EPiC Series in Computing

Volume 106, 2025, Pages 40–54

Proceedings of the 20th Asia Joint Conference on Information Security

An Attack on Ring-LWE Using a Dynamically Adjustable Block Size BKW Algorithm

Kenjiro Hirose, Shinya Okumura, and Atsuko Miyaji

[1] The University of Osaka, Suita, Osaka, Japan
hirose@cy2sec.comm.eng.osaka-u.ac.jp
[2] okumura@comm.eng.osaka-u.ac.jp
[3] miyaji@comm.eng.osaka-u.ac.jp

Abstract

The Ring-LWE problem is a fundamental component of lattice-based cryptography, and evaluating its security is a crucial challenge. The algorithms for solving the Ring-LWE problem can be classified into four categories: lattice basis reduction algorithms, algebraic methods, combinatorial methods, and exhaustive search algorithms. However, the combinatorial approach, the Ring-BKW algorithm, remains insufficiently analyzed. The Ring-BKW algorithm primarily consists of two steps, with the Reduction step being the bottleneck because many samples are required for decryption. In existing implementations of the Ring-BKW Reduction step, the block size remains fixed, preventing it from adapting to the sample reduction process and efficiently inducing collisions. In this study, we introduce a method that allows the block size in the Reduction step of the Ring-BKW algorithm to be variable. We propose two approaches: a static decision method, where users manually specify the block size for each reduction step, and a dynamic decision method, where the algorithm autonomously adjusts the block size. The proposed method increases the number of collisions compared to existing methods, resulting in approximately 55-fold and 425-fold more reduced samples for static and dynamic block size selection, respectively, in the Ring-LWE setting with a modulus $q = 17$ and a ring dimension $n = 2^4$.

1 Introduction

With the advancement of quantum computing, Shor [16] demonstrated that the mathematical hardness assumptions on which the security of currently used cryptographic schemes such as RSA and elliptic curve cryptography is based, namely, the integer factorization problem and the discrete logarithm problem, can be solved in polynomial time. In response, the National Institute of Standards and Technology (NIST) initiated a public call for post-quantum cryptographic algorithms that remain secure even against quantum computers. In 2024, NIST announced the final selection of four encryption schemes as Federal Information Processing Standards (FIPS). Among the four encryption schemes, CRYSTALS-KYBER [7], CRYSTALS-Dilithium [10], and FALCON [14], which are currently undergoing further standardization, are lattice-based cryptographic schemes and are expected to attract increasing attention in the future. The Learning with Errors (LWE) problem is one of the mathematical hardness assumptions that underpin the security of lattice-based cryptography. The currently well-known methods for solving the LWE problem can be broadly classified into four categories. These methods include lattice

basis reduction algorithms [8, 15], algebraic methods [4], combinatorial methods [6, 9], and exhaustive search algorithms [3, 19]. Among these, the Blum-Kalai-Wasserman (BKW) algorithm, a combinatorial method, has not been thoroughly studied in the past. Originally, the BKW algorithm was devised as a solution for the Learning Parity with Noise (LPN) problem [5]. Later, it was shown to be applicable to the LWE problem, leading to a detailed analysis of its computational complexity [1]. The BKW algorithm consists of three steps: Reduction, Hypothesis-testing, and Back-substitution. In attacks on the LWE problem using the BKW algorithm, the primary bottlenecks are the required number of samples and the computational cost in the Reduction step. The previous researches have explored various approaches to mitigate these bottlenecks [2, 11, 12]. Stange proposed the Ring-BKW algorithm, which is an improvement of the BKW algorithm for solving the Ring-LWE problem [17]. The Ring-BKW uses the algebraic properties of rings and employs rotations to increase the number of input samples, thereby reducing the required number of the Ring-LWE sample. Additionally, it optimizes the data storage method for collision tables used in the Reduction step, enabling a reduction in computational cost. Furthermore, in the case of Ring-LWE over power-of-two cyclotomic fields, it has been shown that the Back-substitution step of the BKW algorithm can be omitted by utilizing the trace map.

However, existing methods use a fixed block size throughout the Reduction step, which results in inefficient collisions at each reduction stage. To address this issue, we propose two methods: a static block size selection method, in which the user predefines the block size for each reduction stage, and a dynamic block size selection method, in which the block size is adjusted adaptively during the algorithm based on the reduction progress. Compared to existing methods, our proposed approach increases the number of collisions, resulting in approximately a 55-fold increase in the number of reduced samples with the static method and approximately a 425-fold increase with the dynamic method, in the case of Ring-LWE with a modulus $q = 17$ and ring dimension $n = 2^4$. We also conducted experiments to evaluate the effect of the parameters α and β, which are used in the dynamic method.

2 Preliminary

In this section, we shortly describe the LWE problem and the Ring-LWE problem.

Definition 2.1. *Let n be a positive integer and q be an odd prime. Let χ be a discrete Gaussian distribution over $\mathbb{Z}$ with mean 0 and standard deviation σ. For a secret vector $\mathbf{s} \in \mathbb{F}_q^n$ and an error $e \in \mathbb{Z}$ sampled from the discrete Gaussian distribution χ, define the probability distribution $\mathcal{A}_{\mathbf{s},\chi}$ as the distribution of $(\mathbf{a}, b) \in \mathbb{F}_q^n \times \mathbb{F}_q$, $b \equiv \langle \mathbf{a}, \mathbf{s} \rangle + e \pmod q$. A sample drawn from the probability distribution $\mathcal{A}_{\mathbf{s},\chi}$ is called an LWE sample. Given an arbitrary number of LWE samples $(\mathbf{a}, b) \in \mathbb{F}_q^n \times \mathbb{F}_q$ from $\mathcal{A}_{\mathbf{s},\chi}$, the problem of recovering the secret vector $\mathbf{s} \in \mathbb{F}_q^n$ from these samples is called the (search) LWE problem.*

We consider solving the LWE problem using m different pairs $(\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \pmod q)$ $(1 \le i \le m)$ sampled from the probability distribution $\mathcal{A}_{\mathbf{s},\chi}$. Let $\mathbf{A}$ be an $m \times n$ matrix whose i-th row is $\mathbf{a}_i$, $\mathbf{b} = (b_1, \ldots, b_m)$ and $\mathbf{e} = (e_1, \ldots, e_m)$ the error vector. Then, the relation $\mathbf{b} \equiv \mathbf{s}\mathbf{A}^T + \mathbf{e} \pmod q$ holds, and the LWE samples are treated as a pair $(\mathbf{A}, \mathbf{b}) \in \mathbb{F}_q^{m \times n} \times \mathbb{F}_q^m$.

