On a dual/hybrid approach to small secret LWE

Thomas Espitau[1], Antoine Joux[2,3], and Natalia Kharchenko[4]

[1] NTT Corporation, Tokyo, Japan
[2] CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
[3] Sorbonne Université and Université de Paris,
CNRS, IMJ-PRG, F-75006 Paris, France
[4] Sorbonne Université, LIP 6, CNRS UMR 7606, Paris, France

Abstract. In this paper, we investigate the security of the Learning With Error (LWE) problem with small secrets by refining and improving the so-called dual lattice attack. More precisely, we use the dual attack on a projected sublattice, which allows generating instances of the LWE problem with a slightly bigger noise that correspond to a fraction of the secret key. Then, we search for the fraction of the secret key by computing the corresponding noise for each candidate using the newly constructed LWE samples. As secrets are small, we can perform the search step very efficiently by exploiting the recursive structure of the search space. This approach offers a trade-off between the cost of lattice reduction and the complexity of the search part which allows to speed up the attack. Besides, we aim at providing a sound and *non-asymptotic* analysis of the techniques to enable its use for practical selection of security parameters. As an application, we revisit the security estimates of some fully homomorphic encryption schemes, including the Fast Fully Homomorphic Encryption scheme over the Torus (TFHE) which is one of the fastest homomorphic encryption schemes based on the (Ring-)LWE problem. We provide an estimate of the complexity of our method for various parameters under three different cost models for lattice reduction and show that the security level of the TFHE scheme should be re-evaluated according to the proposed improvement (for at least 7 bits for the most recent update of the parameters that are used in the implementation).

1 Introduction

The Learning With Errors (LWE) problem was introduced by Regev [38] in 2005. A key advantage of LWE is that it is provably as hard as certain lattice approximation problems in the worst-case [11], which are believed to be hard even on a quantum computer. The LWE problem has been a rich source of cryptographic constructions. As a first construction, Regev proposed an encryption scheme, but the flexibility of this security assumption proved to be extremely appealing to construct feature-rich cryptography [12, 27].

Among these constructions, Fully homomorphic encryption (FHE) is a very interesting primitive, as it allows performing arbitrary operations on encrypted

data without decrypting it. A first FHE scheme relying on the so-called ideal lattices was proposed in a breakthrough work of Gentry [26]. After several tweaks and improvements through the years, the nowadays popular approaches to FHE rely on the LWE problem or its variants (e.g. [10, 18, 22, 23, 28]).

Informally, when given several samples of the form $(a, \langle a, s\rangle + e \mod q)$ where s is secret, $a \in Z_q^n$ is uniform and e is some noise vector, the LWE problem is to recover s.

In its original formulation, the secret vector is sampled uniformly at random from $\mathbb{Z}_q^n$, but more recent LWE-based constructions choose to use distribution with small entropy for the secret key to increase efficiency. For example, some FHE schemes use binary [19,22], ternary [14], or even ternary sparse secrets [29]. Theoretical results are supporting these choices, which show that the LWE remains hard even with small secrets [11]. In practice, however, such distributions can lead to more efficient attacks [7, 17, 39].

The security of a cryptosystem, of course, depends on the complexity of the most efficient known attack against it. In particular, to estimate the security of an LWE-based construction, it is important to know which attack is the best for the parameters used in the construction. It can be a difficult issue; indeed, the survey of existing attacks against LWE given in [6] shows that no known attack would be the best for all sets of LWE parameters.

In this article, we are interested in evaluating the practical security of the LWE problem with such small secrets. As an application, we consider the bit-security of several very competitive FHE proposals, such as the Fast Fully Homomorphic Encryption scheme over the Torus [19, 20, 21], FHEW [22], SEAL [32], and HElib [29]. The security of these constructions relies on the hardness of variants of the LWE problem which can all be encompassed in a scale-invariant version named Torus-LWE. This "learning a character" problem, captures both the celebrated LWE and Ring-LWE problems.

In the case of TFHE, in [20], the authors adapted and used the dual distinguishing lattice attack from [2] to evaluate the security of their scheme. Recently, in [21, Remark 9], the authors propose an updated set of the parameters for their scheme and estimate the security of the new parameters using the LWE estimator from [4]. It turns out that this approach falls into the caveat we described above: the estimator relies on attacks that are not fine-tailored to capture the peculiar choice of distributions. According to the LWE estimator, the best attack against the current TFHE parameters is the unique-SVP attack [7].

1.1 Contributions and overview of the techniques

We present our work in the generic context of the so-called scale-invariant LWE problem or Torus-LWE, which appears to give a more flexible mathematical framework to perform the analysis of the attacks. We aim at extending the use-case of the dual lattice attack which is currently one of the two main techniques used to tackle the LWE problem. Given Torus-LWE samples collected in matrix form $\vec{A}^t\vec{s} + \vec{e} = \vec{b} \mod 1$, the vanilla dual attack consists in finding a vector $\vec{v}$

such that $\vec{v}^t\vec{A}^t = 0 \mod 1$, yielding the equation $\vec{v}^t\vec{e} = \vec{v}^t\vec{b} \mod 1$. Since $\vec{v}^t\vec{e}$ should be small, we can then distinguish it from a random vector in the torus.

A refined analysis of the dual attack. First, we introduce a complete *non-asymptotic* analysis of the standard dual lattice attack[5] on LWE. In particular, we prove *non-asymptotic bounds* on the number of samples and the corresponding bit-complexity of the method, allowing precise instantiations for parameter selection[6]. To do so, we introduce an unbiased estimator of the Levy transform for real random variables that allows us to sharpen the analysis of the attack.

The intuition behind these techniques is the following. The crux of the dual attack is distinguishing the Gaussian distribution modulo 1 from the uniform distribution. Since we are working modulo 1, a natural approach is to tackle the problem by harmonic analysis techniques. Moving to the Fourier space is done by the so-called Levy transform (which is essentially given by $x \mapsto e^{2i\pi x}$). In this space, the Levy transform of the Gaussian distribution mod 1 and the full Gaussian distribution coincides, so we somehow get rid of the action of the modulo. Then we use Berry-Esseen inequality to derive sharper bounds. We hope these techniques may find other interests for the community.

A hybrid enumeration/dual attack. In the second step, we show that applying the dual attack to a projected sublattice and combining it with the search for a fraction of the key can yield a more efficient attack. It can be considered as a dimension reduction technique, trading the enumeration time with the dimension of the lattice attack. More precisely, we obtain a trade-off between the quality of lattice reduction in the dual attack part and the time subsequently spent in the exhaustive search. Additionally, for the lattice reduction algorithms using sieving as an SVP-oracle, we demonstrate that the pool of short vectors obtained by the sieving process can be used to amortize the cost of the reduction part. We also discuss possible improvements based on so-called "combinatorial" techniques, where we perform a random guess of the zero positions of the secret if it is sparse enough.

In a word, our attack starts by applying lattice reduction to a projected sublattice in the same way as it is applied to the whole lattice in the dual attack. This way, we generate LWE instances with bigger noise but in smaller dimension, corresponding to a fraction of the secret key. Then, the freshly obtained instances are used to recover the remaining fraction of the secret key. For each candidate for this missing fraction, we compute the noise vector corresponding to the LWE instances obtained at the previous step. This allows us to perform a majority voting procedure to detect the most likely candidates. For small secrets, this step boils down to computing a product of a matrix of the LWE samples with the matrix composed of all the possible parts of the secret key that we are searching

[5] We point out that this attack is slightly more subtle than the vanilla dual technique, as it encompasses a continuous relaxation of the *lazy modulus switching* technique of [2].

[6] Up to our knowledge previous analyses rely on instantiation of asymptotic inequalities and overlook the practical applicability.

for. We show that this computation can be performed efficiently thanks to the recursive structure of the corresponding search space.

Applications. In the last part, we estimate the complexity of our attack under three different models of lattice reduction and compare the complexity of our attack with the standard dual attack and with the primal unique-SVP attack for a wide range of LWE parameters in the case of small non-sparse secrets. Concerning the comparison with the primal unique-SVP attack, both attacks give quite close results. Our attack is better than uSVP when the dimension and the noise parameter are big, the uSVP attack is better when the dimension is big and the noise parameter is small (see Figure 10). We also provide experiments in small dimensions, supporting the whole analysis.

To evaluate the practicality of our approach, we apply our attack to the security analysis of competitive FHE schemes, namely TFHE [21], FHEW [22], SEAL [32], and HElib [29]. We show that our hybrid dual attack gives improvement compared to the unique-SVP or dual technique of [4] for the latest TFHE's, FHEW's and SEAL's parameters. In case of sparse secrets in the HElib scheme, our attack doesn't provide improvements over the dual attack from [2], but gives comparable results.

The results of our comparison for the TFHE scheme are presented in Table 1.

	parameters (n, α)	dual [2, 4]	this work	uSVP [4]
Old param.	switching key $n = 500$, $\alpha = 2.43 \cdot 10^{-5}$	113	**94**	101
	bootstrapping key $n = 1024$, $\alpha = 3.73 \cdot 10^{-9}$	125	112	116
New param.	switching key $n = 612$, $\alpha = 2^{-15}$	140	**118**	123
	bootstrapping key $n = 1024$, $\alpha = 2^{-26}$	134	120	124
Implem. param.	switching key $n = 630$, $\alpha = 2^{-15}$	144	**121**	127
	bootstrapping key $n = 1024$, $\alpha = 2^{-25}$	140	125	129

Table 1: Security estimates of the parameters of TFHE from [21, Table 3, Table 4] and from the public implementation [40]. n denotes the dimension, α is the parameter of the modular Gaussian distribution. The bold numbers denote the overall security of the scheme for a given set of parameters. The "uSVP" column corresponds to the estimates obtained using the LWE Estimator [4] for the primal uSVP attack. For the lattice reduction algorithm, in all the cases, the sieving BKZ cost model is used.

