Automatize parameter tuning in Ring-Learning-With-Errors-based leveled homomorphic cryptosystem implementations

Vincent Herbert
vincent.herbert@protonmail.com

**

Abstract. Lattice-based cryptography offers quantum-resistant cryptosystems but there is not yet official recommendations to choose parameters with standard security levels. Some of these cryptosystems permit secure computations and aim at a wider audience than cryptographic community. We focus on one of them, a leveled homomorphic cryptosystem (LHE): Brakersi/Fan-Vercauteren's (BFV) one. The family of LHE cryptosystems needs to be well-instantiated not only to protect input and output ciphertexts and to perform efficiently computations, but also, for them, parametrization constrains the quantity of homomorphic computations that can be performed with guarantee of correctness. It demands to choose parameters accordingly. In addition, each implementation brings external constraints to optimize performance. All of this makes it tedious for the non-expert user to choose parameters. To solve this, we have developed CinguParam to help user to instantiate implementations of BFV in different libraries: Cingulata, FV-NFLlib and Microsoft SEAL (release 3. 3). CinguParam permits to generate an up-to-date database of parameter sets in function of computation budget, security parameters and implementation choices. This tool includes a notion of budget to ensure correct homomorphic computations and the one of BKZ reduction cost model to grasp the gap from concrete security, nowadays. It makes use of the LWE-Estimator to obtain up-to-date security estimations. CinguParam permits to select automatically a suitable parameter set with Cingulata and it can be used to generate code snippets to set parameters with FV-NFLlib and Microsoft SEAL (release 3. 3).

1 Introduction

Homomorphic encryption offers services to manipulate encrypted data without having access to their associated clear form. As data are encrypted from the beginning to the end of the process, it permits for instance, to share private data with powerful servers in order to delegate costly computations and to get valuable results. But choosing cryptographic parameters is a challenging task in homomorphic cryptography [1] based on the LWE (Learning with Errors) problem [2]. If parameters are inapropriately chosen, not only the attackers could access the data or the key, but also the result of homomorphic computations could be wrong.

In this document, we focus on parameters for the BFV (Brakerski/Fan-Vercauteren) [3,4], *Leveled Homomorphic Encryption* (LHE) scheme, based on the Ring-LWE problem [5], which has multiple open-source implementations [6,7,8,9] in C++.

We have developed CinguParam [10], a tool to ease the choice of parameters in such implementations. We propose different parameter settings rather than fixing one parameter set, in order to propose to the user different choices so that he could get the best set according to his precise needs. It aims to generate correct and secure parameters sets to perform homomorphic computations. It contains a database of parameter sets (see 4.1) which can be easily extended, if needed. In terms of security, LWE attacks are currently the best ones [11, p.8] against Ring-LWE-based cryptosystem. CinguParam makes use of the LWE-Estimator [12] which estimates an up-to-date security of LWE instances. We point that today, nobody has definitive concrete security parameter for lattice-based cryptosystems. Our tool helps to follow research progress. Indeed, there is still some way to go for that:

- Security reduction between lattice problems and LWE problem is not tight for lattice with dimension $n < 187149$ (less than 18 bits) [13].
- The best practical attacks (primal and dual ones) against Ring-LWE make use of BKZ-type algorithms employing enumeration or sieving algorithms. Their cost is unknown, there exist many BKZ cost models [14].
- For most of the models, security estimation is a lower bound under some hypotheses [14]. These are pessimistic views from the defender side.
- There are several recent progresses to better understand and estimate security against LWE attacks, for instance [13,15,16].

** This work was been done while the author was a member of *CEA LIST, Paris-Saclay, France.*

Our tool CinguParam permits to compare the impact of parameter choices for homomorphic computations and to measure the uncertainty on concrete security level, at this stage, in the cryptographic community.

In Section 2, we establish notation for BFV scheme. In Section 3, we provide algorithms used to generate suitable parameters for BFV, and introduce a vocabulary with the willingness to clarify process with a high-level perspective. CinguParam could be extended to determine parameters for other schemes implementations. In Section 4, we present and analyze our experimental results in two parts: we consider CinguParam in standalone mode first, then we combine it with existing tools: Cingulata, FV-NFLlib, Microsoft SEAL (release 3. 3).

