = (512, 4.19, 120833, 7552)$. For each sample, we apply Round() and Recover() functions, giving us

$$\mathbf{z} = \text{Recover}(\text{Round}(\mathbf{a} \cdot \mathbf{s} + 2\mathbf{e}, p, q), p, q) = \mathbf{a} \cdot \mathbf{s} + 2\mathbf{f}.$$

With $\frac{\mathbf{z}-\mathbf{as}}{2} = \mathbf{f}$, we compute standard deviation σ_f. Results are given in Table 3, where the parameter settings can ensure 2^{-60} failure probability.

Due to the uncertainty simulation for runtime with large dimension and large β (>1000 and >200 respectively), we are not sure about the simulation results for our key exchange protocol. We will leave it as future work. However, our parameter choices can cover results from pBKZ simulator. Therefore we show results from pBKZ simulator in Table 3 as well.

The 2016 estimate for AES-128 and AES-192/256 security gives 142.27 and 279.05 bit operations respectively. Practically the exponential memory's access

Table 3. Our simulation data and parameter settings covering security of AES-128/192/256

Security level (n, q, σ)	AES-128 (512,120833,4.19)		AES-192 and AES-256 (1024,120833,2.6)	
Method	pBKZ	2016 estimate	pBKZ	2016 estimate
Logarithmic computational complexity	319.14	142.27	1473.09	279.05
Blocksize	330	366	660	831
GSA Const.	0.983		0.991	
σ (for $\mathbf{s}$ and $\mathbf{e}$) of our parameter choice	4.19		2.6	
σ_f	4.92		4.72	

of sieving algorithm will increase the computation cost by at least one magnitude (x10), therefore we conclude that our parameter choices with $n = 512$ can achieve at least 145.59 bits security, $n = 1024$ can achieve at least 282.37 bits security. With results given in Table 3, we claim that parameter choices given in Table 2 cover security of AES-128/192/256.

Furthermore, Aono et al. proposed a method to compute the lower bound N on the cost of extreme-pruning enumeration algorithm, for a certain pruning success probability α' to find a shortest vector, which bases on a simulated HKZ-reduced basis [7]. We use the formula (17) in [7] and set $\alpha' = 1$. Analogous to the sieving-BKZ model in 2016 estimate, we compute the complexity of BKZ with enumeration subroutine by $T_{enum-BKZ} = 8d \cdot N$ (bits). When we compute $T_{enum-BKZ}$ using the blocksizes 366 and 831 given in Table 3, we get 194 bit security and 596 bit security respectively. It means that our parameter settings are safe under the BKZ with enumeration subroutine model.

4 Implementation and Performance

In this section, we introduce our implementation and performance of our key exchange scheme in Sect. 2.3. Note that in our implementation, a number in $\mathbb{Z}_q$ is represented as $[0, q-1]$. One can convert the regions defined for hint and signal functions from $\{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$ to corresponding regions in $[0, q-1]$.

We use Victor Shoup's NTL library [33] in our implementation, where the fast Number Theoretic Transformation (NTT) technique with adapted butterfly operation is applied, for doing polynomial operations as multiplication, division, GCD, factoring and so on. Simultaneously, we use the Discrete Gaussian Sampler (DGS) based on Cumulative Distribution Table (CDT) in [26].

4.1 Experimental Results

Our implementation uses C++ language. We run 100,000 times experiments for each parameter choice on a computer with Intel Xeon E5-2697 v2 @ 2.70 GHz

Table 4. Runtime (millisecond) of our implementation

Security level	TimeDGS	TimePM	TimeKeyPair	TimeP_i	TimeP_j
AES-128	0.05	0.40	0.48	0.92	0.74
AES-192/256	0.09	0.83	1.00	1.90	1.55

CPU, running CentOS Linux release 7.4.1708, g++ version 6.3.0. We evaluate the average runtime for discrete Gaussian sampling (TimeDGS), polynomial multiplication (TimePM), key generation (TimeKeyPair), party i timing (TimeP_i) and party j timing (TimeP_j) respectively. We show the experimental results in Table 4 with two decimal precision.

Rounding, recovering and error reconciliation are extremely efficient. Most expensive ones are discrete Gaussian sampling and polynomial multiplication.

4.2 Communication Cost Comparison

We show the communication cost of our work with several similar RLWE-based key exchange or KEM protocols in Table 5. Our construction has smallest communication cost compared with rest of the RLWE-based protocols. Thus, we believe that our construction provides better trade-off between security and communication cost.

Table 5. Communication cost comparison between several Diffie-Hellman-like key exchange and KEM constructions from RLWE problem

Name	Type	n	q	Claimed security	Public key (Bytes)	Total (Bytes)
This work	DH	512	120833	AES-128 145-bit	832	1744
	DH	1024	120833	AES-192/256 282-bit	1664	3472
BCNS [14]	DH	1024	$2^{32}-1$	128-bit	4096	8320
NewHope [6]	DH	1024	12289	281-bit	1792	3872
NewHope-Simple [5]	KEM	1024	12289	281-bit	1792	4000
HILA5 [28]	KEM	1024	12289	255-bit	1792	3836

5 Conclusion

It is crucial to build secure and practical post-quantum cryptography primitives for the upcoming post-quantum world. We believe that our new ephemeral-only Diffie-Hellman-like RLWE+Rounding key exchange gives a new solution. We also apply a proper approach to estimate the security of only one RLWE sample, which is closely related to our key exchange protocol design. We also take the overwhelming memory requirement of sieving algorithm into consideration. Our elegant and simple design gives better security and smaller communication cost.