1.2 Related work

The survey [6] outlines three strategies for attacks against LWE: exhaustive search, BKW algorithm [3,9], and lattice reduction. Lattice attacks against LWE can be separated into three categories depending on the lattice used: distinguishing dual attacks [2], decoding (primal) attacks [34, 35], and solving LWE by reducing it to the unique Shortest Vector Problem (uSVP) [5].

The idea of a hybrid lattice reduction attack was introduced by Howgrave–Graham in [31]. He proposed to combine a meet-in-the-middle attack with lattice reduction to attack NTRUEncrypt. Then, Buchmann et al. adapted Howgrave–Graham's attack to the settings of LWE with binary error [13] and showed that the hybrid attack outperforms existing algorithms for some sets of parameters. This attack uses the decoding (primal) strategy for the lattice reduction part. Following these two works, Wunderer has provided an improved analysis of the hybrid decoding lattice attack and meet-in-the-middle attack and re-estimated security of several LWE and NTRU based cryptosystems in [41]. Also, very recently, a similar combination of primal lattice attack and meet-in-the-middle attack was applied to LWE with ternary and sparse secret [39]. This last reference shows that the hybrid attack can also outperform other attacks in the case of ternary and sparse secrets for parameters typical for FHE schemes.

A combination of the dual lattice attack with guessing for a part of the secret key was considered in [2, Section 5], in the context of *sparse secret keys*. Also, recently, a similar approach was adapted to the case of ternary and sparse keys in [17]. Both of these articles can be seen as *dimension reduction* techniques as they both rely on a guess of the part of the secret to perform the attack in smaller dimension. They gain in this trade-off by exploiting the sparsity of the secret: guessing the position of zero bits will trade positively with the dimension reduction as soon as the secret is sparse enough. However, the main difference of this work compared to [2, 17] is that the secret is *not* required to be sparse, and thus can be considered to be slightly more general. We positively trade-off with the dimension gain by exploiting the recursive structure of the small secret space. However, all these techniques are not incompatible! In Section 4.4, we propose a combination of the guessing technique with our approach, allowing us to leverage at the same time the sparsity *and* the structure of small secrets.

Overall we can consider this work as providing a proper dual analog of enumeration-hybrid technique existing for primal attacks.

Outline. This paper is organized as follows. In Section 2, we provide the necessary background on lattice reduction and probability. In Section 3, we revisit the dual lattice attack and provide a novel and sharper analysis of this method. In Section 4, we describe our hybrid dual lattice attack and discuss its extension to sparse secrets. In Section 5, we compare the complexities of different attacks, revisit the security estimate of TFHE, and several other FHE schemes and provide some experimental evidence supporting our analysis.

2 Background

We use column notation for vectors and denote them using bold lower-case letters (e.g. $\mathbf{x}$). Matrices are denoted using bold upper-case letters (e.g. $\mathbf{A}$). For a vector $\mathbf{x}$, $\mathbf{x}^t$ denotes the transpose of $\mathbf{x}$, i.e., the corresponding row-vector. Base-2 logarithm is denoted as log, natural logarithm is denoted as ln. We denote the set of real numbers modulo 1 as the torus $\mathbb{T}$. For a finite set S, we denote by $|S|$ its cardinality and by $\mathcal{U}(S)$ the discrete uniform distribution on its elements. For any compact set $S \subset \mathbb{R}^n$, the uniform distribution over S is also denoted by $\mathcal{U}(S)$. When S is not specified, $\mathcal{U}$ denotes uniform distribution over $(-0.5; 0.5)$.

2.1 The LWE problem

Abstractly, all operations of the TFHE scheme are defined on the real torus $\mathbb{T}$ and to estimate the security of the scheme it is convenient to consider a scale-invariant version of LWE problem.

Definition 1 (Learning with Errors, [11, Definition 2.11]). *Let $n \geqslant 1$, $\mathbf{s} \in \mathbb{Z}^n$, ξ be a distribution over $\mathbb{R}$ and $\mathcal{S}$ be a distribution over $\mathbb{Z}^n$. We define the $\mathrm{LWE}_{\mathbf{s},\xi}$ distribution as the distribution over $\mathbb{T}^n \times \mathbb{T}$ obtained by sampling $\mathbf{a}$ from $\mathcal{U}(\mathbb{T}^n)$, sampling e from ξ and returning $(\mathbf{a}, \mathbf{a}^t\mathbf{s} + e)$.*

Given access to outputs from this distribution, we can consider the two following problems:

- Decision-LWE. *Distinguish, given arbitrarily many samples, between $\mathcal{U}(\mathbb{T}^n \times \mathbb{T})$ and $\mathrm{LWE}_{\mathbf{s},\xi}$ distribution for a fixed $\mathbf{s}$ sampled from $\mathcal{S}$.*
- Search-LWE. *Given arbitrarily many samples from $\mathrm{LWE}_{\mathbf{s},\xi}$ distribution with fixed $\mathbf{s} \leftarrow \mathcal{S}$, recover the vector $\mathbf{s}$.*

To complete the description of the LWE problem we need to choose the error distribution ξ and the distribution of the secret key $\mathcal{S}$. Given a finite set of integers B, we define $\mathcal{S}$ to be $\mathcal{U}(B^n)$ and ξ to be a centered continuous Gaussian distribution, i.e., we consider the LWE problem with binary secret. This definition captures the *binary* and *ternary* variants of LWE by choosing B to be respectively $\{0, 1\}$ and $\{-1, 0, 1\}$. In [11], it is shown that this variation of LWE with small secrets remains hard. Finally, we use the notation $\mathrm{LWE}_{\mathbf{s},\sigma}$ as a shorthand for $\mathrm{LWE}_{\mathbf{s},\xi}$, when ξ is the Gaussian distribution centered at 0 and with standard deviation σ.

2.2 Lattices

A *lattice* Λ is a discrete subgroup of $\mathbb{R}^d$. As such, a lattice Λ of rank n can be described as a set of all integer linear combinations of $n \leqslant d$ linearly independent vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_d\} \subset \mathbb{R}^d$:

$$\Lambda = \mathcal{L}(\mathbf{B}) := \mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_d,$$

called a basis. Bases are not unique, one lattice basis may be transformed into another one by applying an arbitrary unimodular transformation. The *volume of the lattice* $\mathrm{vol}(\Lambda)$ is equal to the square root of the determinant of the Gram matrix $\mathbf{B}^t\mathbf{B}$: $\mathrm{vol}(\Lambda) = \sqrt{\det(\mathbf{B}^t\mathbf{B})}$. For every lattice Λ we denote the length of its shortest non-zero vector as $\lambda_1(\Lambda)$. Minkowski's theorem states that $\lambda_1(\Lambda) \leqslant \sqrt{\gamma_n} \cdot \mathrm{vol}(\Lambda)^{1/n}$ for any d-dimensional lattice Λ, where $\gamma_d < d$ is d-dimensional Hermite's constant. The problem of finding the shortest non-zero lattice vector is called the *Shortest Vector Problem*(SVP). It is known to be NP-hard under randomized reduction [1].

2.3 Lattice reduction

A lattice reduction algorithm is an algorithm which, given as input some basis of the lattice, finds a basis that consists of relatively short and relatively pairwise-orthogonal vectors. The quality of the basis produced by lattice reduction algorithms is often measured by the Hermite factor $\delta = \dfrac{\|\mathbf{b}_1\|}{\det(\Lambda)^{1/d}}$, where $\mathbf{b}_1$ is the first vector of the output basis. Hermite factors bigger than $\left(\frac{4}{3}\right)^{n/4}$ can be reached in polynomial time using the LLL algorithm [33]. In order to obtain smaller Hermite factors, blockwise lattice reduction algorithms, like BKZ-2.0 [16] or S-DBKZ [37], can be used. The BKZ algorithm takes as input a basis of dimension d and proceeds by solving SVP on lattices of dimension $\beta < d$ using sieving [8] or enumeration [25]. The quality of the output of BKZ depends on the blocksize β. In [30] it is shown that after a polynomial number of calls to SVP oracle, the BKZ algorithm with blocksize β produces a basis $\mathbf{B}$ that achieves the following bound:

$$\|\mathbf{b}_1\| \leqslant 2\gamma_\beta^{\frac{d-1}{2(\beta-1)}+\frac{3}{2}} \cdot \mathrm{vol}(\mathbf{B})^{1/d}.$$

However, up to our knowledge, there is no closed formula that tightly connects the quality and complexity of the BKZ algorithm. In this work, we use experimental models proposed in [3, 4] in order to estimate the running time and quality of the output of lattice reduction. They are based on the following two assumptions on the quality and shape of the output of BKZ. The first assumption states that the BKZ algorithm outputs vectors with balanced coordinates, while the second assumption connects the Hermite factor δ with the chosen blocksize β.

Assumption 21 *Given as input, a basis B of a d-dimensional lattice Λ, BKZ outputs a vector of norm close to $\delta^d \cdot \det(\Lambda)^{1/d}$ with balanced coordinates. Each coordinate of this vector follows a distribution that can be approximated by a Gaussian with mean 0 and standard deviation $\delta^d \det(\Lambda)^{1/d}/\sqrt{d}$.*

Assumption 22 *BKZ with blocksize β achieves Hermite factor*

$$\delta = \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}.$$

This assumption is experimentally verified in [15].

BKZ cost models. To estimate the running time of BKZ, we use three different models. The first model is an extrapolation by Albrecht [3] et al. of the Liu–Nguyen datasets [35]. According to that model, the logarithm of the running time of BKZ-2.0 (expressed in bit operations) is a quadratic function of $\log(\delta)^{-1}$:

$$\log(T(\mathrm{BKZ}_\delta)) = \frac{0.009}{\log(\delta)^2} - 27.$$

We further refer to this model as the delta-squared model. The model was used in [20] to estimate the security of TFHE.

