A note on Ring-LWE security in the case of Fully Homomorphic Encryption

Guillaume Bonnoron[1], Caroline Fontaine[2]

[1] Chair of Naval Cyber Defense – Ecole Navale, CC600, F29240 Brest Cedex 9*
[2] CNRS/Lab-STICC and Telecom Bretagne

guillaume.bonnoron@ecole-navale.fr
caroline.fontaine@telecom-bretagne.eu

April 2016

Abstract

Evaluating the practical security of Ring-LWE based cryptography has attracted lots of efforts recently. Indeed, some differences from the standard LWE problem enable new attacks. In this paper we discuss the security of Ring-LWE as found in Fully Homomorphic Encryption (FHE) schemes. These schemes require parameters of very special shapes, that an attacker might use to its advantage. First we present the specificities of this case and recall state-of-the-art attacks, then we derive a new special-purpose attack. Our experiments show that this attack has unexpected performance and confirm that we need to study the security of special parameters sets carefully.

1 Introduction

The Ring-LWE problem has been introduced by [LPR10] as a ring variant of the Learning With Errors due to Regev [Reg05]. Both problems enjoy security reductions to hard lattice problems, so they serve as hardness grounds for many cryptographic constructions. See the recent survey from Peikert [Pei15] for extensive details. Today, the paramount question, that still stands in the way to practical use, concerns the security of concrete instances of these problems. Namely, how shall one choose parameters for these problems to meet a security level objective, say 80 bits of security?

Homomorphic encryption is a type of encryption that allows to compute with encrypted data. The result, once decrypted, equals that of the same computation done over the plain data. It enables many new applications because one no longer needs to trust the computing entity, e.g cloud service providers. It was first realised by the ground-breaking works of Gentry [Gen09] and Aguilar et al. [AMGH10]. Since then, huge efforts have been spent and have led to numerous scheme proposals ([vDGHV10, BGV12, FV12, GSW13, DS15]). Some of the most efficient ones are now implemented, for early users: HElib [Hal], SEAL [DGBL$^+$15]. These efficient schemes are known secure under the assumption that Ring-LWE is intractable. For an application, the parameters need to be chosen to guarantee the expected security level. This choice of parameters is also constrained so that the scheme itself works.

*Funded and supported by Ecole Navale, Telecom Bretagne, Thales and DCNS

Related work: This need for a better understanding of the Ring-LWE security, *in practice*, has already driven some studies. For exemple, Albrecht et al. [APS15] offer a complete overview of the known attacks against LWE. They also give estimates of their costs against some LWE-based cryptographic schemes, not only FHE. We recall them briefly below for our later discussion. Another more recent line of work has been developed against Ring-LWE specifically, taking advantage of the underlying ring structure, see [Pei16] for a summary and guidelines to draw immune parameters.

Today, the only estimates we can use to choose parameters are from the works of Lindner and Peikert [LP11] and Van de Pol and Smart [vdPS13]. Neither include special-purpose attacks for FHE settings.

Our work: Despite these work, it seems to us that a focus on Ring-LWE-based FHE scheme is needed. As we see later, in order to keep correctness in the scheme, an application designer needs to pick very special parameters. It seemed unclear to us what concrete advantage an attacker might have in such cases. Our contribution aims at filling this gap. After reviewing state-of-the-art attacks, we derive a new one, specially designed for this case, and present experimental results.

Roadmap: Introducing notation and definitions in Section 2, we review the state-of-the-art of the attacks against Ring-LWE in Section 3. In Section 4 we present our new attack in details, its performance in Section 5 and draw some conclusions in Section 6.

2 Preliminaries

2.1 Notation

For a positive integer $q > 0$ we note $\mathbb{Z}_q$ the set of elements $\{0, \ldots, q-1\}$.

For $n > 0$ integer, the matrix $\mathbf{I_n}$ refers to the identity matrix of size n. Capital bold letters are used for matrices, e.g. $\mathbf{B}$, and small ones for (row) vectors, e.g. $\mathbf{v}$. We similarly write $\mathbf{B}$ for a matrix or for the ordered family of (row) vectors $\mathbf{B} = (\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n})$ using bold subscripts. For a vector $\mathbf{v}$, we refer to its components with italic subscripts $\mathbf{v}_i$.

When dealing with a polynomial $\mathbf{p}$ of degree $n-1$, we use the coefficient embedding and assimilate it as a coordinate vector: $\mathbf{p} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n)$. We will work with polynomials in polynomial ring $R = \mathbb{Z}[x]/(f(x))$ for some $f \in \mathbb{Z}[x]$. We denote R_q the set of polynomials in R with coefficients in $\mathbb{Z}_q$.

2.2 Lattices

General definitions. In general, a lattice $\mathcal{L}$ of dimension n is a discrete additive subgroup of $\mathbb{R}^n$. *Integer* lattices are discrete additive subgroups of $\mathbb{Z}^n$. In this paper we only work with the integer lattices and simply call them lattices.

Lattices (of size n) are usually represented by a basis $\mathbf{B}$, a set of n independent integer vectors $(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n})$ of size n whose integer linear combinations generate the lattice.

$$\mathcal{L}(\mathbf{B}) = \left\{ \sum_{i=1}^{n} v_i \mathbf{b_i} : v_i \in \mathbb{Z} \right\} = \left\{ \mathbf{B}^T \mathbf{v} : \mathbf{v} \in \mathbb{Z}^n \right\} = \mathbf{B}\mathbb{Z}^n$$

In our lattices, $\mathbf{B}$ is always a square integer matrix, $\mathbf{B} \in \mathbb{Z}^{n \times n}$. For most of the discussion we restrict to this full rank definition and will make it explicit when working with greater generating families.

