


Recovery of the secret on Binary Ring-LWE problem using random known bits - Extended Version

Reynaldo Caceres Villena   [Universidade de São Paulo | reynaldo@ime.usp.br]

Routo Terada  [Universidade de São Paulo | rt@ime.usp.br]

 Institute of Mathematics and Statistics, Universidade de São Paulo, Rua do Matão, 1010, São Paulo, 05508-090, SP, Brazil.

Received: 22 November 2023 • Accepted: 11 April 2024 • Published: 29 April 2024

Abstract There are cryptographic systems that are secure against attacks by both quantum and classical computers. Some of these systems are based on the Binary Ring-LWE problem which is presumed to be difficult to solve even on a quantum computer. This problem is considered secure for IoT (Internet of Things) devices with limited resources. In Binary Ring-LWE, a polynomial \mathbf{a} is selected randomly and a polynomial \mathbf{b} is calculated as $\mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}$ where the secret \mathbf{s} and the noise \mathbf{e} are polynomials with binary coefficients. The polynomials \mathbf{b} and \mathbf{a} are public and the secret \mathbf{s} is hard to find. However, there are Side Channel Attacks that can be applied to retrieve some coefficients (random known bits) of \mathbf{s} and \mathbf{e} . In this work, we analyze that the secret \mathbf{s} can be retrieved successfully having at least 50% of random known bits of \mathbf{s} and \mathbf{e} .

Keywords: Postquantum cryptography, Ring-LWE problem, Binary Ring-LWE problem, Internet of Things

1 Introduction

The cryptographic community is searching for new quantum-resistant primitives because the main asymmetric cryptosystems such as RSA¹, (EC)DLP² are insecure against a quantum computer. Hence, the National Institute of Standards and Technology (NIST) initiated a process to search new public key cryptographic algorithms which are post-quantum secure [Roy *et al.*, 2016]. Some proposals submitted to NIST process are based on the Ring-Learning-with-Errors (Ring-LWE) problem because its implementation in software and hardware is efficient.

A variant of the Ring-LWE problem, the Binary Ring-LWE problem, has been recently proposed [Buchmann *et al.*, 2016b]. This new variant replaces the delayed Gaussian samplings with a collection of bits sampled from a uniform distribution, providing even more efficient constructions. The Binary Ring-LWE reduces the operand sizes and simplifies modular reduction. Even though these reductions lower the security level of Binary Ring-LWE when compared Ring-LWE, they are effective against conventional and quantum cryptanalysis [Göpfert *et al.*, 2017; Albrecht, 2017; Bogdanov *et al.*, 2007].

In Binary Ring-LWE, a polynomial \mathbf{a} is selected randomly and a polynomial \mathbf{b} is calculated as $\mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}$ where the polynomials \mathbf{s} and \mathbf{e} have binary coefficients. The polynomials \mathbf{b} and \mathbf{a} are public and recovery of the secret \mathbf{s} is associated with hard lattice problems which are hard to solve even in a quantum setting [Wunderer, 2016; Göpfert *et al.*, 2017; Buchmann *et al.*, 2016b]. However, its implementation in software or hardware, especially in IoT devices,

can be vulnerable to Side Channel Attacks [Fan and Verbaauwhede, 2012; Aysu *et al.*, 2018].

A Side Channel Attack (SCA) is any attack based on Side Channel Information that is obtained when protocols or schemes are executed. Some examples are execution time, power consumption, electromagnetic leaks, sound, and other information that is produced while the process is running. These Side Channel Information can be applied to retrieve (hints about) the values of some coefficients (bits) of the secret \mathbf{s} [Buchmann *et al.*, 2016a; Dachman-Soled *et al.*, 2020]. Applying the same concepts, one can recover the bits of the noise polynomial \mathbf{e} (see Appendix A).

This paper is an extended version of Villena and Terada [2023]. We presented a shorter (due to a page limit) previous version of this paper at the 23^o Brazilian Symposium on Information and Computational Systems Security.

1.1 Our Contribution

Due to limited resources on IoT devices, the polynomials \mathbf{s} and \mathbf{e} are unprotected and some bits of \mathbf{s} and \mathbf{e} can be retrieved using SCAs. Analyzing the mathematical properties between the public parameters (polynomials \mathbf{b} and \mathbf{a}), the secret key \mathbf{s} and the noise polynomial \mathbf{e} , we show that all secret \mathbf{s} can be successfully retrieved having at least 50% of bits of \mathbf{s} and \mathbf{e} .

2 Preliminaries

For an integer $q \geq 1$, let \mathbb{Z}_q be the residue class ring modulo q and $\mathbb{Z}_q = \{0, \dots, q-1\}$. Let $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$ denote the polynomial ring modulo $x^n + 1$ where the coefficients are in \mathbb{Z}_q . The operations (addition and multiplication) of elements in \mathcal{R}_q are according to these operations on polyno-

¹Rivest, Shamir and Adleman cryptosystem based on the NP problem of Prime Factoring.

²It is a cryptosystem that uses the Discrete Logarithm Problem over the Elliptic curves.

mials.