Another cost model [4] assumes that the running time of BKZ with blocksize β for d-dimensional basis is $T(\mathrm{BKZ}_{\beta,d}) = 8d \cdot T(\mathrm{SVP}_\beta)$, where $T(\mathrm{SVP}_\beta)$ is the running time of an SVP oracle in dimension β. For the SVP oracle, we use the following two widely used models:

$$\begin{aligned} \text{Sieving model:} \quad & T(\mathrm{SVP}_\beta) \approx 2^{0.292\beta+16.4}, \\ \text{Enumeration model:} \quad & T(\mathrm{SVP}_\beta) \approx 2^{0.187\beta\log(\beta)-1.019\beta+16.1}. \end{aligned}$$

Analysing the proof of the sieving algorithm [8] reveals that around $\left(\frac{4}{3}\right)^{\frac{n}{2}}$ short vectors while solving SVP on an n-dimensional lattice. Therefore, when using the sieving model, we shall assume that one run of the BKZ routine produces $\left(\frac{4}{3}\right)^{\frac{\beta}{2}}$ short lattice vectors, where β is the chosen blocksize. As such, we shall provide the following heuristic, which generalizes the repartition given in Assumption 21 when the number of output vectors is small with regards to the number of possible vectors of the desired length:

Assumption 23 *Let $R \ll \delta^{d^2} V_d$ and $R \leqslant (4/3)^{\beta/2}$ where V_d is the volume of the ℓ_2 unit ball in dimension d. Given as input, a basis B of a d-dimensional lattice Λ, BKZ_β with a sieving oracle as SVP oracle outputs a set of R vectors of norm close to $\delta^d \cdot \det(\Lambda)^{1/d}$ with balanced coordinates. Each coordinate of these vector follows a distribution that can be approximated by a Gaussian with mean 0 and standard deviation $\delta^d \det(\Lambda)^{1/d}/\sqrt{d}$.*

In practice, for the dimensions involved in cryptography, this assumption can be experimentally verified. In particular, for the parameters tackled in this work, the number of vectors used by the attack is way lower than the number of potential candidates. An experimental verification of this fact is conducted in Section 5.4. In general settings (when we need to look at all the vectors of the sieving pool), one might see this exploitation as a slight underestimate of the resulting security parameters. An interesting open problem that we leave for future work as it is unrelated to the attacks mounted here, would be to quantify precisely the distribution of the sieved vectors.

A related idea seems quite folklore in the lattice reduction community. In particular, in [2], the output basis of the reduction is randomized with slight enumeration and a pass of LLL for instance. This approach is slightly more

costly than just extracting the sieved vectors but is comparable for its effect as an amortization technique when a batch reduction is needed.

2.4 Background on probability theory

Modular Gaussian distribution. Let $\sigma > 0$. For all $x \in \mathbb{R}$, the density of the centered Gaussian distribution with standard deviation σ is defined as $\rho_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\Big(-\frac{x^2}{2\sigma^2}\Big)$. We define the distribution that is obtained by sampling a centered Gaussian distribution of standard deviation σ and reducing it modulo 1 as the *modular Gaussian distribution* of parameter σ and denote it as $\mathcal{G}_\sigma$.

The support of the distribution is $\big(-\frac{1}{2}; \frac{1}{2}\big)$. The probability density function is given by the absolutely convergent series:

$$g_\sigma(x) = \sum_{k\in\mathbb{Z}} \rho_\sigma(x+k).$$

For large values of σ, the sum that defines the density of a modular Gaussian can be closely approximated.

Lemma 1. *As $\sigma \to \infty$, $g_\sigma(x) = 1 + 2e^{-2\pi^2\sigma^2}\cos(2\pi x) + O(e^{-8\pi^2\sigma^2})$.*

Proof. The Fourier transform of the Gaussian function $\rho_{\sigma,m}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x+m)^2}{2\sigma^2}}$ is given by $\hat{\rho}_{\sigma,m}(y) = e^{-2\pi^2\sigma^2 m^2 + 2\pi i m x}$. Then, using the Poisson summation formula, we obtain:

$$\begin{aligned} g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \sum_{k\in\mathbb{Z}} e^{-\frac{(k+x)^2}{2\sigma^2}} = 1 + 2\sum_{k>0} e^{-2\pi^2\sigma^2k^2}\cos(2\pi k x) = \\ 1 + 2e^{-2\pi^2\sigma^2}\cos(2\pi x) + O(e^{-8\pi^2\sigma^2}). \end{aligned} \tag{1}$$

Probability background

Berry-Esseen inequality. The Berry-Esseen inequality shows how closely the distribution of the sum of independent random variables can be approximated by a Gaussian distribution.

Theorem 1. *Let $X_1, \ldots, X_n$ be independent random variables such that for all $i \in \{1, \ldots, n\}$ $\mathbb{E}\{X_i\} = 0$, $\mathbb{E}\{X_i^2\} = \sigma_i^2 > 0$, and $\mathbb{E}\{|X_i|^3\} = \rho_i < \infty$. Denote the normalized sum $\left(\sum_{i=1}^n X_i\right)^{-1} \sqrt{\sum_{i=1}^n \sigma_i^2}$ as S_n. Also denote by F_n the cumulative distribution function of S_n, and by Φ the cumulative distribution function of the standard normal distribution. Then, there exists a constant C_0 such that*

$$\sup_{x\in\mathbb{R}} |F_n(x) - \Phi(x)| \leqslant C_0 \frac{\sum_{i=1}^n \rho_i}{\Big(\sum_{i=1}^n \sigma_i^2\Big)^{3/2}}.$$

We use the Berry-Esseen inequality in order to estimate how closely the distribution that we obtain after the lattice reduction step of the dual attack can be approximated by a discrete Gaussian distribution (see Lemma 3). The Berry-Esseen inequality requires a finite third absolute moment of the random variables. In the proof of Lemma 3, we need the expression of third absolute moment of a Gaussian distribution. It can be obtained from the following lemma.

Lemma 2. *Let $\sigma > 0$. Let X be a random variable of a Gaussian distribution with mean 0 and standard deviation σ^2. Then, $\mathbb{E}\{|X|^3\} = 2\sqrt{\frac{2}{\pi}}\sigma^3$.*

Proof. Classically we have: $\mathbb{E}\{|X|^3\} = 2 \cdot \frac{1}{\sqrt{2\pi}\sigma} \int_0^\infty x^3 e^{-\frac{x^2}{2\sigma^2}} dx = 2\sqrt{\frac{2}{\pi}}\sigma^3$.

Hoeffding's inequality. Hoeffding's inequality gives an exponentially decreasing upper bound on the probability that the sum of bounded independent random variables deviates from its expectation by a certain amount.

Theorem 2. *Let $X_1, \ldots, X_N$ be independent random variables such that $a_i \leqslant X_i \leqslant b_i$ for all $i \in \{1, \ldots, N\}$. Denote the average $\frac{1}{N}\sum_{i=1}^{N} X_i$ as $\bar{X}$. Then, for $t > 0$, we have*

$$\mathbb{P}\{\left|\bar{X} - \mathbb{E}\{\bar{X}\}\right| \geqslant t\} \leqslant 2\exp\left(-\frac{2N^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right) \tag{2}$$

In this paper, we use Hoeffding's inequality to construct a distinguisher for the uniform and the modular Gaussian distributions (see Section 3.2).

3 Dual distinguishing attack against LWE.

In this first section, we revisit the distinguishing dual attack against LWE (or more precisely for the generic corresponding scale-invariant problem described in [11,21]), providing complete proofs and introducing finer tools as a novel distinguisher for the uniform distribution and the modular Gaussian. In particular, all the results are *non-asymptotic* and can be used for practical instantiations of the parameters. Note that it also naturally encompasses a continuous relaxation of the *lazy modulus switching* technique of [2], as the mathematical framework used makes it appear very naturally in the proof technique.

Settings. In all of the following, we denote by B a finite set of integers (typically $\{0,1\}$ or $\{-1,0,1\}$). Let $\mathbf{s} \in B^n$ be a secret vector and let $\alpha > 0$ be a fixed constant. The attack takes as input m samples $(\mathbf{a}_1, b_1), \ldots, (\mathbf{a}_m, b_m) \in \mathbb{T}^{n+1} \times \mathbb{T}$ which are either all from $\mathrm{LWE}_{\mathbf{s},\alpha}$ distribution or all from $\mathcal{U}(\mathbb{T}^n \times \mathbb{T})$, and guesses the input distribution.

We can write input samples in a matrix form:

$$\mathbf{A} := (\mathbf{a}_1, \ldots, \mathbf{a}_m) \in \mathbb{T}^{n \times m}, \quad \mathbf{b} = (b_1, \ldots, b_m)^t \in \mathbb{T}^m,$$

if input samples are from the $\mathrm{LWE}_{\mathbf{s},\alpha}$ distribution: $\mathbf{b} = \mathbf{A}^t\mathbf{s} + \mathbf{e} \mod 1$.

Distinguisher reduction using a small trapdoor. To distinguish between the two distributions, the attack searches for a short vector $\mathbf{v} = (v_1, \ldots, v_m)^t \in \mathbb{Z}^m$ such that the linear combination of the left parts of the inputs samples defined by $\mathbf{v}$, i.e.:

$$\mathbf{x} := \sum_{i=1}^{m} v_i \mathbf{a}_i = \mathbf{A}\mathbf{v} \mod 1$$

is also a short vector. If the input was from the LWE distribution, then the corresponding linear combination of the right parts of the input samples is also small as a sum of two relatively small numbers:

$$\mathbf{v}^t\mathbf{b} = \mathbf{v}^t(\mathbf{A}^t\mathbf{s} + \mathbf{e}) = \mathbf{x}^t\mathbf{s} + \mathbf{v}^t\mathbf{e} \mod 1. \tag{3}$$

On the other hand, if the input is uniformly distributed, then independently of the choice of the non-zero vector $\mathbf{v}$, the product $\mathbf{v} \cdot \mathbf{b} \mod 1$ has uniform distribution on $(-1/2; 1/2)$. Recovering a suitable $\mathbf{v}$ thus turns the decisional-LWE problem into an easier problem of distinguishing two distributions on $\mathbb{T}$.