2 Scheme parameters

Let us begin to categorize the different parameters of BFV scheme. We employ them in algorithms to generate secure parameters (see Section 3) that can be employed to process correctly a predetermined quantity of homomorphic computations.

BFV is a scheme based on a variant of LWE where data (plaintexts and ciphertexts) are in a polynomial ring. Let the defining polynomial f be a 2-power cyclotomic polynomial.

- ciphertext coefficient count n
- ciphertext modulus q
- Gaussian (noise/error) width σ
- private key distribution χ over $\mathbb{Z}[X]/(f)$

We retain two approaches to determine Gaussian noise width σ. Note that the first one depends on n and is hardness-reduction compliant [17,18]:

$$\sigma = \begin{cases} 2\sqrt{n} \text{ asymptotic approach} \\ \frac{8}{\sqrt{2\pi}} \text{ performance approach [19,20]} \end{cases}$$

Default choice of σ depends on cryptosystem implementation choices, it is indicated in Table 1.

2.1 Correctness parameters

Informally, each circuit[1] to be evaluated by the homomorphic encryption scheme is associated to a circuit cost depending on the cost and on the order of evaluated operations. Circuit budget defines the family of circuits that can be operated over fresh ciphertexts (the ones which are obtained directly from encryption, *i.e.*, without any previous homomorphic operations), the ones whose cost is less than the budget. There exist several approaches to define and estimate operation costs (see Section 3). They depend on:

- the homomorphic scheme
- the circuit budget
- the multiplicative depth L
- the plaintext modulus t

Relinearization Method 1

- integer base ω for gadget decomposition [21], also called decomposition bit count

Relinearization Method 2

- relinearization integer k
- B_k bound on relinearization error distribution
- ciphertext relinearization modulus p

[1] Cingulata take into account Boolean circuits whereas FV-NFLlib and Microsoft SEAL (release 3. 3) perform computations associated with arithmetic circuits.

2.2 Security parameters

There is not yet satisfying solution to choose tightly parameters with a given security level for LWE-based homomorphic cryptosystems. This drives us to decline several notions around security levels, in order to highlight current situation.

- desired security level λ_0
- BKZ lattice reduction cost model
- estimated security level $\lambda(cost\_model)$
- approximated security level $\lambda_1(cost\_model)$ (see Table 3
- tolerance between desired and estimated security levels. Indeed, there can be important gaps between them. Implementations constraints restrain flexibility on parameter choices (*e.g.*, n is a power of two). Other constraints are indicated in Table 1.
- compatibility with Regev quantum security reduction proof [17] which impacts on Gaussian noise width σ
- number of LWE samples (n,q,$relin\_param$)
- success probability ϵ in distinguishing attack on LWE decision problem [22]

3 How to generate up-to-date, correct and secure parameters?

To derive parameters setting for LHE cryptosystems [23] is not a trivial question:

- Security based on variants of LWE is a burning issue. It asks more than ever to keep informed about cryptanalyses results.
- Correct decryption is an additional constraint with LHE cryptosystems compared to traditional cryptosystems. The quantity and the nature of operations that will be operated over the fresh ciphertexts must be known before parametrization. This is a strong constraint.
- Performance is an open problem, as we want to ensure concrete secure computations (*e.g.*, homomorphic computations) in everyday life (see Table 2).

CinguParam makes use of the LWE-Estimator [12] to estimate security against LWE attacks. The LWE-Estimator is also used to generate (n, q) parameters in a document provided by the consortium HomomorphicEncryption.org [19]. This document offer recommendations which consist in tight LWE attack cost estimations based on BKZ Sieve and Q-Core Sieve cost models (Table 4). These recommendations are independant with the scheme used and with implementation choices (Table 2). In particular, it does not take into account noise growth which depends on the chosen scheme and thus do not consider correctness as required for LHE cryptosystems. This explains why for same n, their parameters offer better security levels with smaller q, what may seem counterintuitive [19, pp. 26-29].

Correct decryption depends on (noise/errors) added deliberately during encryption and amplified during each homomorphic operations to the ciphertexts in order to preserve confidentiality. Too much noise leads to decryption failures. Noise is a central notion in LWE-based cryptosystems. Ciphertext noise can be defined pessimistically (from the defender point of view) as the infinite norm of a polynomial which takes part in decryption step. It has different definitions[24, p.11]. Its computation requires the secret key [25]. For practical usage, with a private circuit, we use conservative upper bounds on the circuit noise. The choice of noise definition does not tighten significantly the bound to determine the quantity of secure computations we can operate correctly. In the current version of CinguParam, we employ the *inherent noise* as in the original paper [4].