Definition 2.2. *Let p be a prime number and define the n-th cyclotomic polynomial as $\Phi_n(x) = \prod_{1 \le k \le n,\ \gcd(k,n)=1} (x - \zeta_n^k)$. Here, ζ_n denotes a primitive n-th root of unity. Let K be the number field $\mathbb{Q}(\zeta_n)$ and R its ring of integers given by the isomorphism $R \cong \mathbb{Z}[x]/(\Phi_n(x))$. Define R_p as $R_p = R/pR$, and let D_{σ_s} and D_{σ_e} be probability distributions over R_p. For a uniformly random $a \in R_p$, with $s \leftarrow D_{\sigma_s}$ and $e \leftarrow D_{\sigma_e}$, the pair $(a, b = a \cdot s + e) \in (R_p \times R_p)$ is called a Ring-LWE sample. Given an arbitrary number of Ring-LWE samples (a, b), the problem of recovering the secret $\mathbf{s}$ from these samples is called the (search) Ring-LWE problem.*

3 Existing research

In this section, we describe the Plain BKW algorithm [1], which is an algorithm for solving the LWE problem, and Ring-BKW, which is its adaptation to the Ring-LWE problem [17].

3.1 Plain BKW

The BKW algorithm was proposed as a method for solving the LPN problem using $2^{O(n/\log n)}$ samples and computation time. The BKW algorithm can also be applied to the LWE problem, and in the case of an n-dimensional instance, it can find a solution with a computational complexity of $2^{O(n)}$ [1]. The BKW algorithm consists of three steps: Reduction, Hypothesis-testing, and Back-substitution. The Reduction step aims to reduce the dimensionality of LWE samples by performing sample collisions block by block using a method similar to Gaussian elimination. The Hypothesis-testing step estimates partial vectors of the secret vector using the dimension-reduced samples obtained in the Reduction step. Finally, in the Back-substitution step, the estimated partial vectors are used to reconstruct the entire secret vector. The following provides a detailed explanation of each step.

3.1.1 Reduction step

Select the dimension $n \in \mathbb{Z}$ and the block size $0 < b \leq n$, and set the reduced dimension as $d < b$. Additionally, define $a := \lceil n/b \rceil$. Let $L^{(n)}_{\mathbf{s},\chi}$ be the LWE oracle and define $B^{(n)}_{\mathbf{s},\chi,\ell}$ as the oracle that outputs samples $\mathbf{a}_i$ where the first $b \cdot \ell$ columns are all zero. At this point, $B^{(n)}_{\mathbf{s},\chi,\ell}$ is generated as follows.

- if $\ell = 0$, $B^{(n)}_{\mathbf{s},\chi,0}$ is simply $L^{(n)}_{\mathbf{s},\chi}$.
- if $0 < \ell < a$, $B^{(n)}_{\mathbf{s},\chi,\ell}$ is obtained from the difference of two vectors derived from $B^{(n)}_{\mathbf{s},\chi,\ell-1}$.

The Reduction step is a recursive algorithm and terminates when the dimension of nonzero vectors reaches d. Thus, the solution space can be reduced from $\mathbb{Z}_q^n$ to $\mathbb{Z}_q^d$. The pseudocode for constructing the LWE oracle $B^{(n)}_{\mathbf{s},\chi,\ell}$ for $0 < \ell < a$ is presented in Algorithm 1.

When constructing the algorithm as described above, the cost of querying $B^{(n)}_{\mathbf{s},\chi,\ell-1}$ to obtain a single sample from $B^{(n)}_{\mathbf{s},\chi,\ell}$ is $\lceil q^b/2 \rceil$. Additionally, it is necessary to perform at most one addition of the two outputs of $B^{(n)}_{\mathbf{s},\chi,\ell-1}$ whose first $b \cdot \ell$ entries are common. This operation requires $n + 1 - b \cdot \ell$ additions in $\mathbb{Z}_q$.

3.1.2 Hypothesis-testing Step

Reduction step generates $B^{(n)}_{\mathbf{s},\chi,a}$ which is an oracle that returns $(\mathbf{a}_i, c_i)$, where $\mathbf{a}_i \in \mathbb{Z}_q^d$, and $c_i \in \mathbb{Z}_q$. In the Hypothesis-testing step, equations $f_i = -c_i \pm j + \sum_{k=0}^{d-1} (\mathbf{a}_i)_{(k)} x_{(k)}$ for $0 \leq j < q/2$ is formulated for each sample obtained from this oracle. Here, $(\mathbf{a}_i)_{(k)}$ denotes the k-th element of the vector $\mathbf{a}_i$, and $x_{(k)}$ represents the corresponding unknown variable. This equation is applied to each candidate vector obtained from $B^{(n)}_{\mathbf{s},\chi,a}$ to estimate the partial vector of the secret vector. Additionally, the score of each candidate vector $\mathbf{v}$ is evaluated using the weight function $W\big(-c_i + \sum_{k=0}^{d-1} (\mathbf{a}_i)_{(k)} v_{(k)}\big)$. The pseudocode for Hypothesis-testing using these is presented in Algorithm 2.