Acknowledgement. Jintai Ding is partially supported by NSF grant DMS-1565748 and US Air Force grant FA2386-17-1-4067. Tsuyoshi Takagi and Yuntao Wang are supported by JST CREST Grant Number JPMJCR14D6 and JSPS KAKENHI Grant Number JP17J01987, Japan. Xinwei Gao is supported by China Scholarship Council.

References

1. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing, STOC 2001, pp. 601–610 (2001)
2. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 103–129. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_4
3. Albrecht, M.R., Göpfert, F., Virdia, F., Wunderer, T.: Revisiting the expected cost of solving uSVP and applications to LWE. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 297–322. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_11
4. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Math. Cryptol. **9**(3), 169–203 (2015)
5. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: NewHope without reconciliation. IACR Cryptology ePrint Archive 2016, 1157 (2016). http://eprint.iacr.org/2016/1157
6. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange-a new hope. In: USENIX Security Symposium, pp. 327–343 (2016)
7. Aono, Y., Nguyen, P.Q., Seito, T., Shikata, J.: Lower bounds on lattice enumeration with extreme pruning. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 608–637. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_21
8. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 789–819. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_30
9. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: The progressive BKZ code (2017). http://www2.nict.go.jp/security/pbkzcode/
10. Bai, S., Galbraith, S.D.: Lattice decoding attacks on binary LWE. In: Susilo, W., Mu, Y. (eds.) ACISP 2014. LNCS, vol. 8544, pp. 322–337. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08344-5_21
11. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_42
12. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, pp. 10–24 (2016)
13. Bos, J., et al.: Frodo: take off the ring! practical, quantum-secure key exchange from LWE. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 1006–1018. ACM (2016)

14. Bos, J.W., Costello, C., Naehrig, M., Stebila, D.: Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In: 2015 IEEE Symposium on Security and Privacy (SP), pp. 553–570. IEEE (2015)
15. Bos, J.W., et al.: CRYSTALS - kyber: a CCA-secure module-lattice-based KEM. IACR Cryptology ePrint Archive 2017, 634 (2017). http://eprint.iacr.org/2017/634
16. Bromiley, P.A.: Products and convolutions of Gaussian distributions, vol. 3 (2003)
17. Chen, Y.: Lattice reduction and concrete security of fully homomorphic encryption. Dept. Informatique, ENS, Paris, France, Ph.D. thesis (2013)
18. Chen, Y., Nguyen, P.Q.: BKZ 2.0: better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_1
19. Computer Security Division, Information Technology Laboratory, N.I.O.S., Technology, U.D.O.C.: Post-quantum cryptography—CSRC (2017). https://csrc.nist.gov/projects/post-quantum-cryptography
20. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
21. Ding, J., Xie, X., Lin, X.: A simple provably secure key exchange scheme based on the learning with errors problem. IACR Cryptology ePrint Archive 2012, 688 (2012). http://eprint.iacr.org/2012/688
22. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_3
23. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**(3), 415–440 (1987)
24. Liu, M., Nguyen, P.Q.: Solving BDD by enumeration: an update. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 293–309. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36095-4_19
25. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
26. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_5
27. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM (JACM) **56**(6), 34 (2009)
28. Saarinen, M.-J.O.: HILA5: On reliability, reconciliation, and error correction for ring-LWE encryption. In: Adams, C., Camenisch, J. (eds.) SAC 2017. LNCS, vol. 10719, pp. 192–212. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-72565-9_10
29. Schmidt, M., Bindel, N.: Estimation of the hardness of the learning with errors problem with a restricted number of samples. IACR Cryptology ePrint Archive 2017, 140 (2017). http://eprint.iacr.org/2017/140
30. Schnorr, C.P., Euchner, M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. Math. Program. **66**(1), 181–199 (1994)
31. Schnorr, C.P.: Lattice reduction by random sampling and birthday methods. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 145–156. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36494-3_14
32. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev. **41**(2), 303–332 (1999)

33. Shoup, V.: NTL, a library for doing number theory (2017). http://www.shoup.net/ntl/
34. Stephens-Davidowitz, N.: Discrete Gaussian sampling reduces to CVP and SVP. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1748–1764. Society for Industrial and Applied Mathematics (2016)
35. Wang, W., Wang, Y., Takayasu, A., Takagi, T.: Estimated cost for solving generalized learning with errors problem via embedding techniques. In: Inomata, A., Yasuda, K. (eds.) IWSEC 2018. LNCS, vol. 11049, pp. 87–103. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-97916-8_6
36. Wang, Y., Aono, Y., Takagi, T.: An experimental study of Kannan's embedding technique for the search LWE problem. In: Qing, S., Mitchell, C., Chen, L., Liu, D. (eds.) ICICS 2017. LNCS, vol. 10631, pp. 541–553. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-89500-0_47
37. Wang, Y., Aono, Y., Takagi, T.: Hardness evaluation for search LWE problem using progressive BKZ simulator. IEICE Trans. **101**–**A**(12), 2162–2170 (2018)
38. Wang, Y., Takagi, T.: Improving the BKZ reduction algorithm by quick reordering technique. In: Susilo, W., Yang, G. (eds.) ACISP 2018. LNCS, vol. 10946, pp. 787–795. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93638-3_47
39. Wang, Y., Wunderer, T.: Revisiting the sparsification technique in Kannan's embedding attack on LWE. In: Su, C., Kikuchi, H. (eds.) ISPEC 2018. LNCS, vol. 11125, pp. 440–452. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99807-7_27