An invariant of a lattice is its determinant $\det(L)$. It is defined as the absolute value of the determinants of its bases.

$$\det(L) = |\det(\mathbf{B})|$$

Gram-Schmidt Orthogonalization (GSO). We refer several times to the GSO of a basis. This algorithm takes the matrix to orthogonalize and outputs the resulting matrix $\mathbf{B}^\star$ and a matrix μ (lower triangular with 1 on the diagonal) such that : $\mathbf{B} = \mu \times \mathbf{B}^\star$. We construct $\mathbf{B}^\star$ so that its vectors verify:

- $\mathbf{b_1^\star} = \mathbf{b_1}$
- $\mathbf{b_i^\star}$ is the projection of $\mathbf{b_i}$ orthogonally to the subspace generated by the $i-1$ first vectors of $\mathbf{B}$.

It works in polynomial time in the size of the matrix.

q-ary lattices. When studying LWE and Ring-LWE problems, the lattices we mostly work with are called *q-ary*, because they are defined *modulo* some integer q (not necessarily prime). These lattices are defined as follows:

$$\begin{aligned}\mathcal{L}_q(\mathbf{B}) &= \left\{ \sum_{i=1}^{n} v_i \mathbf{b_i} \bmod q : v_i \in \mathbb{Z} \right\} \\ &= \left\{ \mathbf{B}^T \mathbf{v} \bmod q : \mathbf{v} \in \mathbb{Z}^n \right\}\end{aligned}$$

Hence, since we are working modulo q, we can equivalently consider that all the components of $\mathbf{B}$ and $\mathbf{v}$ are in $\mathbb{Z}_q$.

Lattice reduction. As we discuss below, working with lattices is better when we have *nice* bases. Therefore we have lattice reduction algorithms that, given a basis, make it nicer according to some criteria. There is a wealth of studies on this computational problem.

We usually consider the following criteria and say that a basis is:

- *η-size-reduced* if $\forall i < j, \left|(\mathbf{b_j}|\mathbf{b_i^\star})\right| \leq \eta \,.\left\|\mathbf{b_i^\star}\right\|^2$
- *δ-LLL-reduced* if it is size-reduced and $\forall i, \delta\left\|\mathbf{b_i^\star}\right\|^2 \leq \left(\left\|\mathbf{b_{i+1}^\star}\right\|^2 + \frac{(\mathbf{b_{i+1}}|\mathbf{b_i^\star})^2}{\|\mathbf{b_i}^\star\|^2}\right)$
- *β-BKZ-reduced* if it is LLL-reduced and for all j, $\mathbf{b_j^\star}$ is the shortest vector of the sublattice spanned by $(\mathbf{b_j}, \ldots, \mathbf{b_k})$ with $k = \min(j+\beta-1, n)$

We have algorithms to achieve such reductions:

- The celebrated LLL algorithm from Lenstra, Lenstra and Lovász [LLL82], running in polynomial time of the dimension and the size of the elements. Improvements have been proposed since then and are available in current implementations [NS05].
- The Blockwise Korkine-Zolotarev algorithm [SE94, CN11] achieves BKZ reduction. It makes a polynomial (in n) call to a SVP oracle in a sublattice of size β. It behaves roughly as a sub-exponential in the basis quality [GN08b] and uses LLL as a subroutine.

– Slide Reduction [GN08a] is another block algorithm, in the spirit of BKZ, but simpler and whose performance approaches that of BKZ [MW15].

The quantity we usually use to measure the reduction quality, independently of the algorithm, is the *root Hermite factor* γ. It is defined such as : $\|\mathbf{b_1}\| = \gamma^n \det(L)^{1/n}$ where $\mathbf{b_1}$ is the smallest vector of the basis we qualify. The higher the quality, the smaller it gets.

2.3 Learning With Errors and Ring Variant

We recall here the definitions of the Learning With Errors problem [Reg05] and the Ring-LWE variant [LPR10]. Both exist in a *search* version and a *decision* version.

Definition LWE. *Let n, q be positive integers, χ a probability distribution on $\mathbb{Z}$ of standard deviation σ and $\mathbf{s}$ a secret random vector in $\mathbb{Z}_q^n$. We denote by $L_{\mathbf{s},\chi}$ the probability distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \in \mathbb{Z}$ according to χ and considering it in $\mathbb{Z}_q$, and returning $(\mathbf{a}, c) = (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.*

Decision-LWE *is the problem of deciding whether given pairs $(\mathbf{a}, c)$ are sampled according to $L_{\mathbf{s},\chi}$ or the uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$*
Search-LWE *is the problem of recovering $\mathbf{s}$ from pairs $(\mathbf{a}, c)$ sampled from $L_{\mathbf{s},\chi}$.*

Definition Ring-LWE. *Let R be a ring of degree n over $\mathbb{Z}$ (usually $R = \mathbb{Z}[x]/(f(x))$ for some cyclotomic polynomial $f(x)$). Let q be a positive integer, χ a probability distribution on R of standard deviation σ and $\mathbf{s}$ a secret random element in R_q. We denote by $L_{\mathbf{s},\chi}$ the probability distribution on $R_q \times R_q$ obtained by choosing $\mathbf{a} \in R_q$ uniformly at random, choosing $\mathbf{e} \in R$ according to χ and considering it in R_q, and returning $(\mathbf{a}, \mathbf{c}) = (\mathbf{a}, [\mathbf{a} \cdot \mathbf{s} + \mathbf{e}]_q) \in R_q \times R_q$.*

Decision-Ring-LWE *is the problem of deciding whether given pairs $(\mathbf{a}, \mathbf{c})$ are sampled according to $L_{\mathbf{s},\chi}$ or the uniform distribution on $R_q \times R_q$.*
Search-Ring-LWE *is the problem to recovering $\mathbf{s}$ from pairs $(\mathbf{a}, \mathbf{c})$ sampled from $L_{\mathbf{s},\chi}$.*

The hardnesses of (Ring-)LWE problems depend on the three variables n, σ and q. The hardness reductions presented in the introductory papers stands when $\sigma > 2\sqrt{n}$. Besides that, we aim to establish a link between a choice of parameters and the provided security level, in the context of FHE.