Hypothesis-testing weight functions have been extensively studied; here, we introduce an example of a weight function based on the log-likelihood ratio. Let χ_a be an error distribution under the correct guess ($\mathbf{v} = \mathbf{s}$), and let $\mathcal{U}_a$ be an error distribution under an incorrect guess ($\mathbf{v} \neq \mathbf{s}$). By the

Algorithm 1 Reduction Step $B^{(n)}_{\mathbf{s},\chi,\ell}$ for $0 < \ell < a$

Input: b: integer $0 < b \leq n$, ℓ: integer $0 < \ell < a$
Output: $(\mathbf{a}, c)$: Reduced samples

$T^{\ell} \leftarrow$ array indexed by $\mathbb{Z}_q^b$ maintained across all runs of $B^{(n)}_{\mathbf{s},\chi,\ell}$.
query $B^{(n)}_{\mathbf{s},\chi,\ell-1}$ to obtain $(\mathbf{a}, c)$.
if $\mathbf{a}_{(b\cdot(\ell-1),b\cdot\ell)}$ is $\mathbf{0}$ **then**
  return output $(\mathbf{a}, c)$
end if
while $T_{\mathbf{a}_{(b\cdot(\ell-1),b\cdot\ell)}} = \emptyset$ **do**
  $T_{\mathbf{a}_{(b\cdot(\ell-1),b\ell)}} \leftarrow (\mathbf{a}, c)$
  $T_{-\mathbf{a}_{(b\cdot(\ell-1),b\ell)}} \leftarrow (-\mathbf{a}, -c)$
  query $B^{(n)}_{\mathbf{s},\chi,\ell-1}$ to obtain $(\mathbf{a}, c)$.
  if $\mathbf{a}_{(b\cdot(\ell-1),b\ell)}$ is $\mathbf{0}$ **then**
    return output $(\mathbf{a}, c)$
  end if
end while
$(\mathbf{a}', c') \leftarrow T_{\mathbf{a}_{(b\cdot(\ell-1),b\ell)}}$
return output $(\mathbf{a} + \mathbf{a}', c + c')$

Algorithm 2 Hypothesis testing Step

Input: F - a set of m samples following $B^{(n)}_{\mathbf{s},\chi,a}$, W - a weight function mapping members of $\mathbb{Z}_q^d$ to real numbers.
Output: S - a array filled with scores for each vectors,

$S \leftarrow$array filled with zeros indexed by $\mathbb{Z}_q^d$.
for $\mathbf{v} \in \mathbb{Z}_q^d$ **do**
  $w_{\mathbf{v}} \leftarrow \emptyset$.
  for $f_i \in F$ **do**
    write f_i **as** $-c_i + \sum_{k=0}^{d-1} (\mathbf{a}_i)_{(k)} \cdot x_{(k)}$.
    $j \leftarrow \langle \mathbf{a}_i, \mathbf{v} \rangle - c_i$.
    $w_{\mathbf{v}} \leftarrow w_{\mathbf{v}} \cup W(j)$.
  end for
  $S_{\mathbf{v}} \leftarrow \sum_{w_i \in w_{\mathbf{v}}} w_i/m$.
end for

Neyman-Pearson lemma, the log-likelihood ratio is known to be the most powerful test for determining whether a sample follows one of two known distributions [13]. Thus, the weight function is defined as $W(j) := \log_2 \left(\frac{\Pr[e \leftarrow_{\$} \chi_a : e = j]}{\Pr[e \leftarrow_{\$} \mathcal{U}_a : e = j]} \right)$ for the range $\lceil -q/2 \rceil \leq j \leq \lfloor q/2 \rfloor$.

3.1.3 Back-substitution step

When a candidate vector $\mathbf{s}'$ with a very high probability of being correct is obtained through Hypothesis-testing, Back-substitution is performed using the table T_i. Back-substitution restores the original dimension by performing computations using the table employed in the Reduction step to reduce the sample dimension.

The bottlenecks in the BKW algorithm are the computational complexity and the number of queries in the Reduction step, as well as the estimation accuracy in the Hypothesis-testing step [19]. Methods for improving the performance of the Reduction step have been studied, including Lazy Modulus Switching [2], Coded-BKW [12], and Coded-BKW with sieving [11], which applies the sieve algorithm.

3.2 Ring-BKW

The Ring-BKW algorithm is an application of the BKW algorithm, introduced so far, to the Ring-LWE problem. In this section, we introduce the basic Ring BKW algorithm, Blind BKW, as well as Traditional BKW and Advanced BKW, which improve the performance of the Reduction step by leveraging the characteristics of Ring-LWE samples. First, we define the trace used for decryption in the Ring-BKW algorithm.

Definition 3.1. *If considering Ring-LWE in R_q, where R is the ring of integers of a number field K, then any subfield $L \subseteq K$ gives rise to a subring $S \subseteq R$ and, modulo q, to a subring $S_q \subseteq R_q$. Then S_q is $\mathbb{F}_q$-vector subspace of R_q, and R_q has a module structure over S_q. The extension degree of K over L is the same as the rank of R as S-module and the rank of R_q as S_q-module. There is a linear map $T := \mathrm{Tr}_{S_q}^{R_q} : R_q \to S_q$ satisfying $\mathrm{Tr}_S^R(x) \mod qS = \mathrm{Tr}_{S_q}^{R_q}(x \mod qR)$, where $\mathrm{Tr}_S^R(\cdot)$ is the usual trace map from R to S.*

3.2.1 Blind BKW

The Ring-BKW consists of the Reduction step and the Hypothesis-testing step, as the Back-substitution step required in the BKW algorithm becomes unnecessary by utilizing Theorem 3.1 [17]. Eliminating the Back-substitution step reduces the polynomial-time computational complexity.