We define the following terminology:

- $max\_encryption\_noise$ for an upper bound on the noise of a fresh ciphertext,
- $max\_decryption\_noise$ for an upper bound on the noise allowing a correct decryption,
- $max\_circuit\_noise$ for an an upper bound on the noise after evaluating a homomorphic circuit.
- max_relin_noise for an an upper bound on the noise after relinearizing a ciphertext.

The ratio

$$\frac{max\_decryption\_noise(scheme)}{max\_encryption\_noise(scheme)}$$

indicates how much noise, used to secure communications, can increase during homomorphic computations without degrading decryption correctness. It depends on the used scheme.

Algorithm 1: GenerateParam($politic, L, cost\_model, \lambda_0, t, gen\_method$)

$first\_pass = True$
$(n, q) = (n_0, q_0)$
$\sigma, \chi, \epsilon, relin\_method \leftarrow politic$
if $gen\_method = MinCorrectModulus$ **then**
  while *first_pass or estimated security is inadequate* **do**
    $first\_pass = False$
    $q = MinCorrectModulus(n, \sigma, \chi, \epsilon, L, t, relin\_param)$
    $noise\_rate = \frac{\sigma}{q}$
    $\lambda =$ LWE-Estimator$(n, q, \chi, noise_rate, nr_samples,$
     $cost\_model)$
    $n = n \times 2$
  $n = n/2$
else if $gen\_method = MinSecureDegree$ **then**
  while *first_pass or* $max\_circuit\_noise \geq max\_decryption\_noise$ **do**
    $first\_pass = False$
    $(n, \frac{\sigma}{q}, \lambda) = MinSecureDegree(q, \lambda, \chi, cost\_model, relin\_param,$
     $security\_reduction)$
    $max\_encryption\_noise(scheme)$
    $max\_circuit\_noise(scheme, circuit)$
    $max\_decryption\_noise(scheme)$
    $q = q \times scale$
  $q = q/scale$

Algorithm 2: MinSecureDegree

$n = n_0$
$first\_pass = True$
while *first_pass or estimated security is inadequate* **do**
  $first\_pass = False$
  $noise\_rate = \frac{\sigma}{q}$
  $\lambda =$ LWE-Estimator$(n, q, \chi, noise_rate, nr_samples,$
   $cost\_model)$
  $n = n \times 2$
$n = n/2$
return $(n, \frac{\sigma}{q}, \lambda)$

Algorithm 3: MinCorrectModulus

$q = q_0$
$first\_pass = True$
while *first_pass or* $max\_circuit\_noise \geq max\_decryption\_noise$ **do**
  $first\_pass = False$
  $max\_encryption\_noise(scheme)$
  $max\_circuit\_noise(scheme, circuit)$
  $max\_decryption\_noise(scheme)$
  $q = q \times scale$
    // *scale* depends on implementation constraints and modify generation time, database size
$q = q/scale$
return q

The BFV case

Maximal noise in a fresh ciphertext and maximal noise of a (homomorphic) ciphertext have been computed in [26] for some ring-LWE-based LHE schemes. In particular for the BFV scheme [3,4], noise growth has been well described in [27]. We only recall its bound (Equation 1) to compute maximal circuit noise after evaluating a circuit of given multiplicative depth L. In practice, two difficulties with BFV scheme occur:

- Multiplicative depth does not take into account noise growth caused by additions. How to take it into account to avoid incorrect decryption? This could be done by evaluating the variance of the noise at the end of the circuit rather than the infinite norm. This would be more optimistic, from the defender point of view.
- In most of the cases, multiplicative depth is sufficient to ensure correct decryption. But still, how to determine multiplicative depth? This question is solved with Cingulata compiler toolchain and runtime environnement [8]. It is employed with CinguParam, see Section 4.2 and serves to automatize parameter choice. This implementation focuses on binary ciphertexts manipulated with Boolean circuits and permits to interact easily with CinguParam. It is not done in implementations using arithmetic circuits, in our knowledge.

In the BFV scheme, the simplest way to determine the noise budget is derived from the multiplicative depth L. Let us recall, L is the maximal number of multiplications between any input and any output of the circuit we operate to evaluate secure computations. In the case of homomorphic encryption, inputs and outputs are ciphertexts.