This remaining part of this section is organized in the following way. First, in Section 3.1 we describe how such a suitable vector $\mathbf{v}$ can be discovered by lattice reduction and analyze the distribution of $\mathbf{v}^t\mathbf{b}$. Then, in Section 3.2, we estimate the complexity of distinguishing two distributions on $\mathbb{T}$ that we obtain after this first part. Finally, Section 3.3 estimates the time complexity of the whole attack.

3.1 Trapdoor construction by lattice reduction

Finding a vector $\mathbf{v}$ such that both parts of the sum (3) are small when the input has LWE distribution is equivalent to finding a short vector in the following $(m+n)$-dimensional lattice:

$$\mathcal{L}(\mathbf{A}) = \left\{ \begin{pmatrix} \mathbf{A}\mathbf{v} \mod 1 \\ \mathbf{v} \end{pmatrix} \in \mathbb{R}^{m+n} \;\middle|\; \forall\, \mathbf{v} \in \mathbb{Z}^m \right\}.$$

The lattice $\mathcal{L}(\mathbf{A})$ can be generated by the columns of the following matrix:

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_n & \mathbf{A} \\ \mathbf{0}^{m \times n} & \mathbf{I}_m \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$$

A short vector in $\mathcal{L}(\mathbf{A})$ can be found by applying a lattice reduction algorithm to the basis $\mathbf{B}$. Using Assumption 21, we expect that the lattice reduction process produces a vector $\mathbf{w} = (\mathbf{x}||\mathbf{v})^t \in \mathbb{Z}^{n+m}$ with equidistributed coordinates. Our goal is to minimize the product $\mathbf{v}^t\mathbf{b} = \mathbf{x}^t\mathbf{s} + \mathbf{v}^t\mathbf{e}$. The vectors $\mathbf{e}$ and $\mathbf{s}$ come from

different distributions and have different expected norms. For practical schemes, the variance of $\mathbf{e}$ is much smaller than the variance of $\mathbf{s}$. To take this imbalance into account, one introduces an additional rescaling parameter $q \in \mathbb{R}_{>0}$. The first n rows of the matrix $\mathbf{B}$ are multiplied by q, the last m rows are multiplied by $q^{-n/m}$. Obviously, this transformation doesn't change the determinant of the matrix. A basis $\mathbf{B}_q$ of the transformed lattice is given by

$$\mathbf{B}_q = \begin{pmatrix} q\mathbf{I}_n & q\mathbf{A} \\ \mathbf{0}^{m\times n} & q^{-n/m}\mathbf{I}_m \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}.$$

We apply a lattice reduction algorithm to $\mathbf{B}_q$. Denote the first vector of the reduced basis as $\mathbf{w}_q$. By taking the last m coordinates of $\mathbf{w}_q$ and multiplying them by $q^{n/m}$ we recover the desired vector $\mathbf{v}$. This technique can be thought of as a continuous relaxation of the modulus switching technique. That part of the attack is summarized in Algorithm 3.

The following lemma describes the distribution of the output of Algorithm 3 under Assumption 21 that BKZ outputs vectors with balanced coordinates.

Lemma 3. *Let $\alpha > 0$ be a fixed constant, B a finite set of integers of variance $S^2 = \frac{1}{|B|}\sum_{b\in B} b^2$ and $n \in \mathbb{Z}_{>0}$. Let $\mathbf{s}$ be a vector such that of its coefficients are sampled independently and uniformly in B. Suppose that Assumption 21 holds and let $\delta > 0$ be the quality of the output of the BKZ algorithm. Then, given as input $m = \sqrt{n \cdot \frac{\ln(S/\alpha)}{\ln(\delta)}} - n$ samples from the $LWE_{\mathbf{s},\alpha}$ distribution, Algorithm 3 outputs a random variable x with a distribution that can be approximated by a Gaussian distribution with mean 0 and standard deviation σ*

$$\sigma = \alpha \cdot \exp\Big(2\sqrt{n\ln(S/\alpha)\ln(\delta)}\Big).$$

Denote as F_x the cumulative distribution function of x and denote as Φ_σ the cumulative distribution function of the Gaussian distribution with mean 0 and standard deviation σ. Then, the distance between the two distributions can be bounded: $\sup_{t\in\mathbb{R}} |F_x(t) - \Phi_\sigma(t)| = O\Big(\frac{1}{\sqrt{S^2(m+n)}}\Big)$, as $n \to \infty$.

The crux of the proof of Lemma 3 relies on the Berry-Esseen theorem. We provide the complete details in Section 6.1.

3.2 Exponential kernel distinguisher for the uniform and the modular Gaussian distributions

We now describe a novel distinguisher for the uniform and the modular Gaussian distributions. Formally, we construct a procedure that takes as input N samples which are all sampled independently from one of the two distributions and guesses this distribution.

The crux of our method relies on the use of an empirical estimator of the Levy transform of the distributions, to essentially cancel the effect of the modulus 1 on the Gaussian. Namely, from the N samples $X_1, \ldots, X_N$, we construct the

estimator $\bar{Y} = \frac{1}{N} \cdot \sum_{i=1}^{N} \mathrm{e}^{2\pi i X_i}$. As N is growing to infinity, this estimator converges to the Levy transform at 0 of the underlying distribution, that is to say:

- to 0 for the uniform distribution
- to $e^{-2\pi^2\sigma^2}$ for the modular Gaussian.

Hence, to distinguish the distribution used to draw the samples, we now only need to determine whether the empirical estimator $\bar{Y}$ is closer to 0 or to $e^{-2\pi^2\sigma^2}$. The corresponding algorithm is described in Algorithm 1.

Algorithm 1: Distinguish $\mathcal{U}$ and $\mathcal{G}_\sigma$

input : $X_1, \ldots, X_N \in \left(-\frac{1}{2}; \frac{1}{2}\right)$, $\sigma > 0$, sampled independently from $\mathcal{U}$ or $\mathcal{G}_\sigma$
output: A guess: G if the samples are drawn under $\mathcal{G}_\sigma$ or U otherwise

```
DistinguishGU(X_1, ..., X_N, σ):
    Ȳ = (1/N) · Σ_{i=1}^{N} exp(2πi X_i)
    if (Ȳ ⩽ (1/2) · e^{-2π²σ²}) then return U ;
    return G
```

Lemma 4. *Let $\sigma > 0$ be a fixed constant. Assume that Algorithm 1 is given as input N points that are sampled independently from the uniform distribution $\mathcal{U}$ or from the modular Gaussian distribution $\mathcal{G}_\sigma$. Then, Algorithm 1 guesses the distribution of the input points correctly with a probability at least $p_\sigma = 1 - \exp\left(-\frac{e^{-4\pi^2\sigma^2}}{8} \cdot N\right)$. The time complexity of the algorithm is polynomial in the size of the input.*

Proof. For all $i \in \{1, \ldots, N\}$, denote $e^{2\pi i X_i}$ as Y_i. As $X_i \in \left(-\frac{1}{2}, \frac{1}{2}\right)$, $\Re(Y_i) \in (-1; 1]$. First, we compute the expectation of $\bar{Y} = \frac{1}{N} \cdot \sum_{i=1}^{N} Y_i$ in the two possible cases where X_is are sampled from the uniform distribution, and where X_is are sampled from the modular Gaussian with standard deviation σ. Note that, in both cases, as X_is are sampled independently and identically from the same distribution, $\mathbb{E}\{\bar{Y}\} = \mathbb{E}\{Y_i\}$.

In case of the uniform distribution, the expectation of the real part of $\bar{Y}$ is equal to zero, because the function $\Re(e^{2\pi i x})$ is symmetric around the origin:

$$\mathbb{E}_U\{\Re(\bar{Y})\} = \int_{-1/2}^{1/2} e^{2\pi i x} dx = 0. \tag{4}$$

Now in case of the modular Gaussian distribution, we exploit the 1-periodicity of $t \mapsto e^{2i\pi t}$ to cancel out the modulus 1:

$$\mathbb{E}_G\{\bar{Y}\} = \int_{-1/2}^{+1/2} e^{2\pi i x} \sum_{k\in\mathbb{Z}} \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x+k)^2}{2\sigma^2}} dx \tag{5}$$

$$= \sum_{k\in\mathbb{Z}} \int_{-1/2}^{+1/2} \frac{e^{2\pi i x}}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x+k)^2}{2\sigma^2}} dx \tag{6}$$

$$= \int_{-\infty}^{+\infty} \frac{e^{2\pi i x}}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} dx = \frac{e^{-2\pi^2\sigma^2}}{\sqrt{2\pi}\sigma} \int_{-\infty}^{+\infty} e^{-\frac{(x-2i\pi\sigma)^2}{2\sigma^2}} dx = e^{-2\pi^2\sigma^2}, \tag{7}$$

the sum-integral exchange being justified by uniform convergence of the sum.

Now, using the expectations of $\bar{Y}$ and the Hoeffding's inequality, we can estimate the probability of getting a correct guess.

First, consider the probability wrongly guessing when the distribution of the input is uniform. By Line 3 of Algorithm 1, it is given by:

$$\mathbb{P}\{G|U\} = \mathbb{P}_U\{\bar{Y} > \frac{1}{2} \cdot e^{-2\pi^2\sigma^2}\}.$$

Since Y_is are bounded, i.e., for all $i \in \{1, \ldots, N\}$, $Y_i \in (-1; 1]$, we can use Hoeffding's inequality (see Theorem 2) to bound the probability $\mathbb{P}\{G|U\}$:

$$\mathbb{P}\{G|U\} \leqslant \exp\Big(-\frac{e^{-4\pi^2\sigma^2}}{8} \cdot N\Big). \tag{8}$$

Similarly, we get the same bound on the probability of the wrong guess when the distribution of the input is the modular Gaussian:
$\mathbb{P}\{U|G\} \leqslant \exp\Big(-\frac{e^{-4\pi^2\sigma^2}}{8} \cdot N\Big)$. Together with Equation (8), we get the bound on the probability of the successful guess.