Theorem 3.1. *Let R be the ring of integers of the m-th cyclotomic field, where $m = 2n$ is a power of 2. Let S be the ring of integers of the k-th cyclotomic field. If $k \mid m$, then $S \subseteq R$, and the extension degree of $\mathbb{Q}(\zeta_m)/\mathbb{Q}(\zeta_k)$ is m/k. Assume that the rational prime q is unramified in R.*

Consider a Ring-LWE *instance in $R_q \times R_q$ with secret vector $\mathbf{s} \in R_q$ and error distribution χ that is invariant under multiplication by ζ_m. Assume that $a_0 \in R_q$ is an invertible element. Define $T := \mathrm{Tr}_{S_q}^{R_q}$, and assume $T(a_0)$ is also invertible. Suppose that N samples (a, b) follow the distribution $A_{a_0 S_q, \mathbf{s}, \chi}$.*

Then, the computation of the secret $\mathbf{s} \in R_q$ can be reduced to solving m/k independent search Ring-LWE *problems over S_q, each with N samples and error distribution $\frac{k}{m}T(\chi)$ over S_q, in linear time in n and polynomial time in* $\log q$.

Moreover, these m/k Ring-LWE problems are mutually independent in the sense that solving one does not require any information from the others. Furthermore, if the error distribution χ is constructed from a coefficient distribution χ_0 over $\mathbb{F}_q$ with respect to the ζ_m-basis, then $\frac{k}{m}T(\chi)$ is constructed similarly.

To apply Theorem 3.1, it is necessary to reduce the sample set to the subring S_q in the Reduction step. To achieve this, the Ring-LWE samples are reordered in descending order of the order of the $\zeta-$basis in R_q before performing the Reduction step. That is, given the basis $1, \zeta, \zeta^2, ..., \zeta^{n-1}$, the elements are reordered as $\zeta_m^{n-1}, \zeta_m^{\frac{n}{2}-1}, \zeta_m^{\frac{3n}{4}-1}, \zeta_m^{\frac{n}{4}-1}, ..., \zeta_m^{\frac{3n}{4}}, \zeta_m^{\frac{n}{4}}, \zeta_m^{\frac{n}{2}}, 1$. By reordering the basis as described above and performing the Reduction step, the input samples can be reduced to the subring, enabling the application of Theorem 3.1. After reducing the samples to the subring in this manner, any Hypothesis-testing method can be used to estimate the partial secret vector $\mathbf{s}'$, and the obtained vector is recovered using Theorem 3.1. Algorithm 3 presents the pseudocode for the Ring BKW. Here, S_q and n are the same as above, and B indicates the block size.

Algorithm 3 Ring-BKW Algorithm

Input: $(\mathbf{a}, c)$ - a set of samples following $B^{(n)}_{\mathbf{s},\chi,a}$
Output: $\mathbf{s}$ - the secret

1: Run BKW Reduction with reordered samples on the values a until all samples (a, b) have $a \in S_q$.
2: Use Theorem3.1 to create samples from n/B different Ring-LWE problems in S_q.
3: Solve these Ring-LWE problems using any method of choice.
4: Use Theorem 3.1 to recover the secret s in polynomial time from these solutions.

Algorithm 4 Traditional BKW Reduction Step

Input: B - Block size, n - dimension
Output: $(\mathbf{a}, c)$ - Reduced samples

1: Create empty Table 1 through n/B.
2: **for** each initially available sample (a, b) **do**
3: **for** $j = 0$ to $n - 1$ **do**
4: Rotate the sample by ζ^j, to obtain (a_1, b_1).
5: Send the sample (a_1, b_1) to Table 1.
6: **end for**
7: **end for**
8: **for** each sample (a, b) sent to Table i, $i < n$ **do**
9: **if** a has all 0 entries in block i **then**
10: send sample (a, b) on to Table $i + 1$.
11: **end if**
12: **if** the first non-zero coefficient of a_1 is in the range 1to$(q + 1)/2$ **then**
13: Multiply (a, b) by -1.
14: **end if**
15: **if** a collision is found (i.e. a sample (a_0, b_0) already exists in the table having the same first i blocks of size B) **then**
16: Subtract (a_1, b_1) from (a_0, b_0) to obtain a new sample whose first i block of size B are zero
17: Send the result to Table $i + 1$.
18: **else**
19: Store the associated sample in Table i.
20: **end if**
21: **end for**

3.2.2 Traditional BKW

Next, we describe a method for reducing the required number of samples in the Blind BKW over power-of-two cyclotomic fields by leveraging the applicability of ring properties to Ring-LWE samples. Given any Ring-LWE sample (a, b) in a power-of-two cyclotomic field, considering its negated counterpart $(-a, -b)$ allows us to obtain two samples from a single one. Additionally, it is also possible to rotate the samples within each block. When the block size satisfies $B \mid n$, using $\zeta^{n/B}$ ensures that the first block of $\mathbf{a}$ remains a zero vector. This allows each block to rotate the samples using $1, \zeta^{n/B}, \zeta^{2n/B}, ..., \zeta^{(B-1)n/B}$, thereby reducing the number of required samples to $1/B$ of the original. As described above, by utilizing negation and rotation, the same Reduction step as in the Plain BKW can be performed with only $1/2B$ of the samples required in the Plain BKW. The Ring BKW algorithm that employs this method is called Traditional BKW: Algorithm 4.

4 Dynamically Adjustable block size method

In this section, we propose two methods for improving the Reduction step of the Ring-BKW algorithm.