For $\mathbf{x} \in \mathcal{R}_q$, let $x[i]$ be the (i) -th coefficient of \mathbf{x} for $0 \leq i < n$. \mathbb{Z}_q^l denotes a set of vectors of length l and their components belong to \mathbb{Z}_q . For $\mathbf{x} \in \mathbb{Z}_q^l$, $x[i]$ denotes the (i) -th component of \mathbf{x} for $0 \leq i < l$. $\{0, 1\}^l$ is a set of strings of length l . For $\mathbf{x} \in \{0, 1\}^l$, $x[i]$ denotes the (i) -th bit of \mathbf{x} for $0 \leq i < l$. For a set \mathcal{S} , $x \leftarrow \mathcal{S}$ denotes that an element x is chosen from \mathcal{S} uniformly at random. For a distribution χ , $x \leftarrow \chi$ denotes that an element x is sampled according to the distribution χ . A polynomial $\mathbf{x} \leftarrow \mathcal{R}_q$ means that each coefficient of \mathbf{x} is chosen randomly from \mathbb{Z}_q . A polynomial $\mathbf{x} \leftarrow \chi^l$ means that each coefficient of \mathbf{x} is chosen randomly according to χ .

The integer $\lfloor x \rfloor$ is defined as $\lfloor x + \frac{1}{2} \rfloor \in \mathbb{Z}$.

2.1 Ring-LWE problem and variants

The Ring-LWE problem has been extensively used as basis of cryptographic schemes proposals due to its hardness and implemented extensively and efficiently in software and hardware [Lyubashevsky *et al.*, 2013; Göttert *et al.*, 2012; Roy *et al.*, 2014; Pöppelmann *et al.*, 2015].

The Ring-LWE problem fixes a size parameter n that is a power of 2, a modulus $q \geq 2$. Define \mathcal{R}_q as the ring $\mathbb{Z}_q[x]/(x^n + 1)$ containing all polynomials over the ring \mathbb{Z}_q in which x^n is identified with -1 . For a secret $\mathbf{s} \in \mathbb{Z}_q^n$, we define the Ring-LWE distribution $A_{n,q,s,\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q^n$, obtained by choosing independently a vector $\mathbf{a} \in \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow \chi^n$ (χ can be a Centered Discrete Gaussian or Centered Binomial distribution). One sample of distribution $A_{n,q,s,\chi}$ is $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e})$, where additions and multiplications are performed in \mathcal{R}_q :

$$\{(\mathbf{a}, \mathbf{b}) \mid \mathbf{a} \leftarrow \mathbb{Z}_q^n, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}, \mathbf{e} \leftarrow \chi^n\} \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n \quad (1)$$

Definition 1 (Ring-LWE oracle) A Ring-LWE oracle $A_{n,q,s,\chi}$ is an oracle which outputs independent random samples according to the $A_{n,q,s,\chi}$ distribution.

There are two versions of Ring-LWE problems:

- **Search Ring-LWE problem:** Given access to a Ring-LWE oracle $A_{n,q,s,\chi}$, find the vector \mathbf{s} .
- **Decision Ring-LWE problem:** The Decision Ring-LWE problem is to distinguish between the uniform distribution over \mathbb{Z}_q^{2n} and the samples given by the oracle $A_{n,q,s,\chi}$.

Using the Ring-LWE problem, some cryptographic and homomorphic encryption schemes were designed [Lyubashevsky *et al.*, 2013; Brakerski *et al.*, 2014].

Ring-LWE problems with access to an oracle $A_{n,q,s,\chi}$, with $\mathbf{s} \in \chi^n$, that produces outputs according to **Equation 1** are called by Small Ring-LWE problems. (Notice that polynomials \mathbf{s} and \mathbf{e} are generated by the same distribution, Centered Discrete Gaussian or Binomial distribution). The use of these distributions on Small Ring-LWE compared to Ring-LWE, decreases the storage space needed for the secret \mathbf{s} ($\mathbf{s} \in \chi^n$ on Small Ring-LWE and $\mathbf{s} \in \mathbb{Z}_q^n$ on Ring-LWE). This reduction, on the secret on Ring-LWE problem, does not compromise the security level offered [Applebaum *et al.*, 2009;

Goldwasser *et al.*, 2010].

Another variant is the Binary Ring-LWE problems. This is a new, promising variant of Ring-LWE that achieves smaller key sizes and more efficient computations [Buchmann *et al.*, 2016b]. The Binary Ring-LWE problems has access to an oracle A_{n,q,s,\mathbb{Z}_2} , with $\mathbf{s} \in \mathbb{Z}_2^n$, and outputs

$$\{(\mathbf{a}, \mathbf{b}) \mid \mathbf{a} \leftarrow \mathbb{Z}_q^n, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}, \mathbf{e} \leftarrow \mathbb{Z}_2^n\} \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n.$$

The security analysis of Binary Ring-LWE was estimated by Buchmann *et al.*. For a parameter set ($n = 126, q = 256$), the decoding failure probability is relatively lower than 2^{-32} , and the security levels achieved against conventional computers and quantum computers were 84-bits and 73-bits, respectively [Buchmann *et al.*, 2016a]. Moreover, Hybrid Attacks³, Quantum Hybrid Attacks and novel variants of the Dual-lattice Attacks applied on Binary Ring-LWE, corroborate that the security level estimated is within the security range determined by Buchmann *et al.* [Wunderer, 2016; Göpfert *et al.*, 2017; Albrecht, 2017].