Since Algorithm 1 consists of computing the average of the input sample and performing one comparison, it is polynomial in the size of the input.

Lemma 4 implies that to distinguish the uniform distribution and the modular Gaussian distribution with the parameter σ with a non-negligible probability, we need to take a sample of size $N = O(e^{4\pi^2\sigma^2})$.

Remark 1. The dual attack proposed in [21], does not specify, which algorithm is used for distinguishing the uniform and the modular Gaussian distributions. Instead, to estimate the size of the sample, needed to distinguish the distributions, they estimate the statistical distance ε (see [21, Section 7, Equation(6)] and use $O(1/\varepsilon^2)$ as an estimate for the required size of the sample. However, such an estimate does not allow a practical instantiation in the security analysis since it hides the content of the O. It turns out that the exponential kernel distinguisher, described in Algorithm 1 has the same asymptotic complexity as the statistical distance estimate from [21] *suggests*, while enjoying a sufficiently precise analysis to provide non-asymptotic parameter estimation.

3.3 Complexity of the dual attack

The distinguishing attack is summarized in Algorithm 2. It takes as input $m \times N$ samples from an unknown distribution, then transforms them into N samples which have the uniform distribution if the input of the attack was uniform and the modular Gaussian distribution if the input was from the LWE distribution. Then, the attack guesses the distribution of N samples using Algorithm 1 and outputs the corresponding answer.

Algorithm 2: Dual distinguishing attack

input : $\{(\mathbf{A}_i, \mathbf{b}_i)\}_{i=1}^{N}$, where $\forall i$ $\mathbf{A}_i \in \mathbb{T}^{n \times m}$, $\mathbf{b}_i \in \mathbb{T}^m$, $\alpha > 0$, $S > 0$, $\delta \in (1; 1.1)$

output: guess for the distribution of the input: Uniform or LWE distribution

```
DistinguishingAttack({A_i, b_i}_{i=0}^N, α, S, δ):
    X := ∅
    σ := α · exp(2·sqrt(n ln(S/α) ln(δ)))
    for i ∈ {1, ..., N} do
        x ← LWEtoModGaussian(A_i, b_i, S, α, δ)
        X ← X ∪ x
    if (DistinguishGU(X, σ) = G) then
        return LWE distribution
    else
        return Uniform
```

The following theorem states that the cost of the distinguishing attack can be estimated by solving a minimization problem. The proof is deferred to Section 6.1.

Theorem 3. *Let $\alpha > 0$ be a fixed constant, B a finite set of integers of variance $S^2 = \frac{1}{|B|}\sum_{b \in B} b^2$ and $n \in \mathbb{Z}_{>0}$. Let $\mathbf{s}$ be a vector with all coefficients sampled independently and uniformly in B. Suppose that Assumption 21 holds. Then, the time complexity of solving Decision-LWE$_{\mathbf{s},\alpha}$ with probability of success p by the distinguishing attack described in Algorithm 2 is*

$$T_{DualAttack} = \min_{\delta} \Big(N(\sigma, p) \cdot T(BKZ_\delta) \Big), \tag{9}$$

where $\sigma = \alpha \cdot \exp\left(2\sqrt{n \ln(S/\alpha) \ln(\delta)}\right)$, $N(\sigma, p) = 8 \ln(\frac{1}{1-p}) \cdot e^{4\pi^2\sigma^2}$.

Proof. The cost of the attack is the cost of the lattice reduction multiplied by the number of samples N needed to distinguish the uniform distribution and the modular Gaussian distribution with the parameter σ:

$$T = N \cdot T(\mathrm{BKZ}_\delta). \tag{10}$$

By Lemma 4, Algorithm 1, given as an input a sample of size N, guesses its distribution correctly with the probability at least $1 - \exp\Big(- N \cdot \frac{e^{-4\pi^2\sigma^2}}{8} \Big)$.

Thus, in order to achieve the probability p, we need to produce a sample of size $N(\sigma, p) = 8\ln(\frac{1}{1-p}) \cdot e^{4\pi^2\sigma^2}$.

The parameter σ of the discrete Gaussian distribution as a function of δ can be estimated using Lemma 3. Then, the time complexity can be obtained by optimizing the expression, given by Equation (10), as a function of δ.

4 Towards a hybrid dual key recovery attack

In this section, we show how the dual distinguishing attack recalled in Section 3 can be hybridized with exhaustive search on a fraction of the secret vector to obtain a continuum of more efficient key recovery attacks on the underlying LWE problem. Recall that B is a finite set of integers from which the coefficients of the secret are drawn. Let then $\mathbf{s} \in B^n$ be a secret vector and let $\alpha > 0$ be a fixed constant. Our approach takes as input samples from the LWE distribution of form

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}^t\mathbf{s} + \mathbf{e} \mod 1) \in (\mathbb{T}^{n\times m}, \mathbb{T}^m), \tag{11}$$

where $\mathbf{e} \in \mathbb{R}^m$ has centered Gaussian distribution with standard deviation α. The attack divides the secret vector into two fractions:

$$\mathbf{s} = (\mathbf{s}_1||\mathbf{s}_2)^t, \quad \mathbf{s}_1 \in B^{n_1}, \quad \mathbf{s}_2 \in B^{n_2}, \quad n = n_1 + n_2.$$

The matrix $\mathbf{A}$ is divided into two parts corresponding to the separation of the secret $\mathbf{s}$:

$$\mathbf{A} = \left(\begin{array}{ccc} a_{1,1} & \dots & a_{1,m} \\ \vdots & & \vdots \\ a_{n_1,1} & \dots & a_{n_1,m} \\ \hline a_{n_1+1,1} & \dots & a_{n_1+1,m} \\ \vdots & \dots & \vdots \\ a_{n,1} & \dots & a_{n,m} \end{array}\right) = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix} \tag{12}$$

Then, Equation (11) can be rewritten as

$$\mathbf{A}_1^t\mathbf{s}_1 + \mathbf{A}_2^t\mathbf{s}_2 + \mathbf{e} = \mathbf{b} \mod 1.$$

By applying lattice reduction to matrix $\mathbf{A}_1$ as described in Algorithm 3, we recover a vector $\mathbf{v}$ such that $\mathbf{v}^t(\mathbf{A}_1^t\mathbf{s}_1 + \mathbf{e})$ is small and it allows us to transforms m input LWE samples $(\mathbf{A}, \mathbf{b}) \in (\mathbb{T}^{n\times m}, \mathbb{T}^m)$ into one new LWE sample $(\hat{\mathbf{a}}, \hat{b}) \in (\mathbb{T}^{n_2}, \mathbb{T})$ of smaller dimension and bigger noise:

$$\underbrace{\mathbf{v}^t\mathbf{A}_2^t}_{\mathbf{a}}\mathbf{s}_2 + \underbrace{\mathbf{v}^t(\mathbf{A}_1^t\mathbf{s}_1 + \mathbf{e})}_{\hat{e}} = \underbrace{\mathbf{v}^t\mathbf{b}}_{\hat{b}} \mod 1. \tag{13}$$

The resulting LWE sample in smaller dimension can be used to find $\mathbf{s}_2$. Let $\mathbf{x} \in B^{n_2}$ be a guess for $\mathbf{s}_2$. If the guess is correct, then the difference

$$\hat{b} - \hat{\mathbf{a}}^t\mathbf{x} = \hat{b} - \hat{\mathbf{a}}^t\mathbf{s}_2 = (\hat{e} \mod 1) \sim \mathcal{G}_\sigma \tag{14}$$

Algorithm 3: Transform m LWE samples to one sample from modular Gaussian distribution

input : $\mathbf{A} \in \mathbb{T}^{n \times m}$, $\mathbf{b} \in \mathbb{T}^m$, $S > 0$, $\alpha > 0$, $\delta \in (1; 1.1)$
output: $x \in \mathbb{T}$

`computeV(`$\mathbf{A}$, S, α, δ`):`
  $q := \left(\frac{S}{\alpha}\right)^{\frac{m}{n+m}}$
  $\mathbf{B}_q := \begin{pmatrix} q\mathbf{I}_n & q\mathbf{A} \\ \mathbf{0}^{m \times n} & q^{-n/m}\mathbf{I}_m \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$
  $\mathbf{w}_q \leftarrow \mathrm{BKZ}_\delta(\mathbf{B}_q)$
  $\mathbf{v} := q^{n/m} \cdot (w_{q_{n+1}}, \dots w_{q_{n+m}})^t$
  return $(\mathbf{v})$
`LWEtoModGaussian(`$\mathbf{A}$, $\mathbf{b}$, S, α, δ`):`
  $\mathbf{v} \leftarrow \mathrm{COMPUTEV}(\mathbf{A}, S, \alpha, \delta)$
  return $\mathbf{v}^t\mathbf{b} \mod 1$

is small.

If the guess is not correct and $\mathbf{x} \neq \mathbf{s}_2$, then there exist some $\mathbf{y} \neq \mathbf{0}$ such that $\mathbf{x} = \mathbf{s}_2 - \mathbf{y}$. Then, we rewrite $\hat{b} - \hat{\mathbf{a}}^t\mathbf{x}$ in the following way:

$$\hat{b} - \hat{\mathbf{a}}^t\mathbf{x} = (\hat{b} - \hat{\mathbf{a}}^t\mathbf{s}_2) + \hat{\mathbf{a}}^t\mathbf{y} = \hat{\mathbf{a}}^t\mathbf{y} + \hat{e}.$$

We can consider $(\hat{\mathbf{a}}, \hat{\mathbf{a}}^t\mathbf{y} + \hat{e})$ as a sample from the LWE distribution that corresponds to the secret $\mathbf{y}$. Therefore, we may assume that if $\mathbf{x} \neq \mathbf{s}_2$, the distribution of $\hat{b} - \hat{\mathbf{a}}^t\mathbf{x} \mod 1$ is close to uniform, unless the decision-LWE is easy to solve.