In the Reduction step of [17], collisions were performed using the same number of blocks throughout all steps. However, since the number of samples stored in each table T^i varies at each step, it is difficult to say that an optimal block size is consistently used. In the initial step, since the number of samples is abundant, collisions can occur and the dimension can be reduced even with a fixed block size. However, as the steps progress, the number of samples decreases, reducing the probability of collisions. Therefore, by varying the block size according to properties such as the number of samples stored in each table T^i at each step, the probability of collisions can be increased, which is expected to result in a greater number of samples output by the Reduction step. Therefore, in this section, we describe two methods for varying the block size in the Reduction step of the Ring-BKW algorithm: the static block size determination method and the dynamic block size determination method.

4.1 Static block size determination method

First, we extend the block size, which was previously given as an integer in existing methods, to an array. This extension allows the block size to be set individually for each step. In the Ring-BKW, to reduce the given ring to a subring, the block size must be a power of two in a power-of-two cyclotomic field. Considering this condition, the Ring-BKW algorithm is executed with the block size decreasing as the number of steps increases. We call this method *static block size determination method*.

4.2 Dynamic block size determination method

Static block size determination method requires the user to manually set the block size according to the number of steps. Therefore, when the sample set is first obtained, the user needs to perform the Reduction step multiple times to determine the optimal block size, making the process inefficient. To address this issue, we introduce Change block size, which determines the block size based on the number of samples, and propose a method for dynamically adjusting the block size. We call this method *dynamic block size determination method*.

Change block size is a process performed within the Reduction step and is executed when two conditions are met. The first condition is met when the number of stored columns in the current step falls below a fixed threshold β. The second condition is met when a fixed number of samples has been processed, as determined by the processed sample threshold α. The simplest way to increase collisions in the Reduction step is to increase the number of samples. However, obtaining a large number of samples through an excessive number of queries at the start of the Reduction step is not practical. Therefore, it is important to process each step while minimizing the reduction of the initially given number of samples and ultimately map them to the desired subring. Therefore, we introduce a threshold β $(0 < \beta < 1)$ to ensure the number of columns in each table T^i at every step. At each step, the maximum number of columns is given by $(q^B - 1)/2$, where B is the block size at the corresponding step and q is the modulus [17]. Therefore, we predefine the proportion of samples to be maintained at each step using β and adjust the block size when the number of columns falls below $(q^B - 1) \cdot \beta/2$. However, since the Reduction step is a recursive algorithm, specifying only the column threshold β would cause the block size to be divided into smaller sizes from the beginning of the Reduction step. A smaller block size increases the probability of collisions. However, it also increases the number of required steps, leading to higher computational complexity. Therefore, we introduce the second condition, which triggers when a certain number of samples has been processed. When introducing this condition, we use the processed sample threshold α $(0 < \alpha \leq 1)$ to execute Change block size when the ratio of processed samples to input samples exceeds α. Algorithm 5 presents the procedure for Change block size.

Algorithm 5 Change block size($i,\ t,\ \alpha, \beta$)

Input: i - index of processing steps, t - the number of processed samples, α - a threshold of the number of processed samples, β - a threshold of the number of the stored columns
Output: B - an array of block size

$rownum_{max} = q^{B[i-1]}/2$
$tableRowTH = rownum_{max} \cdot \beta$
if the percentage of processed samples> α **then**
  if the rows number of T^t < tableRowTH **then**
    $tmpB \leftarrow B[t].pop()$
    Insert $[tmpB/2, tmpB/2]$ to $B[t]$
    Initialize T^t
    Create T^{t+1}
    $priority\_index = i - 1$
  end if
end if

Algorithm 6 Dynamic Variable block size Blind-BKW

Input: $(\mathbf{a}, c)$ - Ring-LWE samples, B - an array of block size
Output: $(\mathbf{a}', c')$ - Reduced Ring-LWE samples

Create len(B) Tables.
for All Ring-LWE samples **do**
  if $priority\_index \neq 0$ **then**
    for All Ring-LWE samples in $table[priority\_index]$ **do**
      Reduction sample with new block size
    end for
    $priority\_index = 0$
  end if
  Reduction Ring-LWE sample
  $Change\ blocksize(i, t, \alpha, \beta)$
end for

Change block size is used as a subroutine in the Reduction step of the Ring-BKW algorithm. Algorithm 6 presents the application of Change block size as a subroutine in the Blind BKW, while Algorithm 7 shows its application in the Traditional BKW.

The operation of Algorithm 6 and 7 is described using Figures 1 and 2. The elements of the initial block size array must all be powers of two to utilize Theorem 3.1. First, as a preparation step, we create tables for Reduction corresponding to the length of the initial block size array. This is because the number of elements in the block size array matches the number of steps in the Reduction step. Once a sample is input, the two previously mentioned conditions are evaluated: (1) whether the ratio of processed samples remains below the initially set threshold, and (2) whether the number of columns in the current step's table falls below the threshold. If neither condition is met, the Reduction process proceeds using the initially assigned block size, as shown in Figure 2 (①). On the other hand, if both conditions are met, Change block size is executed. When Change block size is executed, the block size is updated to a new value. In the proposed method, to ensure that the block size remains a power of two, the current step's block size is removed from the array, and two new values, each obtained by halving the removed block size, are inserted into the array. Since splitting the block size in half increases the number of steps by one, an empty storage table is inserted into the table used for collisions, and the table

Algorithm 7 Dynamic Variable block size Trad-BKW

Input: $(\mathbf{a}, c)$ - Ring-LWE samples
Output: $(\mathbf{a}', c')$ - Reduced Ring-LWE samples

Create len(B) Tables.
for All Ring-LWE samples **do**
  if $priority\_index \neq 0$ **then**
    for All Ring-LWE samples in $table[priority\_index]$ **do**
      Reduction sample with new block size
    end for
    $priority\_index = 0$
  end if
  for $j = 0$ to $n - 1$ **do**
    Rotate the sample by ζ^j to obtain $(\mathbf{a}_1, c_1)$
    Store the associated sample $(\mathbf{a}_1, c_1)$ in Table.
  end for
  Reduction Ring-LWE sample
  $Change\ block\ size(i, t, \alpha, \beta)$
end for