3 Recovering the secret using random known bits

We have a Binary Ring-LWE instance $\mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}$ ($\mathbf{b} = \sum_{i=0}^{n-1} b_i x^i, \mathbf{a} = \sum_{i=0}^{n-1} a_i x^i, \mathbf{s} = \sum_{i=0}^{n-1} s_i x^i$ and $\mathbf{e} = \sum_{i=0}^{n-1} e_i x^i$). And, some random bits of \mathbf{s} and \mathbf{e} are known (the recovery of these bits can be seen in Appendix A). A Binary Ring-LWE instance $\mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}$ can be written as matrix operations $\mathbf{b}^T = \mathbf{A} \cdot \mathbf{s}^T + \mathbf{e}^T$, (see **Figure 1**). Notice that the coefficients of \mathbf{s} and \mathbf{e} are bits.

$$\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{bmatrix} = \begin{bmatrix} a_0 & -a_{n-1} & \dots & -a_1 \\ a_1 & a_0 & \dots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \dots & a_0 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-1} \end{bmatrix} + \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{n-1} \end{bmatrix}$$

Figure 1. A Ring-LWE sample expressed as matrix operations

Each b_i can be expressed as a system of equations

$$b_i = \sum_{j=0}^i a_{i-j} \cdot s_j - \sum_{j=i+1}^{n-1} a_j \cdot s_{n+i-j} + e_i \quad \text{for } 0 \leq i \leq n-1 \quad (2)$$

It results in n equations with $2n$ variables (bits of \mathbf{s} and \mathbf{e}) which is hard to solve. However, some bits of \mathbf{s} and \mathbf{e} are known.

Let \mathbf{e}_k and \mathbf{e}_u be the sets of known bits and unknown bits of noise polynomial \mathbf{e} ($|\mathbf{e}_k| + |\mathbf{e}_u| = n$). Let \mathbf{s}_k and \mathbf{s}_u be the sets of known bits and unknown bits of the secret polynomial \mathbf{s} ($|\mathbf{s}_k| + |\mathbf{s}_u| = n$). Let α be the percentage of known bits of \mathbf{s} and \mathbf{e} ($|\mathbf{s}_k| + |\mathbf{e}_k| = \alpha \cdot 2n$). Therefore, considering the known bits of \mathbf{e}_k and \mathbf{s}_k we have n equations with $|\mathbf{e}_u| + |\mathbf{s}_u|$

³Attacks that use hybrid lattice reduction and meet-in-the-middle attack. These attacks have been widely applied to evaluate the security of many lattice-based cryptographic schemes.

variables. One condition to have the solution of a system of equations is that the number of variables must be lower than or equal to the number of equations:

$$\begin{aligned} |\mathbf{e}_u| + |\mathbf{s}_u| &\leq n \\ |\mathbf{e}_u| &\leq |\mathbf{s}_k| \quad \text{because } |\mathbf{s}_k| + |\mathbf{s}_u| = n \end{aligned}$$

The above condition is always accomplished since we can set $|\mathbf{e}_u| = 0$. One way to get $|\mathbf{e}_u| = 0$ is discarding all equations in Equations (2) where the value of $e[i]$ is unknown, resulting in $|\mathbf{e}_k|$ equations and $|\mathbf{s}_u|$ variables. This new system of equations needs $|\mathbf{e}_k| \geq |\mathbf{s}_u|$ to be solved.

$$\begin{aligned} |\mathbf{e}_k| + |\mathbf{s}_k| &\geq |\mathbf{s}_u| + |\mathbf{s}_k| && \text{because } |\mathbf{e}_k| \geq |\mathbf{s}_u| \\ |\mathbf{e}_k| + |\mathbf{s}_k| &\geq n && \text{because } |\mathbf{s}_k| + |\mathbf{s}_u| = n \\ \alpha \cdot 2n &\geq n && \text{because } |\mathbf{s}_k| + |\mathbf{e}_k| = \alpha \cdot 2n \\ \alpha &\geq \frac{1}{2} \end{aligned}$$

In other words, we need at least 50% of bits of \mathbf{s} or \mathbf{e} (regardless of whether they are of \mathbf{s} or \mathbf{e}) to retrieve all unknown bits of \mathbf{s} , allowing us to know the actual value of the secret \mathbf{s} . The same analysis can be applied on Ring-LWE and Small Ring-LWE problems obtaining the same percentage required to retrieve successfully the secret \mathbf{s} .

4 Algorithm

An algorithm was implemented using sageMath. This algorithm uses the Gaussian Elimination method to solve equations.

Algorithm 1: UsingKnownCoefficients has as inputs the public parameters \mathbf{b} , the matrix \mathbf{A} that is easily calculated from \mathbf{a} ; and the known coefficients of \mathbf{s} and \mathbf{e} that are saved in the dictionaries \mathbf{s}_k and \mathbf{e}_k , respectively. And, it returns a matrix $\overline{\mathbf{A}}' \in \mathbb{Z}_q^{|\mathbf{e}_k| \times |\mathbf{s}_u|}$ and a vector $\overline{\mathbf{b}} \in \mathbb{Z}_q^{|\mathbf{s}_u|}$ satisfying $\overline{\mathbf{A}}' \cdot \mathbf{x} = \overline{\mathbf{b}}$ where the vector \mathbf{x} contains the unknown coefficients of \mathbf{s} .

The first row of \mathbf{A} is defined by

$$\mathbf{A}[0] = \overline{\mathbf{a}} = \left(\sum_{i=0}^{n-1} -a_i x^{n-1-i} \right) \cdot \mathbf{x}$$

and the i -th row of \mathbf{A} are defined as $\mathbf{A}[i] = \overline{\mathbf{a}}x^i = \mathbf{A}[i-1] \cdot \mathbf{x}$ for $i \in \llbracket 1, n-1 \rrbracket$ (see **Figure 2**).