In order to recover $\mathbf{s}_2$, the attack generates many LWE samples with reduced dimension. Denote by R the number of generated samples and put them into matrix form as $(\hat{\mathbf{A}}, \hat{\mathbf{b}}) \in \mathbb{T}^{n_2 \times R} \times \mathbb{T}^R$. There are $|B|^{n_2}$ possible candidates for $\mathbf{s}_2$. For each candidate $\mathbf{x} \in B^{n_2}$, the attack computes an R-dimensional vector $\mathbf{e_x} = \mathbf{b} - \mathbf{A}^t\mathbf{s}$. The complexity of this computation for all the candidates is essentially the complexity of multiplying the matrices $\hat{\mathbf{A}}$ and $\mathbf{S}_2$, where $\mathbf{S}_2$ is a matrix whose columns are all vectors of (the projection of) the secret space in dimension n_2. Naively, the matrix multiplication requires $O(n \cdot |B|^{n_2} \cdot R)$ operations. However, by exploiting the recursive structure of $\mathbf{S}_2$, it can be done in time $O(R \cdot |B|^{n_2})$.

Then, for each candidate $\mathbf{x}$ for $\mathbf{s}_2$ the attack checks whether the corresponding vector $\mathbf{e_x}$ is uniform or concentrated around zero distribution. The attack returns the only candidate $\mathbf{x}$ whose corresponding vector $\mathbf{e_x}$ has concentrated around zero distribution.

The rest of this section is organized as follows. First, we describe the auxiliary algorithm for multiplying a matrix by the matrix of all vectors of the secret space that let us speed up the search for the second fraction of the secret key. Then, we evaluate the complexity of our attack.

4.1 Algorithm for computing the product of a matrix with the matrix of all vectors in a product of finite set

Let $B = \{b_1, \ldots, b_k\} \subset \mathbb{Z}$ be a finite set of integer numbers such that $b_i < b_{i+1}$ for all $i \in \{1, \ldots, k-1\}$. For any positive integer d, denote by $\mathbf{S}_{(d)}$ the matrix whose columns are all vectors from $\{b_1, \ldots, b_k\}^d$ written in the lexicographical order. These matrices can be constructed recursively. For $d = 1$ the matrix is a single row, i.e., $\mathbf{S}_{(1)} = \begin{pmatrix} b_1 \ldots b_k \end{pmatrix}$, and for any $d > 1$ the matrix $\mathbf{S}_{(d)} \in \mathbb{Z}^{d \times k^d}$ can be constructed by concatenating k copies of the matrix $\mathbf{S}_{(d-1)}$ and adding a row which consists of k^{d-1} copies of b_1 followed by k^{d-1} copies of b_2 and so on:

$$\mathbf{S}_{(d)} = \begin{pmatrix} b_1 \ldots b_1 \; b_2 \ldots b_2 \; \ldots \; b_k \ldots b_k \\ \mathbf{S}_{(d-1)} \quad \mathbf{S}_{(d-1)} \quad \ldots \quad \mathbf{S}_{(d-1)} \end{pmatrix}. \tag{15}$$

Let $\mathbf{a} = (a_1, \ldots, a_d)^t$ be a d-dimensional vector. Our goal is to compute the scalar products of $\mathbf{a}$ with each column of $\mathbf{S}_{(d)}$. We can do it by using the recursive structure of $\mathbf{S}_{(d)}$. Assume that we know the desired scalar products for $\mathbf{a}_{(d-1)} = (a_2, \ldots, a_d)^t$ and $\mathbf{S}_{(d-1)}$ Then, using Equation (15), we get

$$\begin{aligned} \mathbf{a}^t \mathbf{S}_{(d)} &= \begin{pmatrix} a_1 \; \mathbf{a}_{(d-1)}^t \end{pmatrix} \cdot \begin{pmatrix} b_1 \ldots b_1 \; \ldots \; b_k \ldots b_k \\ \mathbf{S}_{(d-1)} \quad \ldots \quad \mathbf{S}_{(d-1)} \end{pmatrix} \\ &= \Big((a_1 \cdot b_1, \ldots, a_1 \cdot b_1)^t + \mathbf{a}_{(d-1)}^t \mathbf{S}_{(d-1)} \; \Big|\Big| \; \ldots \; \Big|\Big| \\ &\qquad\qquad (a_1 \cdot b_k, \ldots, a_1 \cdot b_k)^t + \mathbf{a}_{(d-1)}^t \mathbf{S}_{(d-1)} \Big) \end{aligned} \tag{16}$$

that is, the resulting vector is the sum of the vector $\mathbf{a}_{(d-1)}^t \mathbf{S}_{(d-1)}$ concatenated with itself k times with the vector consisting of k^{d-1} copies of $a_1 \cdot b_1$ concatenated with k^{d-1} copies of $a_1 \cdot b_2$ and so on. The approach is summarized in Algorithm 4.

Algorithm 4: Compute a scalar product of a matrix of all vectors from $\{b_1, \ldots, b_k\}^d$.

input : $\mathbf{a} = (a_1, \ldots, a_d)^t$, $B = \{b_1, \ldots, b_k\} \subset \mathbb{Z}$ such that $b_1 < b_2 < \cdots < b_k$.
output: $\mathbf{a}^t \mathbf{S}_{(d)}$, where $\mathbf{S}_{(d)} \in \{b_1, \ldots, b_k\}^{k^d \times d}$ is the matrix whose columns are all the vectors from the set $\{b_1, \ldots, b_k\}^d$ written in the lexicographical order

```
computeScalarProductWithAllVectors(a, B):
    x ← (a_d · b_1, ..., a_d · b_k)^t;  y_1 ← ∅, y_2 ← ∅
    for i ∈ {d − 1, ..., 1} do
        for j ∈ {1, ..., k} do
            y_1 ← y_1 ∪ x
            y_2 ← y_2 ∪ (a_i · b_j, ..., a_i · b_j)^t
        x ← y_1 + y_2
        y_1 ← ∅, y_2 ← ∅
    return x
```

Lemma 5. *Let d be a positive integer number and $B = \{b_1, \ldots, b_k\}$ be a set of k integer numbers. Algorithm 4, given as input a d-dimensional vector $\mathbf{a}$, outputs the vector $\mathbf{x}$ of dimension k^d such that for all $x = \mathbf{a}^t \mathbf{S}_{(d)}$. The time complexity of the algorithm is $O(k^d)$.*

Proof (of Lemma 5). The correctness of the algorithm follows from the recursive structure of the matrix $\mathbf{S}_{(d)}$ (see Equations (15) and (16)). At the i-th iteration of the cycle (3-8) the algorithm performs k multiplications and k^{i+1} additions. Number of iterations is $(d-1)$. Then, the overall number of multiplications is $(d-1) \cdot k$ and the overall number of additions is $k + k^2 + \cdots + k^d = \frac{k^{d+1}-1}{k-1} - 1 = O(k^d)$.

Corollary 1. *Let $\mathbf{A}$ be a matrix with R rows and d columns. The product of $\mathbf{A}$ and $\mathbf{S}_{(d)}$ can be computed in time $O(R \cdot k^d)$.*

Proof. In order to compute $\mathbf{A} \cdot \mathbf{S}_{(d)}$ we need to compute the product of each of the R rows of A with $\mathbf{S}_d$. By Lemma 5 it can be done in time $O(k^d)$. Then the overall complexity of multiplying the matrices is $O(R \cdot k^d)$.

4.2 Complexity of the attack

The pseudo-code corresponding to the full attack is given in Algorithm 5.

Theorem 4. *Let $\alpha > 0$, $p \in (0;1)$, $S \in (0;1)$, and $n \in \mathbb{Z}_{>0}$ be fixed constants. Let $\mathbf{s} \in B^n$ and $\sigma > 0$. Suppose that Assumption 21 holds. Then, the time complexity of solving the Search-LWE$_{\mathbf{s},\alpha}$ problem with the probability of success p by the attack described in Algorithm 5 is*

$$T_{dual\ hybrid} = \min_{\delta, n_2} \left(\left(|B|^{n_2} + T(BKZ_\delta) \right) \cdot R(n_2, \sigma, p) \right), \tag{17}$$

where $R(n_2, \sigma, p) = 8 \cdot e^{4\pi^2 \sigma^2} (n_2 \ln(2) - \ln(\ln(1/p)))$.

Proof. The attack can be divided in two steps: the lattice reduction step and the exhaustive search for the second fraction of the secret key. The first step of the attack takes $R \times m$ LWE$_{\mathbf{s},\alpha}$ samples and transforms them into R LWE$_{\mathbf{s}_2,\sigma}$ samples such that $\mathbf{s}_2$ is the second fraction of the secret key $\mathbf{s}$ and the noise parameter σ is bigger than the noise parameter α of the input. It takes time $R \cdot T(\text{BKZ}_\delta)$. Denote the matrix form of obtained LWE samples as $(\hat{\mathbf{A}}, \hat{\mathbf{b}}) \in (\mathbb{T}^{n_2 \times R}, \mathbb{T}^R)$.