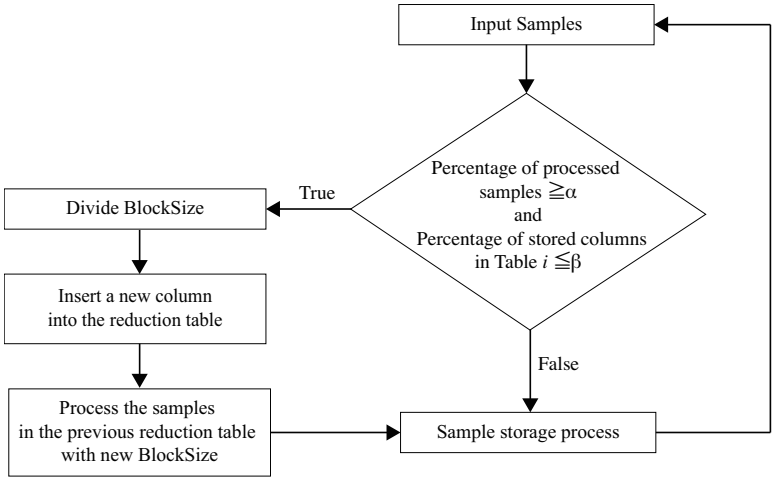

Figure 1: Flow of the Dynamic Decision Method

that previously stored collisions for the corresponding step is reset, as shown in Figure 2 (②). Here, the reason for resetting the table is to store the sample set that has been collided using the newly assigned block size, while also reusing the table to reduce space complexity. After that, the samples from the previous table of the corresponding step undergo the Reduction process using the newly assigned block size in Change block size, as shown in Figure 2 (③). In Algorithm 6 and 7, the priority_index stores the index of the previous step. The priority_index is initialized to 0 at the start of the algorithm. Within Blind BKW and Traditional BKW, if $priority\_index \neq 0$, the sample processing for the corresponding index is prioritized before performing the usual sample processing.

With the proposed method, the number of blocks in the Reduction step can be dynamically adjusted, increasing the probability of sample collisions. Additionally, in the static decision method, prior

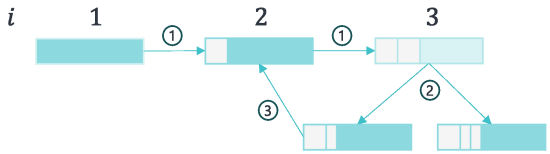

Figure 2: The behavior of T for a total of 3 steps

information such as the step at which the number of samples decreases was required to specify an appropriate block size array in the Reduction step. However, by using the dynamic decision method, it is possible to ensure the sample count and generate reduced-dimensional samples without requiring prior information.

5 Experiments and consideration

5.1 Experiment environment

In this experiment, to compare with existing methods, we perform the Reduction step using Blind BKW with a dynamically determined block size on Ring-LWE samples over power-of-two cyclotomic fields. We conduct experiments on both the static and dynamic methods for varying the block size. For the static method, the Reduction step is first executed with a fixed block size, and based on the obtained results, the block size array is determined and then executed. For the dynamic method, we use a processed sample threshold $\alpha = 0.8$ and a stored column threshold $\beta = 0.06$. Additionally, for the dynamic decision method, we conducted experiments to determine the optimal values for α and β by varying one parameter while keeping the other fixed. These experiments measure the execution time, the number of reduced samples, and the table size. We use a computer whose CPU is the Apple M2. Its macOS is the sonoma version 14.7.2. We use SageMath version 10.3 [18] as software. We developed a program by improving the Ring-BKW algorithm [17], which was then used for experimentation. The source code is available at https://github.com/Kenjiro56/VariableBlockSizeBKW.

5.2 The number of Reduced Samples

Tables 1–2 present the experimental results for the Ring-LWE samples. The initial sample count represents the number of samples input into the Reduction step. The final block size refers to the ultimate block size used in the Reduction step. The table size indicates the total number of columns, excluding the final column, in the table created for collisions in the Reduction step. The number of reduced samples represents the final count of samples that successfully underwent dimensionality reduction to the desired dimension. Additionally, the execution time represents the time required for the Reduction step to complete, measured using the timeit function in SageMath. For the final block size, if the same block size appears consecutively multiple times, it is abbreviated as [block size * number of occurrences]. Applying the BKW algorithm with dynamically variable block sizes to Ring-LWE samples increases the number of reduced samples compared to Blind BKW or static decision methods As the number of reduced samples increases, the table size, excluding the final table, decreases. Regarding execution time, compared to existing methods, the static decision method is approximately 150 times slower for both cases when $q = 17, n = 2^4$, while in the dynamic decision method, Blind BKW is about 1000 times slower and Traditional BKW is about 3000 times slower. For $q = 211, n = 2^3$, the execution time of the static decision method is approximately 1.8 times faster than that of existing method This is likely because increasing the number of reduction steps due to block size subdivision enhances the

Table 1: Reduction results for Ring-LWE samples ($q = 17, n = 2^4$)

	Conventional	Our Proposal	
	Ring-BKW [17]	Static Method	Dynamic Method
Initial Samples	$2000 \cdot 2^4$		
Block Size	2^2	$[2^2, 2^2, 2, 2, 2^2]$	$[2^0 * 12, 2^2]$
Table Size	31985	31123	21610
Runtime	6.13 ms	911 ms	6.04 s
Reduced Samples	15	838	6388

Table 2: Reduction results for Ring-LWE samples($q = 211, n = 2^3$)

	Conventional	Our Proposal	
	Ring-BKW [17]	Static Method	Dynamic Method
Initial Samples	$4000 \cdot 2^3$		
Block Size	2^2	$[2, 2, 2^2]$	$[2^0 * 12, 2^2]$
Table Size	31999	27839	424
Runtime	1.43 s	800 ms	4.1 s
Reduced Samples	1	4161	25173

collision probability but also increases processing time. However, despite the increase in the number of reduced samples, a comparison of the final block size reveals that it has been subdivided to the finest possible level, significantly increasing execution time. This excessive execution time is due to an overuse of Change block size caused by inappropriate settings for the processed sample threshold and stored column threshold. Since the block size setting involves a trade-off between the number of reduced samples and execution time, it is necessary to determine appropriate parameters based on the required number of reduced samples.