In a more general setting, noise growth does not necessarily depend on the multiplicative depth. For instance, with the TFHE scheme [28], the focus is directed on the number of gates.

$$max\_encryption\_noise(\text{BFV}) = B_{error}(\epsilon,\sigma)(1+2\delta B\_key) \tag{1}$$

where δ is the polynomial multiplication expansion factor, *i.e.*, $\max\{\frac{\|a\times b\|_\infty}{\|a\|_\infty\|b\|_\infty} : a,b \in \mathbb{Z}[X]/(X^n+1)\}$, $B_{error}(\epsilon,\sigma) = \sqrt{2}\text{erfc}^{-1}(\epsilon)\sigma$ and B_{key} are respectively an upper bound on the error distribution (resp. the private key distribution). The function erfc is the complementary error function and the quantity ϵ denotes the distinguishing attack advantage on LWE-decision problem, *i.e.*, the statictical distance within noise distribution and Gaussian distribution. For short, noise distribution is bounded by $B_{error}(\epsilon,\sigma)$ with probability $1-\epsilon$.

$$max\_circuit\_noise(\text{BFV},L) = X^L \times max\_encryption\_noise + L \times X^{L-1}(Y + max\_relin\_noise)$$

with

$$X = \delta t(4+\delta B_{key}),$$
$$Y = \delta^2 B_{key}(B_{key}+t^2)$$

and

$$max\_relin\_noise(\text{BFV}) = \begin{cases} \delta \times \left\lceil \frac{\log(q)}{\log(\omega)} \right\rceil \times \omega \times B_{error}(\epsilon,\sigma) & \text{if } relin\_param = 1 \\ \frac{q \times B_k \times \delta}{p} + \frac{\delta \times B_{key}+1}{2} & \text{if } relin\_param = 2 \end{cases}$$

A ciphertext is well decrypted if the circuit noise is smaller than the decryption noise. Circuit budget is used to generate secure and correct parameters with CinguParam. To ensure correct decryption, it has to be greater than decryption cost.

We can extend the vocabulary, similarly to [25], to express intuitively noise constraints:

- *min_decryption_cost*, the minimal cost to ensure correct decryption,
- *min_circuit_budget*, the minimal cost of the circuit evaluation,

$$\begin{aligned} max\_circuit\_noise(\text{BFV},L) &= w \\ min\_circuit\_budget(\text{BFV},L) &= -\log_2 2w \\ max\_decryption\_noise(\text{BFV}) &= \frac{\left\lfloor \frac{q}{t} \right\rfloor - (q \mod t)}{2} \\ min\_decryption\_cost(\text{BFV}) &= -\log_2(\left\lfloor \frac{q}{t} \right\rfloor - (q \mod t)) \end{aligned}$$

Parameters must fulfill these equivalent conditions to ensure a correct result after decryption:

$$max\_circuit\_noise \leq max\_decryption\_noise \tag{2}$$
$$min\_circuit\_budget \geq min\_decryption\_cost \tag{3}$$

Implementation politic	Cingulata _BFV	SEAL_BFV[a]	FV-NFLlib
ciphertext modulus q	power of 2	upper bounded[b] product of distinct primes each prime congruent to 1 modulo $2n$ (NTT multiplications)[d]	product of particular primes[c] of size of the type used to store the polynomial (*i.e.*, 16,32 or 64 bits) minus two
plaintext modulus t	2	≥ 2 (integer encoder) a prime congruent to 1 modulo $2n$ (batch encoder)	≥ 2
default Gaussian noise width σ	$2\sqrt{n}$	3.2	3.2
standard deviation $\sigma\sqrt{2\pi}$	$2\sqrt{2\pi n}$	8	8
statistical distance ϵ within noise distribution and Gaussian distribution	2^{-64}	2^{-28}	2^{-128} (see [29])
noise max width $B_{error}(\epsilon, \sigma)$	10σ (as recommended in [4])	6σ	14σ
private key distribution χ	uniform distribution in $\{0,1\}^n$ with Hamming weight 63	uniform distribution in $\{-1,0,1\}^n$	noise distribution
compatible with Regev's security reduction	yes	no	no
relinearization method [4]	2	1	1

Table 1. Implementation politics are parameter constraints in different implementations of BFV scheme. The parameter n is the ciphertext polynomial degree, a power of two, for performance reason.