At the search step, the goal is to recover $\mathbf{s}_2$ using the obtained LWE samples. For each of the candidates for $\mathbf{s}_2$ the attack computes the error vector that corresponds to R LWE samples obtained at the previous step. It is equivalent to computing the following matrix expression:

$$\hat{\mathbf{E}} = \hat{\mathbf{B}} - \hat{\mathbf{A}}^t \mathbf{S}_{(n_2)} \mod 1,$$

where $\mathbf{S}_{(n_2)}$ is the matrix composed of all vectors of the secret space of length n_2 written in lexicographic order and $\hat{\mathbf{B}} \in \mathbb{T}^{R \times |B|^{n_2}}$ is the matrix formed of

Algorithm 5: Hybrid key recovery attack

input : $\{(\mathbf{A}_i, \mathbf{b}_i)\}_{i=1}^{R}$, where $\forall i\ \mathbf{A}_i \in \mathbb{T}^{n\times m}$, $\mathbf{b}_i \in \mathbb{T}^m$, $\alpha > 0$, $S > 0$, $\delta > 1$, $n_1 \in \{2, \ldots, n-1\}$

output: $\mathbf{s}_2 \in B^{n-n_1}$

recoverS$(\{(\mathbf{A}_i, \mathbf{b}_i)\}_{i=1}^{R}, \alpha, S, \delta, n_1)$**:**

  $n_2 \leftarrow (n - n_1)$

  $\sigma \leftarrow \alpha \cdot \exp\left(2\sqrt{n_1 \ln(S/\alpha)\ln(\delta)}\right)$

  $\hat{\mathbf{A}} \leftarrow \emptyset$, $\hat{\mathbf{b}} \leftarrow \emptyset$

  /* lattice reduction part */

  for $i \in \{1, \ldots, R\}$ **do**

    $\mathbf{A} \leftarrow \mathbf{A}_i$, $\mathbf{b} \leftarrow \mathbf{b}_i$

    $(\mathbf{A}_1, \mathbf{A}_2) \leftarrow$ splitMatrix$(\mathbf{A}, n_1)$ ▷ see Equation (12)

    $\mathbf{v} \leftarrow$ computeV$(\mathbf{A}_1, S, \alpha, \delta)$ ▷ Algorithm 3

    $\hat{\mathbf{A}} \leftarrow \hat{\mathbf{A}} \cup \{\mathbf{A}_2\mathbf{v}\}$, $\hat{\mathbf{b}} \leftarrow \hat{\mathbf{b}} \cup \{\mathbf{v}^t\mathbf{b}\}$

  /* search for $\mathbf{s}_2$ */

  $\mathbf{S}_{(n_2)} \leftarrow$ matrix of all vectors in the secret space of dimension n_2 in lexicographical order

  $\hat{\mathbf{B}} \leftarrow (\hat{\mathbf{b}}, \ldots, \hat{\mathbf{b}}) \in \mathbb{T}^{R\times|B|^{n_2}}$

  $\hat{\mathbf{E}} \leftarrow \hat{\mathbf{B}} - \hat{\mathbf{A}}^t\mathbf{S}_{(n_2)} \mod 1$ ▷ see Corollary 1 and Algorithm 4

  for $i \in \{1, \ldots, |B|^{n_2}\}$ **do**

    $\hat{\mathbf{e}} \leftarrow \hat{\mathbf{E}}[i]$

    /* guess the distribution of e (see Algorithm 1) */

    if (distinguishGU$(\hat{\mathbf{e}}, \sigma) = \mathcal{G}$) **then**

      return $\mathbf{S}_{(n_2)}[i]$

$|B|^{n_2}$ repetition of the vector $\hat{\mathbf{b}}$. The complexity of computing that expression is dominated by the complexity of computing the product of $\hat{\mathbf{A}}^t \in \mathbb{T}^{R\times n_2}$ and $\mathbf{S}_{(n_2)}$. By Corollary 1, it can be computed in $O(R \cdot |B|^{n_2})$ operations. Once the attack obtain an error vector for each of the candidates, it guesses the distribution of each error vector using Algorithm 1 and returns the candidate whose error vector has concentrated around zero modular Gaussian distribution.

The time complexity of the attack is the sum of the complexities of the two steps:

$$T_{\text{attack}} = R \cdot \left(|B|^{n_2} + T(\text{BKZ}_\delta)\right). \tag{18}$$

Now the goal is to evaluate the number of samples R needed to recover $\mathbf{s}_2$ with probability p. By Lemma 4, using Algorithm 1, we can guess correctly the distribution of a sample of size R with probability at least $p_\sigma = 1 - \exp\left(-\frac{e^{-4\pi^2\sigma^2}}{8} \cdot R\right)$. In order to recover $\mathbf{s}_2$, we need successfully guess the distribution for each of $|B|^{n_2}$ candidates. Assume that the distributions, produced by the candidates are independent. Then, the probability to correctly recover $\mathbf{s}_2$ is at least $p_\sigma^{|B|^{n_2}}$. Thus, to recover $\mathbf{s}_2$ we need to choose the size of the sample R that

satisfies:

$$p_\sigma^{|B|^{n_2}} = \left(1 - \exp\left(-\frac{e^{-4\pi^2\sigma^2}}{8} \cdot R\right)\right)^{|B|^{n_2}} \geqslant p. \tag{19}$$

Let R be given by the following expression:

$$R = 8 \cdot e^{4\pi^2\sigma^2}(n_2 \ln(2) - \ln(\ln(1/p))). \tag{20}$$

Combining Equations (19) and (20), we obtain:

$$p_\sigma^{|B|^{n_2}} = \left(1 - \frac{\ln(1/p)}{|B|^{n_2}}\right)^{|B|^{n_2}}. \tag{21}$$

Then, when $n_2 \to \infty$, $p_\sigma^{|B|^{n_2}} \to p$. Thus, the sample size R, given by Equation (20) is sufficient to recover $\mathbf{s}_2$ with the probability p.

By combining Equations (18) and (20) we obtain the time complexity of the attack.

4.3 Using sieving in the hybrid attack

Assume that BKZ uses a sieving algorithm (for instance [8]) as an SVP oracle. At its penultimate step, the sieving algorithm produces many short vectors. Thus we may suppose that BKZ with the sieving SVP oracle produces many short vectors in one run. Then, if we need N short lattice vectors, we need to run the lattice reduction only $\lceil \frac{N}{m} \rceil$ times, where m is the number of short vectors, returned by the lattice reduction. In the following corollary from Theorem 4, we use this property to revisit the time complexity of our attack under the sieving BKZ cost model.

Corollary 2. *Let α, p, n, σ and $\mathbf{s} \in \{0;1\}^n$ be as in Theorem 4. Assume that the lattice reduction algorithm, used by Algorithm 2, uses the sieving algorithm from [8] as an oracle for solving SVP. Suppose that Assumption 23 holds. Then, the time complexity of solving the Search-LWE$_{\mathbf{s},\alpha}$ problem with probability of success p by the attack described in Algorithm 5 is*

$$T_{hybrid+sieving} = \min_{\delta, n_2}\left(|B|^{n_2} \cdot R(n_1, \sigma, p) + T(BKZ_\delta) \cdot \left\lceil \frac{R(n_2, \sigma, p)}{(4/3)^{\beta/2}} \right\rceil\right), \tag{22}$$

where β is the smallest blocksize such that the lattice reduction with the blocksize β achieves the Hermite factor δ; $R(n_2, \sigma, p)$ is as defined in Theorem 4.

Proof. By Theorem 4, the time complexity of Algorithm 5 can be seen as the sum of complexities of the two parts of the algorithm. The first part is producing R short lattice vectors and the second part is evaluating R scalar products for each of $|B|^{n_2}$ candidates for the secret key. As in the sieving model one run of the lattice reduction produces $(4/3)^{\beta/2}$ short vectors, the first part of Algorithm 5 attack takes time $T(\mathrm{BKZ}_\delta) \cdot \left\lceil \frac{R(n_2,\sigma,p)}{(4/3)^{\beta/2}} \right\rceil$, which implies that the complexity of Algorithm 5 in the sieving BKZ cost model is given by Equation (22).

4.4 The sparse case: size estimation and guessing a few bits

When the secret is sparse we can use so-called combinatorial techniques [2] to leverage this sparsity. Assume that only h components of the secret are non-zero. Then, we guess k zero components of the secret $\vec{s}$ and then run the full attack in dimension $(n-k)$. If the guess was incorrect, we restart with a new and independent guess for the positions of zeroes. For sparse enough secrets, the running time of the attack in smaller dimension trade-offs positively with the failure probability.

Also, the variance of the scalar product $\vec{v}^t\vec{s}$ is smaller in the sparse case because the variance of the key contains many zeros. Combining these observations, we obtain the following result for sparse secrets:

Theorem 5. *Let $\alpha > 0, n > 0$ and fix $\mathbf{s} \in B^n$. Suppose that $\mathbf{s}$ has exactly $0 \leqslant h < n$ non-zero components. Suppose that Assumption 21 holds. Assume that the lattice reduction algorithm, used by Algorithm 2, uses the sieving algorithm from [8] as an oracle for solving SVP. Then, the time complexity of solving Decision-LWE$_{\mathbf{s},\alpha}$ with probability of success p by the distinguishing attack described in Algorithm 2 is given by*

$$T_{sparse} = \min_{0 \leqslant k \leqslant h} \left(\binom{n-h}{k}^{-1} \binom{n}{k} \cdot T_{hybrid}(n-k, \alpha) \right) \tag{23}$$

where β is the smallest blocksize such that the lattice reduction with the blocksize β achieves the Hermite factor δ; σ and $N(\sigma, p)$ are as defined in Theorem 3.

Proof. Asserting that only h coefficients are non-zero yields that the probability of guessing correctly k zero positions in the secret $\vec{s}$ is exactly: $\binom{n-h}{k}\binom{n}{k}^{-1}$. Hence the number of the fresh restarts required to get a constant probability of success is $\binom{n-h}{k}^{-1}\binom{n}{k}$ The end of the proof scheme is the same as for Theorem 4, with the only difference that the variance of the vector $\tilde{e}$ of Equation (14) is now equal to: $\sigma(\delta, n, S, \alpha) = \alpha \cdot \exp\left(2\sqrt{n \ln(S/\alpha) \ln(\delta)}\right)$, for

$$S = \sqrt{\frac{n-h}{n}} \frac{1}{|B|-1} \sum_{b \in B \setminus \{0\}} b^2, \tag{24}$$

instead of being $\frac{1}{|B|}\sum_{b \in B} b^2$.