5.3 The relations between the number of reduced samples and parameters

Figures 3–4 show the execution time and the number of reduced samples when α is fixed in the block size dynamic decision method. Similarly, Figures 5–6 present the execution time and the number of reduced samples when β is fixed. The horizontal axis represents the stored column threshold β and the processed sample threshold α, while the vertical axis indicates the execution time and the number of reduced samples, respectively.

When β is varied, both execution time and the number of reduced samples increase significantly at 0.05 and 0.45, remaining unchanged beyond these values. On the other hand, when α is varied, increasing it results in a decrease in execution time and the number of reduced samples, with no significant changes observed beyond 0.5. We analyze the settings for the processed sample threshold α and the stored column threshold β. When β is small, the number of reduced samples is low, and the execution time is short. In this parameter comparison experiment, we used 32,000 samples, $q = 17$, $\alpha = 0.8$, and an initial block size of 4. As a result, $32,000 \times 0.8 = 25,600$ samples were processed, and Change block size was executed for the first time when $(17^4 - 1) \times \beta/2 = 41,760 \times \beta$. When 80% of the input samples have been processed, the parameters in this experiment indicate that a certain number of collisions occur, allowing the reduction steps to progress. In this case, when β is small, the required number of stored columns is low, preventing Change block size from being executed in the early reduction steps. Instead, block size is modified only in the later reduction steps, where the number of stored columns is low. As a result, the

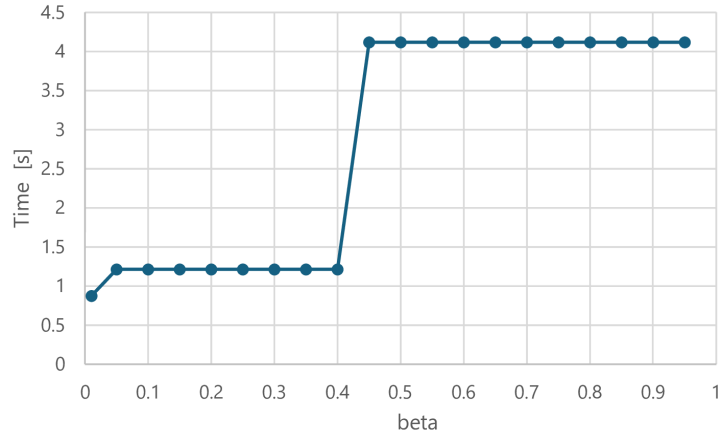

Figure 3: Execution time of the dynamic decision method ($\alpha = 0.8$)

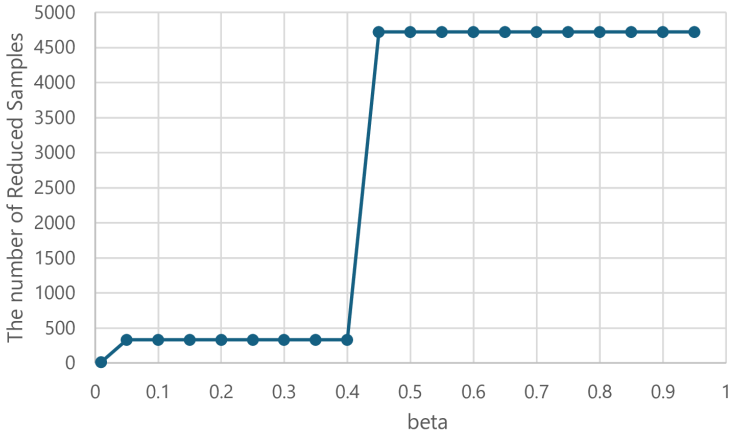

Figure 4: Number of reduced samples in the dynamic decision method ($\alpha = 0.8$)

execution time is shorter since no block size changes occur in the early steps, and the number of reduced samples is lower compared to when β is large. Conversely, as β increases, the required number of stored columns also increases in the early reduction steps, leading to earlier block size changes. Consequently, execution time increases, and the number of reduced samples also increases. Next, we analyze the impact of α on block size changes. In this experiment, β was fixed at 0.06. As a result, when $32,000 \times \alpha$ samples have been processed, if the number of columns in the ongoing reduction step at that moment does not satisfy $(17^4 - 1) \times 0.06/2 \simeq 2506$, the first Change block size is executed. When α is small, Change block size is executed early in the sample input process, reducing block size to the smallest possible size. This results in longer execution times and a higher number of reduced samples. Conversely, as α increases, the timing of Change block size execution is delayed, leading to block size refinement occurring only in the later stages. As a result, the number of reduced samples is expected to be lower compared to earlier stages. In this study, we have not yet clarified the relationship between the number of reduced samples