The i -th row of \mathbf{A} for which e_i are unknown are discarded. For each known value e_i , we save the value $tmp = b_i - e_i$ and subtract the multiplication of s_j by its respectively $\mathbf{A}[i][j]$ if only if s_j is a known coefficient. At the end, the values tmp s are saved in a new vector $\overline{\mathbf{b}}'$ (each known value s_j is used being multiplied to the values in column j of \mathbf{A}).

The Columns j , where s_j are unknown, are saved in a new Matrix $\overline{\mathbf{A}}'$. The Algorithm 1 returns a matrix $\overline{\mathbf{A}}'$ and a vector $\overline{\mathbf{b}}'$ where both are related mathematically $\overline{\mathbf{A}}' \cdot \mathbf{x} = \overline{\mathbf{b}}'$ where the vector \mathbf{x} contains the unknown coefficients of \mathbf{s} . The vector \mathbf{x} can be easily calculated using Gaussian Elimination (sageMath library). The code is below and the command $\text{GF}(q)$ specifies that all operations are executed in a finite ring modulo q .

Algorithm 1: UsingKnownCoefficients

```

Input :  $\mathbf{b} \in \mathcal{R}_q, \mathbf{A} \in \mathbb{Z}_q^{n \times n}, \mathbf{s}_k, \mathbf{e}_k;$ 
Output :  $\overline{\mathbf{A}}' \in \mathbb{Z}_q^{|\mathbf{e}_k| \times |\mathbf{s}_u|}, \overline{\mathbf{b}} \in \mathbb{Z}_q^{|\mathbf{s}_u|};$ 
1  $\overline{\mathbf{A}}' = [];$ 
2  $\overline{\mathbf{b}} = [];$ 
3 for  $i = 0$  to  $n$  do
4   if  $e_i$  in  $\mathbf{e}_k$  then
5      $tmp = b_i - e_i;$ 
6      $\overline{\mathbf{a}}' = [];$ 
7     for  $j = 0$  to  $n$  do
8       if  $s_j$  in  $\mathbf{s}_k$  then
9          $tmp = tmp - \mathbf{A}[i][j] \cdot s_j;$ 
10        end
11       else
12          $\overline{\mathbf{a}}'.append(\mathbf{A}[i][j]);$ 
13       end
14     end
15      $\overline{\mathbf{A}}'.append(\overline{\mathbf{a}}');$ 
16      $\overline{\mathbf{b}}.append(tmp);$ 
17   end
18 end
19 return  $\overline{\mathbf{A}}', \overline{\mathbf{b}};$ 

```

```

1  $\overline{\mathbf{A}}' = \text{matrix}(\text{GF}(q), \overline{\mathbf{A}}');$ 
2  $\mathbf{s}_u = \overline{\mathbf{A}}'.\text{solve\_right}(\text{vector}(\text{GF}(q)), \overline{\mathbf{b}});$ 

```

Proof of the Algorithm: Let \mathbf{s}'_k and \mathbf{s}'_u be two vectors defined as follows:

$$\mathbf{s}'_k[i] = \begin{cases} \mathbf{s}_k[i] & \text{if } \mathbf{s}_k[i] \text{ is known} \\ 0 & \end{cases}$$

and $\mathbf{s}'_u[i] = \mathbf{s}[i] - \mathbf{s}'_k[i]$. Therefore $\mathbf{s}'_k + \mathbf{s}'_u = \mathbf{s}$. We have $\mathbf{b}^T = \mathbf{A} \cdot \mathbf{s}^T + \mathbf{e}^T$ (see **Figure 1**):

$$\begin{aligned} \mathbf{b}^T &= \mathbf{A} \cdot \mathbf{s}^T + \mathbf{e}^T \\ \mathbf{b}^T &= \mathbf{A} \cdot (\mathbf{s}'_k{}^T + \mathbf{s}'_u{}^T) + \mathbf{e}^T \\ \mathbf{b}^T - \mathbf{A} \cdot \mathbf{s}'_k{}^T - \mathbf{e}^T &= \mathbf{A} \cdot \mathbf{s}'_u{}^T \end{aligned} \quad (3)$$

We can discard the i -th elements of \mathbf{b}^T and \mathbf{e}^T and i -th rows of matrix \mathbf{A} , while maintaining the equality valid (see **Figure 2**). On left side of **Equality 3**, we have the value $\overline{\mathbf{b}}^T$ where $\overline{\mathbf{b}}$ is the value returned by **Algorithm 1**.