2.4 Ring-LWE based FHE schemes

Since the works of Gentry [Gen09] and Aguilar et al. [AMGH10], lots of homomorphic encryption schemes have been proposed. They can be divided into two families: those based on integers and those based on lattices. LWE and Ring-LWE serve as building ground for the latter family. Generally, those based on Ring-LWE derive from an equivalent scheme on LWE and are more efficient in terms of space and/or time. The most common Ring-LWE based schemes are: [BGV12], [FV12], YASHE [BLLN13], SHIELD [KGV15] and F-NTRU [DS15].

For expository purpose, we recall here only elements of the [FV12] scheme. The discussion remains valid for all schemes as well. The interested reader should refer to the original papers for extensive details about the schemes. In our present study case, we are interested in the elements that will be the attack target, namely the public key. In [FV12] the key generation process goes as follows:

1. `FV.ParameterChoice(`λ`)`: choose (n, σ, q) to guarantee of level of security λ and set $R = \mathbb{Z}[x]/(\Phi_n(x))$
2. `FV.KeyGen(`n, σ, q`)`: sample $\mathbf{s} \leftarrow R_2$, $\mathbf{a} \leftarrow R_q$, $\mathbf{e} \leftarrow \chi$ and output

$$\mathtt{sk} = \mathbf{s} \text{ and } \mathtt{pk} = ([-(\mathbf{a} \cdot \mathbf{s} + \mathbf{e})]_q, \mathbf{a})$$

Security. Let aside the sign of the first element, the public key `pk` is exactly a Ring-LWE pair as described above. The objective of the attacker is to solve *search-Ring-LWE* (i.e. find $\mathbf{s}$) when given access to this public key and the public parameters (n, σ, q and R).

Correctness. The condition for the scheme to be correct (i.e. decryption yields a result consistent with the computation) is [FV12, Eq. 6]:

$$4 \cdot \beta(\epsilon) \cdot \delta_R^{L_{\max}} \cdot (\delta_R + 1.25)^{L_{\max}+1} \cdot t^{L_{\max}-1} < \frac{q}{\sigma}$$

where

- $L_{\max}$ is the maximum multiplicative depth before bootstrapping is needed.
- t is the size of the plaintext space (R_t).
- $\delta_R = \max\{\|\mathbf{a} \cdot \mathbf{b}\| / (\|\mathbf{a}\| \cdot \|\mathbf{b}\|) : \mathbf{a}, \mathbf{b} \in R\}$, e.g. $\delta_{\mathbb{Z}[x]/(x^n+1)} = n$.
- $\beta(\epsilon)$ is such that the error samples are bounded by $B = \beta(\epsilon) \cdot \sigma$ with probability $1 - \epsilon$, e.g. $\beta(2^{-64}) \approx 9.2$.

As a result of this constraint, we can see that q will have to be very large, in front of σ, to conserve correctness with interesting depth, say $L = 9$. To prevent q from being enormous, many authors tend to take σ very tiny, e.g. 3.2 or 8. However, this completely violates the bound $\sigma > 2\sqrt{n}$ required for the hardness reductions to hold [LPR10].

3 Existing Attacks

In January 2015, Albrecht et al. [APS15] aimed at assessing the concrete hardness of LWE and provided an excellent survey of the state-of-art approaches as of this date. These methods apply also to Ring-LWE. We recall here, from a high level perspective, the different families of methods, together with more recent attacks when such exist. The sections of their paper contains extensive details and are therefore mentioned here as reference for the interested reader. Then we briefly present the attacks targeting Ring-LWE specifically and summarized in the recent survey [Pei16].

Bruteforce on s. Attempting an exhaustive search is always possible, yet rarely efficient to solve the problem if the parameter choice is sound. For completeness they express the time complexity of such attack, in general [APS15, Sect 5.1] and in the case where $||\mathbf{s}|| \leq 1$ [APS15, Sect 6.1]. There are better performing methods, as follow.

Distinction. In [APS15, Sect 5.3], they present a way to distinguish LWE samples from uniformly random samples, as stated by the Decision-LWE problem. However this does not directly recover the secret.

Arora-Ge Attack [AG11]. A purely algebraic attack [APS15, Sect 5.6 and 6.5], this one consists in constructing from LWE samples a polynomial whose root is the secret. It is particularly efficient in cases where $||\mathbf{e}||$ is very small, for example binary errors. For this case however, we are outside of the domain of validity of the hardness reduction from [Reg05] which requires $\sigma > 2\sqrt{n}$.

Blum-Kalai-Wasserman [BKW03]. This combinatorial method works like the Gauss elimination procedure [APS15, Sect 5.2 and 6.4]. It requests many samples, searches collisions between parts of these samples and gradually creates linear combinations where more and more components equal 0. At the end, the combination allows to deduce the secret. This method has been improved a lot since its original presentation. To date, the best is from Guo et al [GJS15]. Due to the number of samples required, we do not consider this attack further. Indeed we prefer to assume availability of only the public key and see what can be done.

Decoding. In LWE samples, Gaussian errors can be considered bounded, with high probability, by a sufficiently large bound. An approach is then to use decoding algorithms to remove the error. This addresses the LWE problem as a *Bounded-Distance Decoding (BDD)* instance in a specially crafted lattice. The BDD problem asks to recover the lattice point closest to a given one (not in the lattice), with the promise that the target point is close to the lattice [APS15, Sect 5.4].