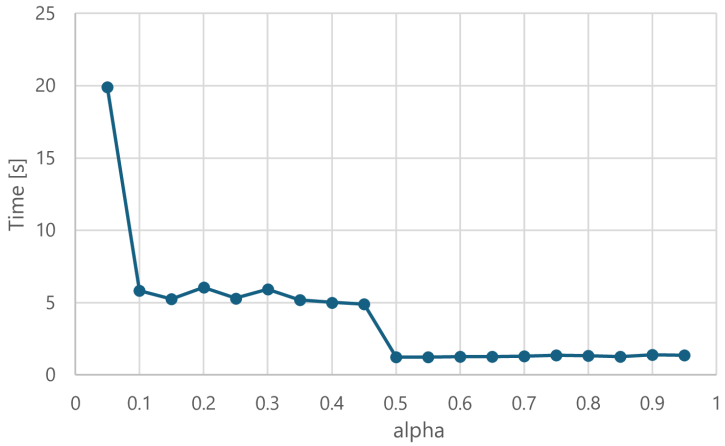

Figure 5: Execution time of the dynamic decision method ($\beta = 0.06$)

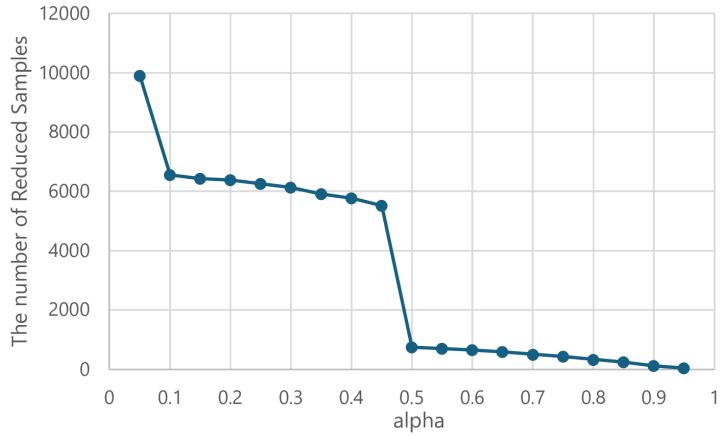

Figure 6: Number of reduced samples in the dynamic decision method ($\beta = 0.06$)

required in Hypothesis-testing and the success probability of the attack. However, by elucidating this relationship in future work, β can also be appropriately configured. Furthermore, as mentioned earlier, the setting of the processed sample threshold α serves as an auxiliary condition to properly manage the recursive Reduction step, which determines the execution conditions for Change block size. If the stored column threshold β is appropriately set, it is also expected that an appropriate value for α can be determined.

6 Conclusion

In this study, we improved the Reduction step by dynamically setting an appropriate number of BlockSize values for each reduction step instead of using a fixed BlockSize, thereby increasing the number of reduced

samples. This result demonstrates an improvement in the performance of the Ring-BKW algorithm, as it allows more candidate vectors to be output using the same number of Ring-LWE samples, while also suggesting the possibility of further enhancements to the BKW algorithm. In this study, we introduced a threshold for the number of available samples at each step as a method for dynamically determining the number of BlockSize values. However, other approaches to determining BlockSize are conceivable, and devising methods that utilize different statistical properties of the sample set remains an open problem. Additionally, further analysis is required to appropriately configure the parameters used in the proposed Change BlockSize method. To achieve this, it is also necessary to clarify the number of samples required at each step for the BKW algorithm to successfully attack the LWE and Ring-LWE problems, enabling more detailed research on the Reduction step.

Acknowledgment

This work is partially supported by JSPS KAKENHI Grant Number JP21H03443 and SECOM Science and Technology Foundation.

References

[1] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. On the complexity of the BKW algorithm on LWE. Cryptology ePrint Archive, Paper 2012/636, 2012.

[2] Martin R. Albrecht, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Lazy modulus switching for the BKW algorithm on LWE. Cryptology ePrint Archive, Paper 2014/019, 2014.

[3] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9:169 – 203, 2015.

[4] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jiří Sgall, editors, *Automata, Languages and Programming*, pages 403–415, Berlin, Heidelberg, 2011.

[5] Avrim Blum, Merrick Furst, Michael Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '93, page 278–291, Berlin, Heidelberg, 1994.

[6] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, July 2003.

[7] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018*, pages 353–367. IEEE, 2018.

[8] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011.

[9] Xiaoyang Dong, Jialiang Hua, Siwei Sun, Zheng Li, Xiaoyun Wang, and Lei Hu. Meet-in-the-middle attacks revisited: Key-recovery, collision, and preimage attacks. Cryptology ePrint Archive, Paper 2021/427, 2021.

[10] Léo Ducas, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - dilithium: Digital signatures from module lattices. *IACR Cryptol. ePrint Arch.*, page 633, 2017.

[11] Qian Guo, Thomas Johansson, Erik Mårtensson, and Paul Stankovski. Coded-bkw with sieving. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 323–346, Cham, 2017. Springer International Publishing.

[12] Qian Guo, Thomas Johansson, and Paul Stankovski. Coded-BKW: Solving LWE using lattice codes. Cryptology ePrint Archive, Paper 2016/310, 2016.

[13] Jerzy Neyman and Egon Sharpe Pearson. On the Problem of the Most Efficient Tests of Statistical Hypotheses. *Phil. Trans. Roy. Soc. Lond. A*, 231(694-706):289–337, 1933.

[14] Thomas Pornin and Thomas Prest. More Efficient Algorithms for the NTRU Key Generation Using the Field Norm. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography*, volume 11443 of *Lecture Notes in Computer Science*, pages 504–533. Springer, 2019.

[15] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(1):181–199, Aug 1994.

[16] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[17] Katherine E. Stange. Algebraic aspects of solving ring-lwe, including ring-based improvements in the blum–kalai–wasserman algorithm. *SIAM Journal on Applied Algebra and Geometry*, 5(2):366–387, 2021.

[18] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.3)*, 2021.

[19] Yu Wei, Lei Bi, Xianhui Lu, and Kunpeng Wang. Security estimation of lwe via bkw algorithms. *Cybersecurity*, 6(1):24, Sep 2023.