We have $\overline{\mathbf{b}}^T = \mathbf{A}' \cdot \mathbf{s}'_u{}^T$ (the rows i are removed from \mathbf{A} if e_i is unknown, the result is saved in \mathbf{A}'). Note that some coefficients in $\mathbf{s}'_u{}^T$ are 0's (the coefficients $\mathbf{s}'_u{}^T[j] = 0$ if s_j is known). These elements 0's can be removed, discarding the columns j in \mathbf{A}' and elements j in $\mathbf{s}'_u{}^T$, resulting in a matrix $\overline{\mathbf{A}}$ and a new vector that contains the unknown coefficients of \mathbf{s} (\mathbf{s}_u) (see **Figure 3**). The vector $\overline{\mathbf{b}}$ and matrix $\overline{\mathbf{A}}$ are returned by the execution of **Algorithm 1**.

$$\begin{bmatrix} b_0 \\ \vdots \\ b_i \\ \vdots \\ b_{n-1} \end{bmatrix} - \begin{bmatrix} a_0 & -a_{n-1} & \dots & -a_1 \\ \vdots & \vdots & \ddots & \vdots \\ a_i & a_{i-1} & \dots & -a_{i+1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \dots & a_0 \end{bmatrix} \mathbf{s}'_k{}^T - \begin{bmatrix} e_0 \\ \vdots \\ e_i \\ \vdots \\ e_{n-1} \end{bmatrix} = \begin{bmatrix} a_0 & -a_{n-1} & \dots & -a_1 \\ \vdots & \vdots & \ddots & \vdots \\ a_i & a_{i-1} & \dots & -a_{i+1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \dots & a_0 \end{bmatrix} \mathbf{s}'_u{}^T$$

Figure 2. Discarding the rows i (in \mathbf{b} , \mathbf{A} and \mathbf{e}) where e_i is unknown.

$$\bar{\mathbf{b}}^T = \begin{bmatrix} a_0 & \dots & -a_{n-j+1} & \dots & -a_1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_i & \dots & a_{n-j+i+1} & \dots & -a_{i+1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n-1} & \dots & a_{n-j} & \dots & a_0 \end{bmatrix} \begin{bmatrix} s_0 \\ \vdots \\ \theta \\ \vdots \\ s_{n-1} \end{bmatrix}$$

Figure 3. Discarding the rows j (in matrix \mathbf{A}') and elements j (in $\mathbf{s}'_u{}^T$) where s_j is known.

5 Experiments and results

It was executed on a processor Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz with 3 Mb of cache and 8 GB of DDR4 Memory. The code is available online at <https://github.com/reynaldocv/Recover-Secret-On-Ring-LWE/>. We work in four experimental sets: the first, Binary Ring-LWE ($n = 256, q = 256$) where $\mathbf{s}, ppE \in \mathbb{Z}_2^n$ (parameters defined in Aysu *et al.* [2018]). The following three experimental sets work with parameters ($n = 1024, q = 12289$)⁴ for the systems Binary Ring-LWE ($\mathbf{s}, \mathbf{e} \in \mathbb{Z}_2^n$), Small Ring-LWE (where $\mathbf{s}, \mathbf{e} \in \psi_8^n$), and Ring-LWE (where $\mathbf{s} \in \mathbb{Z}_q^n, ppE \in \psi_8^n$). ψ_8 is a Centered Binomial distribution with a standard deviation of 8, The samples generated are in $\llbracket -8, 8 \rrbracket$. For each experimental set and for each value $\alpha \in \llbracket 48, 60 \rrbracket$, 10000 instances (\mathbf{b}, \mathbf{a}) were generated with α percentage of known coefficients of \mathbf{s} and \mathbf{e} (saved in \mathbf{s}_k and \mathbf{e}_k ,

respectively). In total, 540000 experiments were executed. **Figure 4** shows the number of secrets \mathbf{s} retrieved using α fraction of coefficients of \mathbf{s} and \mathbf{e} . In the experiments with parameters ($n = 1024$ and $q = 12289$), all 33000 secrets \mathbf{s} were retrieved successfully for $\alpha \geq 50\%$. And, no secret can retrieved for $\alpha < 50\%$. Moreover, some secrets were retrieved for Binary Ring-LWE with parameters ($n = 256$ and $q = 256$) with $\alpha < 50\%$. Exactly 4 and 157 secrets were retrieved for $\alpha = 48\%$ and $\alpha = 49\%$, respectively. For $\alpha < 50\%$ we have more equations than variables ($|\mathbf{e}_k|$ equations and $|\mathbf{s}_u|$ variables, $|\mathbf{e}_k| < |\mathbf{s}_u|$). And, The Gaussian Elimination method (defined in sageMath) calculates the first $|\mathbf{e}_k|$ unknown coefficients of \mathbf{s}_u assuming that the remaining coefficients are all 0's. This corresponds to 5 coefficients for $\alpha = 48\%$, and 3 coefficients for $\alpha = 49\%$; being 0's. This condition was presented in some experiments, therefore the recovery of some secrets was feasible. For $\alpha = 50, 51,$ and 52% with 10000 experiments generated for each one, 6058, 9849, and 9991 secrets were retrieved, respectively. We assumed that is because of parameter $q = 256$ since q is not a prime number producing ambiguous solutions for the unknown coefficients. In all experimental sets, for $\alpha \geq 0.53$, all secrets can retrieved successfully.