Decoding with Embedding. Another technique to cope with the difficulties of direct decoding is to embed the lattice into another. The latter has higher dimension and more properties that enable some optimizations. Two embeddings are presented: one in the general case [APS15, Sect 5.5] by Kannan [Kan87] and one in the specific case of $||\mathbf{s}|| \leq 1$ [APS15, Sect 6.3] from [BG14].

Using the Ring Structure. Several recent works [EHL14, ELOS15, CLS15, CIV16] use the underlying ring structure to attack Ring-LWE. Peikert describe a unified framework that encompasses all these attacks and sort them into two classes: reduction to errorless LWE and reduction modulo an ideal for which decision-Ring-LWE is tractable [Pei16]. Consequently, we know some rings to be vulnerable and others to be immune.

4 Our New Attack

Out of these surveys, we conclude to keep the decoding attacks. Algebraic attacks are not usable within the bounds of hardness reductions [Reg05, LPR10] and combinatorial attacks require too many samples, we let them aside. In the case of [FV12], the secret is small, $||\mathbf{s}|| \leq 1$, so when it comes to embedding, the solution [BG14] is best.

4.1 Bai-Galbraith Embedding Improved

We detail here the original idea of this embedding together with our improvement.

Let $(\mathbf{A}, \mathbf{b} = \mathbf{A}^T\mathbf{s} + \mathbf{e} \bmod q)$ be a LWE instance for some n, q and σ. They consider the general case where we might have m samples. As we derive the matrix $\mathbf{A}$ from a single Ring-LWE sample (the key), we fix $m = n$, $\mathbf{A} \in \mathbb{Z}^{n \times n}$. Write $\mathbf{A'} = (\mathbf{A}|\mathbf{I_n})$, being a $n \times 2n$

matrix. We have the following equality

$$\mathbf{b} = \mathbf{A}' \begin{pmatrix} \mathbf{s} \\ \mathbf{e} \end{pmatrix} \mod q$$

where $\left(\begin{smallmatrix} \mathbf{s} \\ \mathbf{e} \end{smallmatrix}\right)$ is a short vector with respect to q.

Clearly for $\mathbf{w} = \left(\begin{smallmatrix} 0 \\ \mathbf{b} \end{smallmatrix}\right)$ we have $\mathbf{A}'\mathbf{w} = \mathbf{b}$. We then try to approximate this vector $\mathbf{w}$ by a point $\mathbf{v_0}$ from the lattice $\mathcal{L}'$ defined by

$$\mathcal{L}' = \left\{ \mathbf{v} \in \mathbb{Z}^{2n} : \mathbf{A}'\mathbf{v} = 0 \bmod q \right\}$$

For such a vector $\mathbf{v_0}$, we have $\mathbf{v_0} \approx \mathbf{w}$ so $\mathbf{w} - \mathbf{v_0}$ is a short vector (with respect to q) and

$$\begin{aligned} \mathbf{A}'(\mathbf{w} - \mathbf{v_0}) \bmod q &= \mathbf{A}'\mathbf{w} - \mathbf{A}'\mathbf{v_0} \bmod q \\ &= \mathbf{b} - 0 \bmod q \\ &= \mathbf{b} \end{aligned}$$

Consequently we have $\mathbf{w} - \mathbf{v_0} = \left(\begin{smallmatrix} \mathbf{s} \\ \mathbf{e} \end{smallmatrix}\right)$. Hence, finding $\mathbf{v_0}$ in $\mathcal{L}'$ allows us to recover $\mathbf{s}$ (and also $\mathbf{e}$). We can use BDD algorithms for this, if we have a basis $\mathbf{B}$ for $\mathcal{L}'$. We obtain it by embedding as follows:

$$\mathbf{B}^T = \begin{pmatrix} \mathbf{I_n} & \mathbf{0} \\ -\mathbf{A} & q\mathbf{I_n} \end{pmatrix} \in \mathbb{Z}^{2n \times 2n}$$

It can be verified that the columns of this matrix are linearly independent and each of them satisfies the definition of $\mathcal{L}'$, so $\mathbf{B}$ is a basis of $\mathcal{L}'$.

This embedding differs slightly from what Bai and Galbraith presented. They instead introduced the matrix

$$\mathbf{M} = \left(\begin{array}{c|c} \mathbf{I_n} & \\ & q\mathbf{I_{2n}} \\ -\mathbf{A} & \end{array} \right) \in \mathbb{Z}^{2n \times 3n}$$

and compute its column Hermite Normal Form to end up with a full rank matrix generating the lattice. Our technique avoids this computation and yields a basis for $\mathcal{L}'$ more efficiently. We can see that with linear operations on columns their $\mathbf{M}$ can be transformed into our $\mathbf{B}^T$ as follow:

$$\mathbf{M} = \begin{pmatrix} \mathbf{I_n} & q\mathbf{I_n} & \mathbf{0} \\ -\mathbf{A} & \mathbf{0} & q\mathbf{I_n} \end{pmatrix} \to \begin{pmatrix} \mathbf{I_n} & \mathbf{0} & \mathbf{0} \\ -\mathbf{A} & q\mathbf{A} & q\mathbf{I_n} \end{pmatrix} \to \begin{pmatrix} \mathbf{I_n} & \mathbf{0} & \mathbf{0} \\ -\mathbf{A} & \mathbf{0} & q\mathbf{I_n} \end{pmatrix}$$

Removing the n middle columns gives our matrix.