[a] Up to release 3.2 of this library, ciphertext modulus bitsize is restricted to multiple of 10. This impeaches tight parametrization in a similar manner as FV-NFLlib, in Table 2.
[b] See the file native/src/seal/util/hestdparms.h in Microsoft SEAL (release 3. 3) for upper bounds on size depending on modulus degree.
[c] See the file include/nfl/params.hpp for exact form in FV-NFLlib library
[d] The congruence is required for batching with CRT.

4 Analysis and presentation of experimental results

4.1 CinguParam in standalone mode

CinguParam permits to:

- generate and store a database of parameter sets for different
  - desired levels of security
  - multiplicative depths
  - implementation politic
- estimate with hindsight the security of parameter sets against LWE attacks

Input parameters Database contains files whose names indicate main input parameters in the following order:

- The circuit multiplicative depth, an integer $0 \leq L \leq 20$
- The BKZ lattice reduction cost model (see Table 4)
- The approximated security level λ_1 (80,128,192,256)
- The plaintext modulus t (see Table 1)

The database can be easily extended to customized parameters. It is important to note that there is no parameter set for all $(L, cost\_model, \lambda_0, t)$. This is due to the gap between desired security level λ_0 and estimated security level λ which can be greater than 64 bits. For this reason, at the end of database generation, each file is renamed by approximating estimated security level λ (see 3 with standard security levels. We only consider parameter sets in the range $[\![80, 256]\!]$. For this reason, the database can contain less parameter sets than expected and we employ two methods: Algorithm 2 and 3 to select parameters inside Algorithm 1. Our experimental results indicate Algorithm 3 generate more performant parameters in terms of ciphertext size, *i.e.*, $n \log_2(q)$. Note, it is not possible to employ Algorithm 3 for FV-NFLlib and old versions of SEAL (up to release 3.2) since we can not ensure output modulus would have a bitsize with particular properties, as imposed by these implementations (see Table 1 and 2). Parameter set contain a decomposition of ciphertext modulus bitsize suited for Microsoft SEAL (release 3. 3) to ease its use in the library.

Implementation politic	Cingulata_BFV		SEAL_BFV		FV-NFLlib	
Plaintext modulus	2		65537		65537	
Generation method	MinCorrectModulus		MinCorrectModulus		MinSecureDegree	
L_cost_model_λ_1	$\log_2(q)$	n	$\log_2(q)$	n	$\log_2(q)$	n
0_q_core_sieve_80 0_core_sieve_80 0_bkz_sieve_128 0_bkz_enum_192	54	2048	54	2048		
0_bkz_sieve_80 0_bkz_enum_128					62	2048 2048
0_paranoid_sieve_128 0_q_core_sieve_192	54	4096	54	4096	62	4096
0_core_sieve_192 0_bkz_sieve_192						
5_bkz_sieve_80 5_bkz_enum_80	158	4096	254	8192	310	8192
5_bkz_enum_128			254	8192		
5_paranoid_sieve_80 5_core_sieve_128 5_bkz_sieve_128	166	8192			310	16384
5_q_core_sieve_128 5_bkz_sieve_192 5_paranoid_sieve_128 5_core_sieve_192			262	16384	310	16384
5_paranoid_sieve_192	174	16384				
10_bkz_sieve_80 10_bkz_enum_128	302	8192	478	16384	558	16384
10_paranoid_sieve_80 10_bkz_sieve_128 10_core_sieve_128 10_q_core_sieve_128	326	16384			558	32768
10_paranoid_sieve_128 10_bkz_sieve_192 10_core_sieve_192 10_q_core_sieve_192			502	32768		
10_paranoid_sieve_192	342	32768				
15_core_sieve_80 15_bkz_enum_80	470	16384	478 702	16384	868	32768
15_bkz_enum_192 15_bkz_sieve_80	470	16384			868	32768
15_bkz_sieve_128 15_q_core_sieve_80 15_paranoid_sieve_80 15_core_sieve_128			734	32768	868	32768
15_paranoid_sieve_128 15_q_core_sieve_192 15_bkz_sieve_192 15_core_sieve_192	502	32768			868	65536
15_paranoid_sieve_192			766	65536		
20_bkz_enum_80 20_bkz_sieve_80	614	16384			1116	32768
20_bkz_enum_128 20_core_sieve_80			966	32768		
20_paranoid_sieve_80 20_bkz_sieve_128 20_core_sieve_128 20_q_core_sieve_128	654	32768			1116	65536
20_paranoid_sieve_128 20_bkz_sieve_192 20_core_sieve_192 20_q_core_sieve_192			1006	65536		
20_paranoid_sieve_192	694	65536				