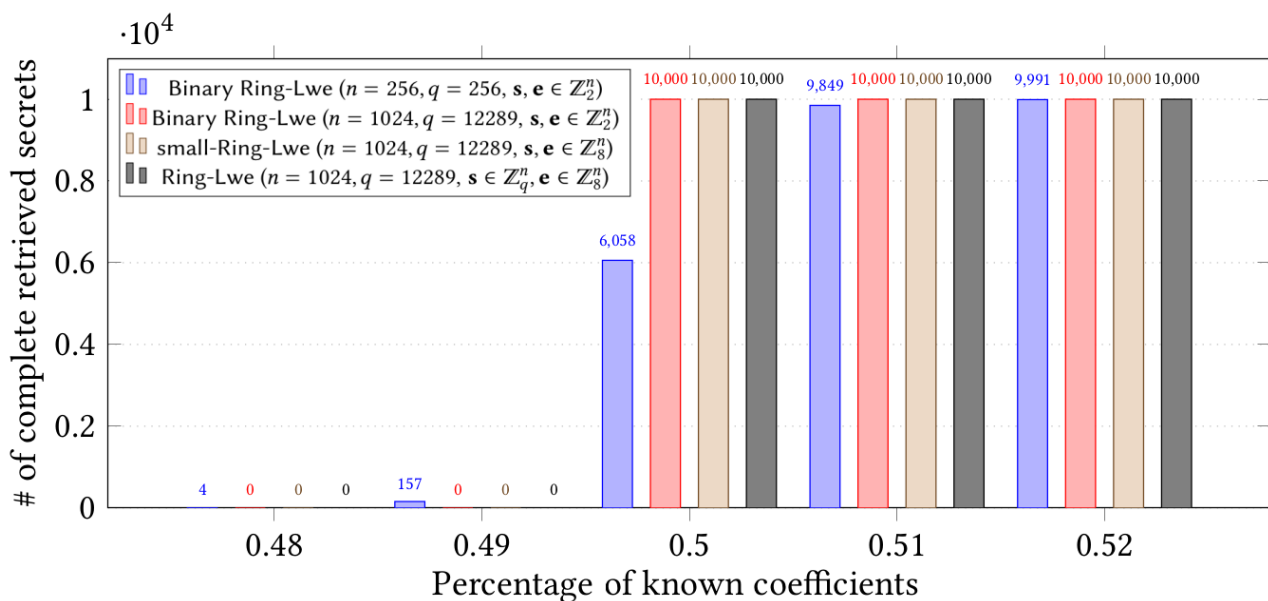


Figure 4. Number of retrieved secrets

⁴Similar parameters specified to NewHope Key Encapsulation Mechanism [Alkim *et al.*, 2016]).

6 Conclusion

We described a scenario where some random bits of the polynomials s and e can be retrieved. Using the mathematical definition of the Binary Ring-LWE problem and these retrieved known bits, the unknown bits of the secret s can be retrieved using the Gaussian Elimination method. Our result shows experimentally that we need at least 50% of random known bits of s and e to retrieve the actual value of the secret s . In other words, with a sufficient number of known bits of s and e , the (Binary) Ring-LWE is a solvable system of equations. This work was extended to Small Ring-LWE and Ring-LWE problems giving us better experimental results. However, we need to retrieve some random known coefficients of s and e , but this task can be more difficult, yet not impossible because the coefficients of s and e are integers, not bits (0 or 1).

As we know, the hardness of the (Binary) Ring-LWE problem is to find s . There are some works focused on the protection of the secret s [Aysu *et al.*, 2018] and the polynomial e is left out since e is only used one time (in the **KeyGen** process, see Appendix A). We must be more careful with the noise polynomial e because the recovery of its coefficient makes the (Binary) Ring-LWE problem weaker.

References

- Albrecht, M. R. (2017). On dual lattice attacks against small-secret lwe and parameter choices in helib and seal. In *Advances in Cryptology—EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30–May 4, 2017, Proceedings, Part II*, pages 103–129. Springer. DOI: 10.1007/978-3-319-56614-6₄.
- Alkim, E., Ducas, L., Pöppelmann, T., and Schwabe, P. (2016). Newhope without reconciliation. <https://eprint.iacr.org/2016/1157>.
- Applebaum, B., Cash, D., Peikert, C., and Sahai, A. (2009). Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology—CRYPTO 2009: 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16–20, 2009. Proceedings*, pages 595–618. Springer. DOI: 10.1007/978-3-642-03356-8₃₅.
- Aysu, A., Orshansky, M., and Tiwari, M. (2018). Binary ring-lwe hardware with power side-channel countermeasures. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1253–1258. IEEE. DOI: 10.23919/DATE.2018.8342207.
- Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J., Seurin, Y., and Vikkelsoe, C. (2007). Present: An ultra-lightweight block cipher. In *Cryptographic Hardware and Embedded Systems—CHES 2007: 9th International Workshop, Vienna, Austria, September 10–13, 2007. Proceedings 9*, pages 450–466. Springer. DOI: 10.1007/978-3-540-74735-2₃₁.
- Brakerski, Z., Gentry, C., and Vaikuntanathan, V. (2014). (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36. DOI: 10.1145/2090236.2090262.
- Buchmann, J., Göpfert, F., Güneysu, T., Oder, T., and Pöppelmann, T. (2016a). High-performance and lightweight lattice-based public-key encryption. In *Proceedings of the 2nd ACM international workshop on IoT privacy, trust, and security*, pages 2–9. DOI: 10.1145/2899007.2899011.
- Buchmann, J., Göpfert, F., Player, R., and Wunderer, T. (2016b). On the hardness of lwe with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack. In *Progress in Cryptology—AFRICACRYPT 2016: 8th International Conference on Cryptology in Africa, Fes, Morocco, April 13–15, 2016, Proceedings*, pages 24–43. Springer. DOI: 10.1007/978-3-319-31517-1₂.
- Dachman-Soled, D., Ducas, L., Gong, H., and Rossi, M. (2020). Lwe with side information: Attacks and concrete security estimation. *Cryptology ePrint Archive*, Paper 2020/292. Available at: <https://eprint.iacr.org/2020/292>.
- Fan, J. and Verbaauwhede, I. (2012). An updated survey on secure ecc implementations: Attacks, countermeasures and cost. *Cryptography and Security: From Theory to Applications: Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday*, pages 265–282. DOI: 10.1007/978-3-642-28368-0₁₈.
- Goldwasser, S., Kalai, Y. T., Peikert, C., and Vaikuntanathan, V. (2010). Robustness of the learning with errors assumption. Available at: <https://web.eecs.umich.edu/~cpeikert/pubs/robustlwe.pdf>.
- Gong, D.-S. D. D. L. and Ristenpart, H. R. M. M. D. (2020). T lwe with side information: attacks and concrete security estimation. In *Advances in Cryptology—CRYPTO*, volume 2020. Available at: <https://eprint.iacr.org/2020/292>.
- Göpfert, F., van Vredendaal, C., and Wunderer, T. (2017). A hybrid lattice basis reduction and quantum search attack on lwe. In *Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26–28, 2017, Proceedings 8*, pages 184–202. Springer. DOI: 10.1007/978-3-319-59879-6₁₁.
- Göttert, N., Feller, T., Schneider, M., Buchmann, J., and Huss, S. (2012). On the design of hardware building blocks for modern lattice-based encryption schemes. In *Cryptographic Hardware and Embedded Systems—CHES 2012: 14th International Workshop, Leuven, Belgium, September 9–12, 2012. Proceedings 14*, pages 512–529. Springer. DOI: 10.1007/978-3-642-33027-8₃₀.
- Lyubashevsky, V., Peikert, C., and Regev, O. (2013). On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):1–35. DOI: 10.1007/978-3-642-13190-5₁.
- Pöppelmann, T., Oder, T., and Güneysu, T. (2015). High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers. In *International conference on cryptology and information security in Latin America*, pages 346–365. Springer. DOI: 10.1007/978-3-319-22174-8₁₉.
- Roy, S. S., Karmakar, A., and Verbaauwhede, I. (2016). Ring-lwe: applications to cryptography and their efficient realization. In *Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE*