After the embedding part, we leave untouched the rescaling operation from [BG14]. Since $||\mathbf{s}|| \leq 1$ whereas the standard deviation for $\mathbf{e}$ is σ (which is not small as discussed above), the components of the difference $\mathbf{w} - \mathbf{v_0}$ are unbalanced. Therefore we multiply the n first components of the basis vectors. It makes the lattice reduction easier, the difference more balanced and still short (with respect to q). Consequently the next stage of the attack works in the lattice whose basis is

$$\mathbf{B}^T = \begin{pmatrix} \sigma\mathbf{I_n} & \mathbf{0} \\ -\mathbf{A} & q\mathbf{I_n} \end{pmatrix} \in \mathbb{Z}^{2n \times 2n}$$

To sum up, the ground method for this step is mainly due to Bai and Galbraith. We provided slight modifications to it to avoid Hermite Normal Form computations.

4.2 Lattice Reduction

Once the embedding and rescaling are done, the next step is to find the closest point $\mathbf{v_0}$ to $\mathbf{w}$ in $\mathcal{L}'$. The existing algorithms for BDD require a somehow reduced basis of the lattice, otherwise they have very poor performance. Since our basis $\mathbf{B}$ in its present form verifies none of the criteria for reduction given at the beginning 2.2, we need to reduce it. We detail here the different options and our strategy.

Lattice reduction has been studied for long since it comes to be very useful in cryptanalysis, integer programming, to cite only a few. The first and celebrated work of Lenstra, Lenstra, Lovász [LLL82] presented a polynomial time algorithm for lattice reduction, whose output quality suffices for many applications. Since then, several blockwise algorithms have been introduced, the best performing are BKZ [CN11] and Slide Reduction [GN08a]. Both achieve better output quality than LLL but have worse time complexities. Since BDD becomes quicker with a better reduced basis, we need to find a trade-off between the time spent on the reduction step and on the BDD step, so that the overall attack time is minimized. Therefore, an attacker might attempt to use LLL first and if the BDD step fails, try a stronger algorithm. Same discussion apply when the algorithm is fixed, the attacker can tweak its parameters and achieve different output qualities for different computation times.

The attacker may also look for specialised lattice reduction algorithm since the previous ones work for any lattices and do not take advantage of any structure in them. To the best of our knowledge, the only specialized algorithm that fits our case is the variant from Gama et al. [GHGN06]. The lattices it reduces are *symplectic*. Denoting $J_{2n} = \begin{pmatrix} \mathbf{0} & \mathbf{I_n} \\ \mathbf{-I_n} & \mathbf{0} \end{pmatrix} \in \mathbb{Z}^{2n \times 2n}$, a lattice with basis $\mathbf{B}$ is symplectic if and only if $\mathbf{B}^T J_{2n} \mathbf{B} = J_{2n}$ The bases of our case are indeed symplectic, as can be verified. So this variant of LLL would be interesting to try. The authors claim it brings a speed-up factor of nearly 10 when compared to reference implementation of classical LLL. However their code has not been released, so we were not able to use it in our tests, nor confirm its performance.

In the next section we present the details of our experiments on algorithms and parameters. Our conclusion is that in our case, the BDD step is successful even when we only perform an LLL reduction with quite weak parameters. This fact is absolutely no general conclusion. Here we are in a specific case, our basis $\mathbf{B}$ has several properties (integer, upper triangular, blockwise upper triangular with scaled identity on the diagonal, etc...) and so do our parameter n, σ and q.

4.3 Pruned Enumeration for BDD

Finally, to find $\mathbf{v_0}$ in $\mathcal{L}'$ close to $\mathbf{w}$ with the reduced basis, we use Liu-Nguyen pruned enumeration [LN13]. To our knowledge pruned enumeration is the best performing algorithms in practise, see [HPS11] for a description of the other candidates.

This method of pruned enumeration is an adaptation to BDD of the extreme pruning technique introduced by Gama et al. [GNR10]. This algorithm enumerates lattice points that are close enough to the target, with a bound provided as a *pruning function*. Since the enumeration works by adding one component at a time to the current partial solution, we may define different bounds for each positions. Unlike previous work [Bab86, LP11], Liu-Nguyen pruned enumeration evaluates the heuristic on the *projections* of the current candidate and not on its *components*.

This concludes the algorithmic description of our attack. We turn now to implementation aspects and the results we get.

5 Implementation and Results

In this section we present the implementation details of our new attack, described in the previous section. Since we were able to use it successfully against many [FV12] keys, we thoroughly analyse its performance and give out some conclusions.

5.1 Implementation Details

In order to implement our attack, we use the following existing codes:

- Lepoint's implementation [Lep14] of [FV12],
- Stehlé's `fplll` [ACPS] for lattice reduction algorithms,
- Shoup's `NTL` [Sho15] for additional lattice operations.

Then, we implemented the embedding/rescaling, the pruned enumeration algorithm from Liu-Nguyen (following their pseudo-code [LN13]) and created the glue between the different libraries to lead the attack from beginning to end.

Lepoint's implementation. In [LN14], in order to compare [FV12] and YASHE, Lepoint implemented the code needed to use both schemes and perform homomorphic operations. We use his constructor method with light modification. It allows us to create a public key $\mathtt{pk} = ([-(\mathbf{a}\cdot\mathbf{s}+\mathbf{e})]_q, \mathbf{a})$, given n, σ and q. The attack aims to recover the secret when given the `FVKey` object as input.