Table 2. Some parameters from CinguParam database for different politics and plaintext moduli. Let us recall, ciphertexts are represented by polynomials of degree n with coefficients of bitsize $\log_2(q)$, in BFV cryptosystem. Computation budget (multiplicative depth, in this case) has an important impact on memory/time cost. Different files have same parameters, the approximated security level λ_1 reflects the degree of pessimism/optimism of each BKZ cost model and the difficulty to determine security level compared to traditional cryptosystems.

Security and correctness In general, security strength is defined in terms of bit of security (*e.g.*, 80 for legacy standard protection level, 128 for security at least ten-years, 192 for long-term protection). For post-quantum systems, 5 different categories [30, pp.15-19] have been defined by NIST because of uncertainties on security estimations. The first approach is known from a wider audience. For this reason, we make use of it, in CinguParam.

In LWE-based cryptography, one important difficulty is to estimate the attack cost. Lattice reduction algorithm are employed in LWE attacks, it makes use of an SVP oracle (two families: enumeration and sieving one). There

exist many models in the literature[2]. Each model correspond to a bound on the cost of BKZ-family lattice reduction algorithm. There exist important gaps between estimations as shown in [14]. In other words, it is not possible to obtain desired security level with certitude, we only have estimations according to a BKZ cost model.

The database in CinguParam contains parameter sets considering this selection of BKZ lattice reduction cost models and can be extended for any model of your choice. It is stored in xml files. We select five cost models which are commonly used in the literature. They are given in Table 4 from the most optimistic to the most pessimitic, from the defender point of view.

Security is estimated a priori against primal attack [11] via uSVP. A posteriori, we estimate security against dual-lattice attack and small/sparse secret variant [31] as well as lattice decoding attacks [32]. All estimations are operated with LWE-Estimator [12].

In addition, homomorphic cryptosystem is costly in terms in performance. This forces to reduce the parameter range to lessen costs. For instance, the ciphertext polynomial degree is a power of two. This kind of restriction prevents to have a parameter set with a chosen security level $\lambda(cost\_model)$.

To face the situation, we associate a desired security level with an interval of estimated security in Table 3.

Estimated security range	Approximated security level
$[\![80, 128 - tolerance\_gap]\!]$	80
$[\![128 - tolerance\_gap, 192 - tolerance\_gap]\!]$	128
$[\![192 - tolerance\_gap, 256 - tolerance\_gap]\!]$	192
$[\![256 - tolerance\_gap, 312]\!]$	256

Table 3. There is a gap between concrete security and desired security in lattice based cryptography. For the time being, we only associate an approximated security level with an estimated security range. Estimations are bounds on attack cost given a BKZ lattice reduction cost model. Arbitrarily, we choose $tolerance\_gap = 8$ during database generation. This can be easily modified if desired.

BKZ cost models	Short description	Reference
BKZ Enum	Enumeration algorithm prevail for dimension < 70, sieving one for greater dimensions.	[33, Table 4]
BKZ Sieve	The number of calls to an SVP oracle is polynomial and estimated heuristically.	[34, Figure 3]
Core Sieve (mode classical)	Core models evaluate one call to an SVP oracle	[11]
Q-Core Sieve (mode quantum)	The attacker benefit from Grover's search algorithm	
Paranoid Sieve (mode paranoid)	Best plausible attack cost	

Table 4. Default BKZ lattice reduction cost models in CinguParam database. We can easily extend it for better models to come. The costs are exponential in the blocksize β for each model, they are defined in `https://bitbucket.org/malb/lwe-estimator/raw/HEAD/estimator.py`.