2016, Hyderabad, India, December 14-18, 2016, *Proceedings 6*, pages 323–331. Springer. DOI: 10.1007/978-3-319-49445-6_18.

Roy, S. S., Vercauteren, F., Mentens, N., Chen, D. D., and Verbaauwhede, I. (2014). Compact ring-lwe cryptoprocessor. In *Cryptographic Hardware and Embedded Systems—CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16*, pages 371–391. Springer. DOI: 10.1007/978-3-662-44709-3_21.

Villena, R. C. and Terada, R. (2023). Recovery of the secret on binary ring-lwe problem using random known bits. In *Anais do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. Sociedade Brasileira de Computação (short paper)*. DOI: 10.5753/sb-seg.2023.233103.

Wunderer, T. (2016). Revisiting the hybrid attack: Improved analysis and refined security estimates. *Cryptology ePrint Archive*, Paper 2016/733. Available at: <https://eprint.iacr.org/2016/733>.

A Recovering random known bits

In this section, we describe one Public Key Encryption Scheme based on Binary Ring-LWE problem and the recovery of some bits of the secret and noise polynomials is explained.

A.1 Binary Ring-LWE Public Key Encryption Scheme

Binary Ring-LWE Public Key Encryption was proposed by Lyubashevsky *et al.* [2013]. The algorithms are shown in **Figure 5** and described below.

- **KeyGen**: Key generation sets a polynomial $\mathbf{a}' \in \mathcal{R}_q$ and samples two polynomials $\mathbf{r}_1, \mathbf{r}_2 \in \{0, 1\}^n$ and compute $\mathbf{p} = \mathbf{r}_1 - \mathbf{r}_2 \cdot \mathbf{a}' \in \mathcal{R}_q$. The public key is $\mathbf{pk} = \langle \mathbf{p}, \mathbf{a}' \rangle$ and the private key (secret key) is $\mathbf{sk} = \langle \mathbf{r}_2 \rangle$.
- **Encrypt**: Firstly, three polynomials $\mathbf{e}_1, \mathbf{e}_2$ and \mathbf{e}_3 are selected uniformly random over \mathbb{Z}_2^n . The ciphertext is the pair of polynomials $\mathbf{c}_1 = \mathbf{a}' \cdot \mathbf{e}_1 + \mathbf{e}_2$ and $\mathbf{c}_2 = \mathbf{p} \cdot \mathbf{e}_1 + \mathbf{e}_3 + \overline{\mathbf{m}} \in \mathcal{R}_q$. The value of $\overline{\mathbf{m}}$ is obtained by multiplying each coefficient of message \mathbf{m} with $\lfloor \frac{q}{2} \rfloor$.
- **Decrypt**: This algorithm reconstructs the message \mathbf{m} by using the secret key $\mathbf{sk} = \langle \mathbf{r}_2 \rangle$. It Computes $\mathbf{m}' = \mathbf{c}_1 \cdot \mathbf{r}_2 + \mathbf{c}_2$ and decodes the coefficients of \mathbf{m}' using the threshold decoder $\mathbf{Th}(\cdot)$. Each coefficient of \mathbf{m}' is processed separately, returning a binary value, if $\mathbf{m}'[i]$ lies in the range $\lfloor \frac{q}{4}, \frac{3q}{4} \rfloor$, then the value of $\mathbf{m}[i]$ is 1 else the value of $\mathbf{m}[i]$ is 0. The threshold decoder $\mathbf{Th}(\cdot)$ can be defined as $\mathbf{Th}(x) = \lfloor 2 \cdot x / q \rfloor \pmod{2}$.