From [FV12] to R-LWE lattice. From the $\mathbf{a}$ in the public key, we construct a lattice basis $\mathbf{A}$ so that $[\mathbf{a}\cdot\mathbf{s}+\mathbf{e}]_q$ (polynomial operation) equals $\mathbf{As}+\mathbf{e} \mod q$ (matrix/vector operation). For instance, with $R = \mathbb{Z}[x]/(x^n+1)$ we have:

$$\mathbf{A}^T = \begin{pmatrix} a_1 & a_2 & a_3 & \cdots & a_n \\ -a_n & a_1 & a_2 & \cdots & a_{n-1} \\ -a_{n-1} & -a_n & a_1 & \cdots & a_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_2 & -a_3 & -a_4 & \cdots & a_1 \end{pmatrix}$$

Embedding and rescaling. Then we embed this matrix $\mathbf{A}$ into $\mathbf{B}$ following Section 4.1. Rescaling is also done here. $\mathbf{B}$ is ready for the next stage.

Lattice reduction. For this step we rely on the different routines from `fplll` [ACPS]. Both LLL and BKZ algorithms are available and can be tweaked conveniently. We detail below the settings we experimented.

BDD enumeration. Finally, we implemented the pruned enumeration algorithm from Liu-Nguyen to solve BDD in the reduced lattice $\mathcal{L}'$. Later, we discovered an undocumented implementation of enumeration in `fplll`, that we included in our experiments.

5.2 Attack Settings – Early Benchmarks

Before launching a full-scale attack, we run it on small cases. It allows us to explore the choices we have, for algorithms and their parameters, for the different stages.

5.2.1 Lattice Reduction Tweaking

State-of-the-art results [APS15] tell us that strong reductions like BKZ is needed to make enumeration possible. Yet after a few toy examples we realized that most of the attack time was spent in the lattice reduction stage. So we tried different settings for BKZ and LLL on cases where $n \leq 100$.

Beginning with BKZ, we could confirm [CN11], an intermediate blocksize $\beta = 20$ leads to moderate running times. Whereas smaller values like 5 or greater like 40 lead to prohibitive running times. Even with such blocksize, the attack time is not balanced between reduction and enumeration, so we rapidly shifted to study LLL.

LLL algorithm is governed by two parameters δ and η. We recall the size reduction condition and the Lovász condition from 2.2:

$$\forall i < j, \quad \left|(\mathbf{b_j}|\mathbf{b_i^\star})\right| \leq \eta \,.\left\|\mathbf{b_i^\star}\right\|^2 \qquad \text{with } 1/2 \leq \eta \leq \sqrt{\delta}$$

$$\forall i, \quad \delta\left\|\mathbf{b_i^\star}\right\|^2 \leq \left(\left\|\mathbf{b_{i+1}^\star}\right\|^2 + \frac{(\mathbf{b_{i+1}}|\mathbf{b_i^\star})^2}{\left\|\mathbf{b_i^\star}\right\|^2}\right) \qquad \text{with } 1/4 \leq \delta \leq 1$$

The default values in `fplll` are $(\delta, \eta) = (0.99, 0.51)$.

Optimising δ. First, we tried to decrease δ to loosen the Lovász condition, however with (0.75, 0.51), the time improvement is very limited and only for $n \geq 100$, whereas it has dramatic effect on the observed success rate of the overall attack. The enumeration stage fails due to a lattice basis of insufficient quality.

Optimising η. Then we restored δ and experimented with an increased $\eta \in [0.60, 0.98]$, to ease the size-reduction condition.

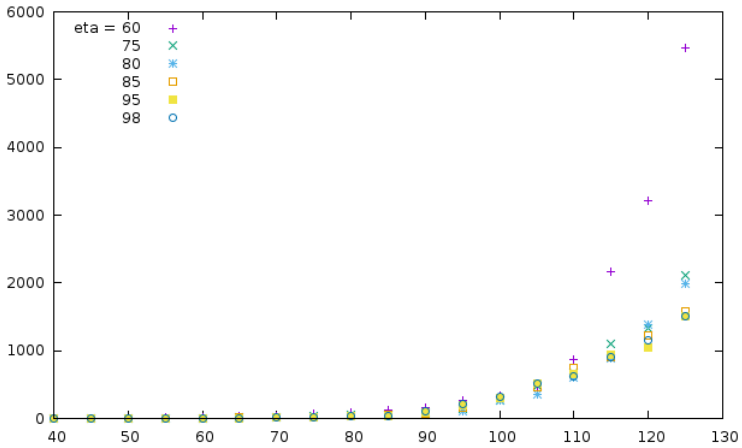

Figure 1: Execution time in seconds in term of n for different η

As we can see in Figure 1, loosening the size reduction condition decreases the running time of LLL. We observe a greater gain (in time) between 0.60 and 0.75 than between 0.75 and 0.95. In addition, with $\eta = 0.95$ we get a less successful attack, whereas for η equal to 0.51 or 0.75 the observed success rates are similar.

γ	$\log_2 q = 47$	$\log_2 q = 95$
$n = 50$	1.00238	1.00129
$n = 200$	1.00300	1.00147

Table 1: Observed reduction quality for different lattice parameters

Observed root Hermite factor. We report in Table 1 the quality we get with reduction parameters $\delta = 0.99$ and $\eta = 0.71$. As a reminder, the best proved γ for LLL is $\gamma = 1.0754$ and Gama and Nguyen observed $\gamma = 1.0219$ for random lattices of similar dimensions [GN08b]. In our case, we observe that LLL reduces lattices to a very good quality, even with weak reduction parameters δ and η.

5.2.2 Enumeration Behavior

For the enumeration step, we use the technique of pruned enumeration from [LN13]. Its only setting is the pruning function $(R_1^2, \ldots, R_{2n}^2)$. As we know precisely the expected distance between $\mathbf{w}$ and the lattice point $\mathbf{v_0}$, $\mathbf{w} - \mathbf{v_0} = \binom{\sigma \mathbf{s}}{\mathbf{e}}$, we can set the bound R_{2n} to the expected norm of $\binom{\sigma \mathbf{s}}{\mathbf{e}}$. For a Gaussian error distribution we have with high probability $||\mathbf{e}||^2 \leq n \times (3\sigma^2)$, so $R_{2n}^2 = n\sigma^2 + n(3\sigma)^2 = 10n\sigma^2$.