4.2 CinguParam combined with cryptosystems implementations

We focus here on implementations with recent activities. Let us present them briefly. On one hand, Microsoft SEAL is a library permitting to use two homomorphic encryption schemes: full-RNS BFV [35] and CKKS [36]. The release 3.3 (june 2019) increases flexibility on parameter choices by removing implementation constraints on bitsize of ciphertext moduli q. On the other hand, Cingulata is a compiler toolchain and runtime environment for running programs over encrypted data with two homomorphic encryption schemes since june 2019: BFV and TFHE [28]. With Microsoft SEAL (release 3. 3) and potentially any implementation of BFV, we can generate automatically snippets to initialize cryptographic parameters by selecting any parameter set in CinguParam. We demonstrate with Cingulata, a more practical usage of CinguParam. Parameters are selected automatically using CinguParam as a submodule for any program written for BFV implementation in Cingulata. During compilation of the program describing the homomorphic computations, Cingulata evaluates the circuit multiplicative depth associated to each test written in a C++ program. Afterwards, CinguParam makes use of it to select automatically parameter set in

[2] https://estimate-all-the-lwe-ntru-schemes.github.io/docs/

the database. Default security parameters are $\lambda_1 = 128$, cost_mode=BKZ Sieve. Plaintext modulus is $t = 2$ in Cingulata. If no parameter set is found given input parameters $(L, cost\_model, \lambda_1, t)$, a static choice is made by default to find some approaching parameter set. For instance, a parameter set with multiplicative depth $L + 1$ or $L + 2$ is chosen if no parameter set is found for the quadruplet of input parameters. An interactive mode is also proposed to select manually a parameter set in a filtered database if nothing corresponds to input parameters.

4.3 Future works

There exist several trails to extend our work :

- Tighten noise estimation to improve performance (see Table 2) using noise variance, in the average case (see Section 3). In counterpart, this approach requires additional hypotheses.
- Take into account hybrid relinearization technique introduced in [37].
- Extend CinguParam with other LWE-based homomorphic cryptosystems.

References

1. Gentry, C., et al.: Fully homomorphic encryption using ideal lattices. In: Stoc. Volume 9. (2009) 169–178
2. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) **56**(6) (2009) 34
3. Brakerski, Z.: Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP
4. Fan, J., Vercauteren, F.: Somewhat Practical Fully Homomorphic Encryption. IACR Cryptology ePrint Archive **2012** (2012) 144
5. Lyubashevsky, V., Peikert, C., Regev, O.: On Ideal Lattices and Learning with Errors over Rings. In Gilbert, H., ed.: Advances in Cryptology – EUROCRYPT 2010, Berlin, Heidelberg, Springer Berlin Heidelberg (2010) 1–23
6. : FV-NFLlib. `https://github.com/CryptoExperts/FV-NFLlib` (2016) CryptoExperts.
7. Carpov, S., Dubrulle, P., Sirdey, R.: Armadillo: A Compilation Chain for Privacy Preserving Applications. In: Proceedings of the 3rd International Workshop on Security in Cloud Computing. SCC '15, New York, NY, USA, ACM (2015) 13–19
8. : Cingulata. `https://github.com/CEA-LIST/Cingulata` (2019) CEA List, Paris-Saclay, France.
9. : Microsoft SEAL (release 3.3). `https://github.com/Microsoft/SEAL` (2019) Microsoft Research, Redmond, WA.
10. : CinguParam. `https://github.com/CEA-LIST/CinguParam` (2019) CEA List, Paris-Saclay, France.
11. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange — A New Hope. In: 25th {USENIX} Security Symposium ({USENIX} Security 16). (2016) 327–343
12. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3) (2015) 169–203
13. Sarkar, P., Singha, S.: Verifying Solutions to LWE with Implications for Concrete Security. Cryptology ePrint Archive, Report 2019/728 (2019) `https://eprint.iacr.org/2019/728`.
14. Albrecht, M.R., Curtis, B.R., Deo, A., Davidson, A., Player, R., Postlethwaite, E., Virdia, F., Wunderer., T.: Estimate all the LWE, NTRU schemes! `https://estimate-all-the-lwe-ntru-schemes.github.io/docs/`.
15. Bai, S., Miller, S., Wen, W.: A refined analysis of the cost for solving LWE via uSVP . Cryptology ePrint Archive, Report 2019/502 (2019) `https://eprint.iacr.org/2019/502`.
16. Guo, Q., Johansson, T., Mårtensson, E., Wagner, P.S.: On the Asymptotics of Solving the LWE Problem Using Coded-BKW with Sieving. Cryptology ePrint Archive, Report 2019/009 (2019) `https://eprint.iacr.org/2019/009`.
17. Peikert, C.: How (not) to instantiate ring-LWE. In: International Conference on Security and Cryptography for Networks, Springer (2016) 411–430
18. Regev, O.: The Learning With Errors problem. Invited survey in CCC **7** (2010)
19. Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Halevi, S., Hoffstein, J., Laine, K., Lauter, K., Lokam, S., Micciancio, D., Moody, D., Morrison, T., Sahai, A., Vaikuntanathan, V.: Homomorphic Encryption Security Standard. Technical report, HomomorphicEncryption.org, Toronto, Canada (November 2018)
20. Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Hoffstein, J., Lauter, K., Lokam, S., Moody, D., Morrison, T., Sahai, A., Vaikuntanathan, V.: Security of Homomorphic Encryption. Technical report, HomomorphicEncryption.org, Redmond WA, USA (July 2017)
21. Bonnoron, G., Fontaine, C., Gogniat, G., Herbert, V., Lapotre, V., Migliore, V., Roux-Langlois, A.: Somewhat/Fully Homomorphic Encryption: Implementation Progresses and Challenges. In: C2SI 2017 : 2nd International Conference on Codes, Cryptology and Information Security. Volume 10194 - LNCS (Lectures Notes in Computer Science)., Rabat, Morocco, Springer (April 2017) 68 – 82
22. Lindner, R., Peikert, C.: Better Key Sizes (and Attacks) for LWE-Based Encryption. In Kiayias, A., ed.: Topics in Cryptology – CT-RSA 2011, Berlin, Heidelberg, Springer Berlin Heidelberg (2011) 319–339