The Binary Ring-LWE scheme was proposed for Lightweight applications (e.g. constrained IoT nodes). The security level achieved is 84 bits against conventional computers and 73 bits against quantum computers [Wunderer, 2016; Göpfert *et al.*, 2017]. This scheme was implemented in hardware using a configuration that sets

```

1 function KEYGEN( $n$ )
2    $\mathbf{a}', \mathbf{r}_1, \mathbf{r}_2 \leftarrow_{\$} R_q, \{0, 1\}^n, \{0, 1\}^n;$ 
3    $\mathbf{p} \leftarrow \mathbf{r}_1 - \mathbf{a}' \cdot \mathbf{r}_2;$ 
4   return  $\mathbf{pk} = \langle \mathbf{p}, \mathbf{a}' \rangle, \mathbf{sk}' = \mathbf{r}_2;$ 

```

```

1 function ENCRYPT( $\mathbf{pk} = \langle \mathbf{p}, \mathbf{a}' \rangle, \mathbf{m}$ )
2    $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3 \leftarrow_{\$} \{0, 1\}^n, \{0, 1\}^n, \{0, 1\}^n;$ 
3    $\mathbf{c}_1 \leftarrow \mathbf{a}' \cdot \mathbf{e}_1 + \mathbf{e}_2;$ 
4    $\overline{\mathbf{m}} = \lfloor \frac{m}{2} \rfloor, \mathbf{m};$ 
5    $\mathbf{c}_2 \leftarrow \mathbf{p} \cdot \mathbf{e}_1 + \mathbf{e}_3 + \overline{\mathbf{m}};$ 
6   return  $(\mathbf{c}_1, \mathbf{c}_2);$ 

```

```

1 function DECRYPT( $\mathbf{c}_1, \mathbf{c}_2, \mathbf{pk} = \langle \mathbf{r}_2 \rangle$ )
2    $\mathbf{m}' \leftarrow \mathbf{c}_1 \cdot \mathbf{r}_2 + \mathbf{c}_2;$ 
3    $\mathbf{m} \leftarrow \mathbf{Th}(\mathbf{m}');$ 
4   return  $\mathbf{m};$ 

```

Figure 5. Binary Ring-LWE Public Key Encryption

$n = 256$ and $q = 256$ [Aysu *et al.*, 2018].

In Binary Ring-LWE public key encryption scheme, the public key $\mathbf{pk} = \langle \mathbf{p}, \mathbf{a}' \rangle$ and the secret key $\mathbf{sk} = \langle \mathbf{r}_2 \rangle$ are mathematically related $\mathbf{p} = \mathbf{r}_1 - \mathbf{r}_2 \cdot \mathbf{a}'$, and it can be expressed as a Binary Ring-LWE instance $\mathbf{b} = \mathbf{s} \cdot \mathbf{a} + \mathbf{e}$ with $\mathbf{b} = \mathbf{p}, \mathbf{s} = \mathbf{r}_2, \mathbf{e} = \mathbf{r}_1$, and $\mathbf{a} = -\mathbf{a}'$.

We know the value of \mathbf{r}_1 is used only in the **KeyGen** process, therefore we can retrieve some bits of \mathbf{r}_1 applying a SCA when the **KeyGen** process is executed. As we know, the value \mathbf{p} is defined as $\mathbf{p} = \mathbf{r}_1 - \mathbf{r}_2 \cdot \mathbf{a}$. Firstly, the value $\mathbf{r}_2 \cdot \mathbf{a}$ is calculated, and the value of \mathbf{r}_1 is added. Each i -th bit of \mathbf{r}_1 with a value equal to zero, does not modify the value of the bit $(\mathbf{r}_2 \cdot \mathbf{a})[i]$. However, when the i -th bit of \mathbf{r}_1 is one, the value of $(\mathbf{r}_2 \cdot \mathbf{a})[i]$ is altered. This adjustment provokes a power consumption, timing delay, and other information that can be measured, allowing us to differentiate the bits one from zero of \mathbf{r}_1 Aysu *et al.* [2018]. Using other SCA, the absolute value of one coefficient of \mathbf{s} can be retrieved. Therefore the recovery of some bits of \mathbf{s} is feasible since $\mathbf{s} \in \{0, 1\}^n$ [Gong and Ristenpart, 2020]. The value of \mathbf{r}_2 is used in a multiplication operation in **KeyGen** and **Decrypt** processes. By analyzing these multiplications, it is possible to recover the bits of \mathbf{r}_2 . If the bit $\mathbf{r}_2[i]$ is zero no register is modified; else, when the bit $\mathbf{r}_2[i]$ is equal to one, a sum operation is done. This difference can help retrieve the bits of \mathbf{r}_2 . This correlation between the bits and power consumption (Simple Power Analysis and Differential Power Analysis) is studied to retrieve bits of \mathbf{r}_2 . There are methods to avoid it and one lower-cost option to mitigate this vulnerability was using redundant addition and memory update presented in Aysu *et al.* [2018].

Summarizing, the recovery of some bits of $\mathbf{s} = \mathbf{r}_2$ and $\mathbf{e}_1 = \mathbf{r}_1$ is feasible using some Side Channel Attacks.