In their paper, Gama et al. [GNR10] introduce three pruning functions: linear, step and piecewise linear, whose they study the resulting time complexity and success probability. They also mention another pruning function obtained by numerical optimization. In our case, we start with a linear pruning function defined by $R_k^2 = (k/2n)R_{2n}^2$.

One of the surprising finding of our experiments is that the enumeration terminates with the first candidate, in nearly every cases. Enumeration for BDD begins with getting a vector *quite close* to the target, and then, enumerates around, towards the one which minimizes the distance. This first vector is usually computed with Babai's Nearest Plane algorithm [Bab86]. It so happens that this solution falls below our pruning bound for most of our test cases, and leads to a successful key recovery. Both our implementation of BDD and `fplll`'s `ClosestVector` routine show this result. The latter has a slight advantage in that it handles floating-point computation more carefully than we do. This prevented us from success on a few cases, where `fplll` worked. It is the only difference we noticed between the implementations.

A qualitative explanation for this unexpected costless enumeration is that the error norm (depending on σ) is very small compared to the Gram-Schmidt vectors, whose norms highly depend on q. Consequently, the simple rounding from Babai algorithm seems enough to get directly to the closest vector, component by component. This result stands also for other FHE schemes, in all which we have the same properties for σ and q.

5.3 Attacking Higher Dimensions

With these first conclusions we decided to go for a greater range of values for n and q. In terms of possible circuit depth evaluation, this range covers up to $L = 13$. We set the attack to an LLL reduction with $\delta = 0.99$ and $\eta \in \{0.51, 0.61, 0.71\}$ and an enumeration limited to Nearest Plane, and launched it for three weeks.

With n up to 250 we see in Figure 2 that with $\eta = 0.71$ the attack takes fairly less time than 0.51, roughly 5 times less and still finishes successfully. This motivated us to keep only the version $\eta = 0.71$ and continue to higher dimensions.

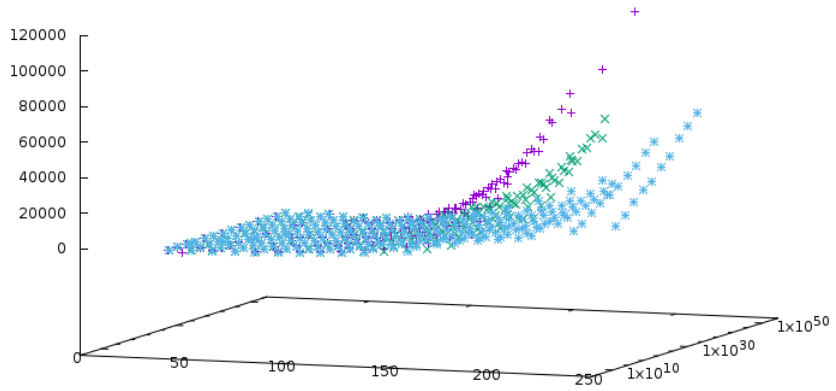

Figure 2: Execution time in seconds in terms of n and q for different η

In the end we were able to successfully break an [FV12] key with $n = 320$ and $\log q = 68$ in little less than 29 hours, see Figure 3. Our result compares favorably with the work of Laine and Lauter [LL15], who were able to recover a key in dimension 350 in 3.5 days, yet with a less generic attack working for q very large and σ very small.

Such parameters are not considered secure, even prior to our work. One would take much greater n for instance. Yet they serve as good examples to understand the performance of our special-purpose attack. When composed with an LLL reduction and an enumeration with Nearest Plane, our proposal requires only a polynomial number of operations (in the parameters of the lattice) to complete. Moreover, the estimator from [APS15], which takes into account only generic attacks, predicts one month of computation to break such key.

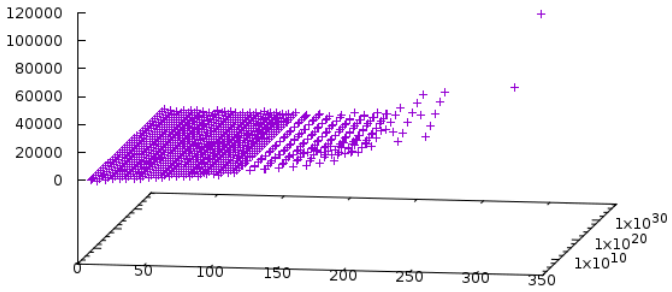

Figure 3: Execution time in seconds in terms of n and q for different $\eta = 0.71$

6 Conclusion

In this work, we aimed at assessing the practical security of FHE schemes based on Ring-LWE. To us, this is a topic that needs more focus, since the requirements of correctness in FHE lead to very special shape of Ring-LWE parameters. After reviewing state-of-the-art attacks, we presented a new special-purpose attack for the case at hands. Our experiments show that such attack has unexpected performance: with only a polynomial number of steps, it successfully breaks keys with parameters beyond toy sizes.

Our main results, on lattice reduction and enumeration in FHE cases, confirm our opinion that attacker may have unexpected advantage in special situations. Our result does not contradict the security reductions of Ring-LWE, but when picking practical parameters, it is really important to consider such results. The discussion about sizing parameters to guarantee a security level objective is far from being closed. We hope we raise interest for this kind of work which is of great importance to move forward with FHE implementation.

References

[ACPS] M. Albrecht, D. Cadé, X. Pujol, and D. Stehlé. fplll-4.0, a floating-point LLL implementation. Available at `http://perso.ens-lyon.fr/damien.stehle`.