23. Migliore, V., Bonnoron, G., Fontaine, C.: Practical Parameters for Somewhat Homomorphic Encryption Schemes on Binary Circuits. IEEE Transactions on Computers **67**(11) (2018) 1550–1560
24. Costache, A., Laine, K., Player, R.: Homomorphic noise growth in practice: comparing BGV and FV. Cryptology ePrint Archive, Report 2019/493 (2019) `https://eprint.iacr.org/2019/493`.
25. Laine, K.: Simple encrypted arithmetic library 2.3. 1
26. Bos, J.W., Lauter, K., Loftus, J., Naehrig, M.: Improved security for a ring-based fully homomorphic encryption scheme. In: IMA International Conference on Cryptography and Coding, Springer (2013) 45–64
27. Lepoint, T., Naehrig, M.: A comparison of the homomorphic encryption schemes FV and YASHE. In: International Conference on Cryptology in Africa, Springer (2014) 318–335
28. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast Fully Homomorphic Encryption Library (August 2016) `https://tfhe.github.io/tfhe/`.
29. Dwarakanath, N.C., Galbraith, S.D.: Sampling from discrete Gaussians for lattice-based cryptography on a constrained device. Applicable Algebra in Engineering, Communication and Computing **25**(3) (Jun 2014) 159–180
30. : Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography` (2016) NIST.
31. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. Cryptology ePrint Archive, Report 2017/047 (2017) `https://eprint.iacr.org/2017/047`.
32. Bai, S., Galbraith, S.D.: Lattice Decoding Attacks on Binary LWE. In Susilo, W., Mu, Y., eds.: Information Security and Privacy, Cham, Springer International Publishing (2014) 322–337
33. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better Lattice Security Estimates. In Lee, D.H., Wang, X., eds.: Advances in Cryptology – ASIACRYPT 2011, Berlin, Heidelberg, Springer Berlin Heidelberg (2011) 1–20
34. Albrecht, M.R., Ducas, L., Herold, G., Kirshanova, E., Postlethwaite, E.W., Stevens, M.: The General Sieve Kernel and New Records in Lattice Reduction. In: Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II. (2019) 717–746
35. Bajard, J.C., Eynard, J., Hasan, A., Zucca, V.: A Full RNS Variant of FV like Somewhat Homomorphic Encryption Schemes. In: Selected Areas in Cryptography - SAC, St. John's, Newfoundland and Labrador, Canada (August 2016)
36. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: International Conference on the Theory and Application of Cryptology and Information Security, Springer (2017) 409–437
37. Chen, H., Dai, W., Kim, M., Song, Y.: Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference. Cryptology ePrint Archive, Report 2019/524 (2019) `https://eprint.iacr.org/2019/524`.