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In *Automata, Languages and Programming*, pages 403–415. Springer, 2011.

[AMGH10] Carlos Aguilar-Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with d-operand multiplications. In *Advances in Cryptology–CRYPTO 2010*, pages 138–154. Springer, 2010.

[APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of Learning with Errors. Cryptology ePrint Archive, Report 2015/046, 2015.

[Bab86] László Babai. On lovász'lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[BG14] Shi Bai and Steven D Galbraith. Lattice decoding attacks on binary LWE. In *Information Security and Privacy*, pages 322–337. Springer, 2014.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. ACM, 2012.

[BKW03] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[BLLN13] Joppe W Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *Cryptography and Coding*, pages 45–64. Springer, 2013.

[CIV16] Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Provably weak instances of Ring-LWE revisited. *EUROCRYPT '16, Lecture Notes in Computer Science*, 2016.

[CLS15] Hao Chen, Kristin Lauter, and Katherine E. Stange. Attacks on Search RLWE. Cryptology ePrint Archive, Report 2015/971, 2015. `http://eprint.iacr.org/`.

[CN11] Yuanmi Chen and Phong Q Nguyen. BKZ 2.0: Better lattice security estimates. In *Advances in Cryptology–ASIACRYPT 2011*, pages 1–20. Springer, 2011.

[DGBL+15] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Manual for using homomorphic encryption for bioinformatics. Technical Report MSR-TR-2015-87, November 2015.

[DS15] Yarkın Doröz and Berk Sunar. Flattening NTRU for Evaluation Key Free Homomorphic Encryption. 2015. `http://eprint.iacr.org/`.

[EHL14] Kirsten Eisenträger, Sean Hallgren, and Kristin Lauter. Weak Instances of PLWE. In *Selected Areas in Cryptography–SAC 2014*, pages 183–194. Springer, 2014.

[ELOS15] Yara Elias, Kristin E Lauter, Ekin Ozman, and Katherine E Stange. Provably weak instances of ring-lwe. In *Advances in Cryptology–CRYPTO 2015*, pages 63–92. Springer, 2015.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 2012:144, 2012.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

[GHGN06] Nicolas Gama, Nick Howgrave-Graham, and Phong Q Nguyen. Symplectic lattice reduction and NTRU. In *Advances in Cryptology-EUROCRYPT 2006*, pages 233–253. Springer, 2006.

[GJS15] Qian Guo, Thomas Johansson, and Paul Stankovski. Coded-BKW: Solving LWE using lattice codes. In *Advances in Cryptology–CRYPTO 2015*, pages 23–42. Springer, 2015.

[GN08a] Nicolas Gama and Phong Q Nguyen. Finding short lattice vectors within mordell's inequality. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 207–216. ACM, 2008.

[GN08b] Nicolas Gama and Phong Q Nguyen. Predicting lattice reduction. In *Advances in Cryptology–EUROCRYPT 2008*, pages 31–51. Springer, 2008.

[GNR10] Nicolas Gama, Phong Q Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In *Advances in Cryptology–EUROCRYPT 2010*, pages 257–278. Springer, 2010.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer, 2013.

[Hal] Shai Halevi. Helib. Available at `https://github.com/shaih/HElib`.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Algorithms for the shortest and closest lattice vector problems. In *Coding and Cryptology*, pages 159–190. Springer, 2011.

[Kan87] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of operations research*, 12(3):415–440, 1987.

[KGV15] A. Khedr, G. Gulak, and V. Vaikuntanathan. SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers. *IEEE Transactions on Computers*, PP(99), 2015.

[Lep14] Tancrède Lepoint. A proof-of-concept implementation of the homomorphic evaluation of SIMON using FV and YASHE leveled homomorphic cryptosystems, 2014. [Online; accessed 18-August-2015].

[LL15] Kim Laine and Kristin Lauter. Key Recovery for LWE in Polynomial Time. Cryptology ePrint Archive, Report 2015/176, 2015.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[LN13] Mingjie Liu and Phong Q Nguyen. Solving BDD by enumeration: An update. In *Topics in Cryptology–CT-RSA 2013*, pages 293–309. Springer, 2013.

[LN14] Tancrède Lepoint and Michael Naehrig. A comparison of the homomorphic encryption schemes fv and yashe. In *Progress in Cryptology–AFRICACRYPT 2014*, pages 318–335. Springer, 2014.

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Topics in Cryptology–CT-RSA 2011*, pages 319–339. Springer, 2011.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Advances in Cryptology–EUROCRYPT 2010*, pages 1–23, 2010.

[MW15] Daniele Micciancio and Michael Walter. Practical, predictable lattice basis reduction. Cryptology ePrint Archive, Report 2015/1123, 2015. `http://eprint.iacr.org/`.

[NS05] Phong Q Nguyen and Damien Stehlé. Floating-point LLL revisited. In *Advances in cryptology–Eurocrypt 2005*, pages 215–233. Springer, 2005.

[Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. `http://eprint.iacr.org/`.

[Pei16] Chris Peikert. How (Not) to Instantiate Ring-LWE. Cryptology ePrint Archive, Report 2016/351, 2016.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 84–93. ACM, 2005.

[SE94] Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66(1-3):181–199, 1994.

[Sho15] Victor Shoup. NTL – A Library for doing Number Theory, 2015. [Online; accessed 18-August-2015].

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in cryptology–EUROCRYPT 2010*, pages 24–43. Springer, 2010.

[vdPS13] Joop van de Pol and Nigel P Smart. Estimating key sizes for high dimensional lattice-based systems. In *Cryptography and Coding*, pages 290–303. Springer, 2013.