5 Bit-security estimation and experimental verification

We implement an estimator script for the attack that, given parameters of an LWE problem and a BKZ cost model as an input, finds optimal parameters for the dual attack (see Section 3) and our hybrid attack (see Section 4). Using this script, we evaluate the computational costs for a wide range of small-secret LWE parameters. In this section, we report the results of our numerical estimation and

show that the security level of the TFHE scheme should be updated with regards to the hybrid attack. We also apply our attack to the parameters of FHEW, SEAL, and HElib. For completeness purposes, in Section 6.5 we also provide an in-depth comparison with the primal unique-SVP technique. Eventually, we support our argument by an implementation working on a small example.

5.1 Bit-security of LWE parameters

We numerically estimate the cost of solving LWE problem by the dual attack (as described in Section 3) and by our attacks for all pairs of parameters the (n, α) from the following set: $(n, -\log(\alpha)) \in \{100, 125, \ldots, 1050\} \times \{5, 6.25, \ldots, 38.5\}$. We create a heatmap representing the cost of our attack as a function of parameters n and α.

In Figure 1 we present an estimation of the bit-security of the LWE parameters according to the combination of our attack and the collision attack, with time complexity $2^{n/2}$. Thus, Figure 1 represents the function $\min(T_{\text{ourAttack}}(n, \alpha), 2^{n/2})$, where $T_{\text{ourAttack}}(n, \alpha)$ is the cost of our attack for the parameters n and α. Figure 1 is obtained under the sieving BKZ cost model.

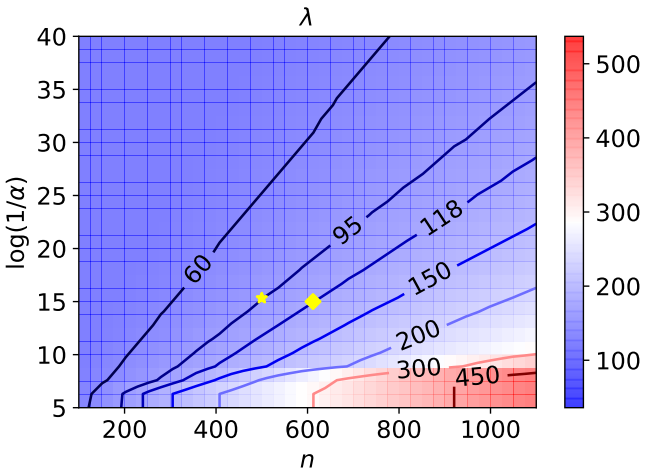


Fig. 1: Bit-security as a function of the LWE parameters n and α assuming the sieving BKZ cost model. Here, n denotes the dimension, α denotes the standard deviation of the noise, the secret key is chosen from the uniform distribution on $\{0, 1\}^n$. The picture represents the security level λ of LWE samples, $\lambda = \log(\min(T_{\text{ourAttack}}(n, \alpha), 2^{n/2}))$. The numbered lines on the picture represent security levels. The star symbol denotes the old TFHE key switching parameters from [20], the diamond symbol denotes the key switching parameters recommended in [21, Table 4].

We also created similar heatmaps for our hybrid dual attack and the dual attack described in Section 3 under three BKZ cost models: enumeration, sieving, and delta-squared. For completeness, these heatmaps are presented in Section 6.4.

5.2 Application to FHE schemes

Non-sparse small secrets. **TFHE.** The TFHE scheme uses two sets of parameters: for the switching key and for the bootstrapping key. The security of the scheme is defined by the security of the switching key, which is the weaker link.

The parameters of the TFHE scheme were updated several times. In Table 2, we presents the results of our estimates for the recently updated parameters from the public implementation [40, v1.1]. For completeness, we also re-evaluate the security all the previous sets of TFHE parameters. The results for the previous parameters of TFHE can be found in Section 6.2.

Table 2: Security of the parameters of the TFHE scheme from the public implementation [40] (parameter's update of February 21, 2020) against dual attack (as described in Section 3) and hybrid dual attack (as described in Section 4). λ denotes security in bits, δ and n_1 are the optimal parameters for the attacks. "-" means that the distinguishing attack doesn't have the parameter n_1.

BKZ model	**switching key** $n = 630$, $\alpha = 2^{-15}$				**bootstrapping key** $n = 1024$, $\alpha = 2^{-25}$			
	attack	λ	δ	n_1	attack	λ	δ	n_1
delta-squared	dual	270	1.0042	-	dual	256	1.0042	-
	new attack	**176**	1.005	485	**new attack**	**190**	1.0048	862
sieving	dual	131	1.0044	-	dual	131	1.0044	-
	new attack	**121**	1.0047	576	**new attack**	**125**	1.0046	967
enumeration	dual	292	1.0042	-	dual	280	1.0041	-
	new attack	**192**	1.0052	469	**new attack**	**209**	1.0049	842

FHEW. The fully homomorphic encryption scheme FHEW [22], as TFHE, uses binary secrets. Its parameters are given as $n = 500, \sigma = 2^{17}, q = 2^{32}$. The bit-security of these parameters under our hybrid dual attack in the sieving model is 96 bits, which is slightly better than the primal or dual attack estimated with [4] giving respectively 101 and 115 bits of security.

SEAL. The SEAL v2.0 homomorphic library [32] uses ternary non-sparse secrets. We target these parameters directly with our hybrid approach and compare the (best) results with the dual attack of [2]. The results are compiled in

Table 7 of Section 6.3. The results are very slightly better for our techniques, although being very comparable.

Sparse secrets: HElib. The HElib homomorphic library [29] uses ternary sparse secrets which have exactly 64 non-zero components. We can then target these parameters using the combination of our hybrid attack with guessing. The results are compiled in Table 8 and Table 9, both given in Section 6.3. The results are very slightly worse for our techniques, although are still very comparable. A reason might be that the exploitation of the sparsity in our case is more naive than the range of techniques used in [2]. An interesting open question would be to merge the best of these two worlds to get even stronger attacks. We leave this question for future work as it is slightly out of the scope of the present paper.

5.3 Experimental verification

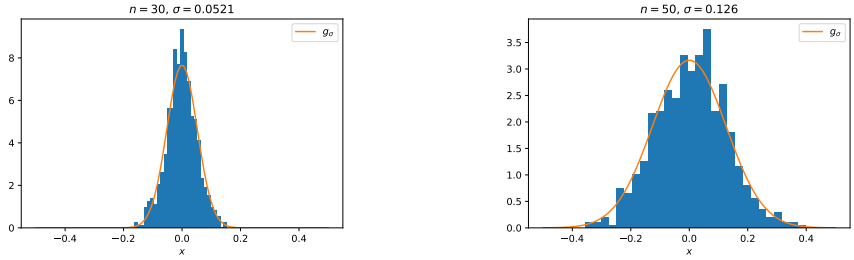
In order to verify the correctness of our attack, we have implemented it on small examples. Our implementation recovers 5 bits of a secret key for LWE problems with the following two sets of parameters: $(n, \alpha) = (30, 2^{-8})$ and $(n, \alpha) = (50, 2^{-8})$. For implementation purposes, we rescaled all the elements defined over torus $\mathbb{T}$ to integers modulo 2^{32}. For both examples, we use BKZ with blocksize 20, which yields the quality of the lattice reduction around $\delta \lesssim 1.013$. We computed the values of parameters of the attack required to guess correctly 5 bits of the key with probability 0.99 assuming that quality of the output of BKZ. The required parameters for both experiments are summarized in Table 3.

$(n, -\log(\alpha))$	m	σ	R
(30,8)	76	0.0521	32
(50,8)	90	0.126	74

Table 3: Parameters required for guessing 5 bits of the key with $\delta = 1.013$. m is the number of samples needed for one lattice reduction (28), σ is the parameter of modular Gaussian distribution $\mathcal{G}_\sigma$ (Lemma 3), R is the number of samples needed to distinguish distributions $\mathcal{G}_\sigma$ and $\mathcal{U}$ (20).

The first experiment was repeated 20 times, the second was repeated 10 times. For both experiments, the last five bits of the key were successfully recovered at all attempts. The correctness of both attacks rely on assumptions made in Lemma 3 for approximating the distribution of $\mathbf{v}^t(\mathbf{A}^t\mathbf{s} + \mathbf{e}) \mod 1$ by modular Gaussian distribution $\mathcal{G}_\sigma$. In order to verify these assumptions, while running both experiments we have collected samples to check the distribution: each time when the attack found correctly the last bits of the secret key $\mathbf{s}_2$, we collected

the corresponding $\tilde{e} = \tilde{\mathbf{b}} - \tilde{\mathbf{a}}^t \mathbf{s}_2 = \mathbf{v}^t(\mathbf{A}^t \mathbf{s}_1 + \mathbf{e})$. For the first experiment, the size of the collected sample is $20 \times R_1 = 640$, for the second experiment, it is $10 \times R_2 = 740$. The collected data is presented in Figure 2.



(a) (b)

Fig. 2: Distribution of $\tilde{e} = \mathbf{v}^t(\mathbf{A}^t \mathbf{s}_1 + \mathbf{e}) \mod 1$.
Figure 3a represents data from the experiment with parameters $(n, \alpha) = (30, 2^{-8})$, figure 3b represents data from the experiment with parameters $(n, \alpha) = (50, 2^{-8})$. Blue histograms denote observed data, orange lines denote theoretical predictions of the distribution.

In Table 4, we compare theoretical predictions and estimations obtained from the experiments for the parameters of modular Gaussian distribution $\mathcal{G}_\sigma$. Experimental estimations of mean and variance in both cases match closely theoretical predictions.

Table 4: Estimated mean and variance for the Gaussian $\mathcal{G}_\sigma$.

(n, α)	sample size	σ	$\mathrm{Var}(\mathcal{G}_\sigma)$	est. variance	average of sample
$(30, 2^{-8})$	640	0.0521	0.00271	0.002619	-0.00207
$(50, 2^{-8})$	740	0.126	0.1587	0.14515	0.0064

5.4 Experimental verification of the assumptions on sieving's output

For the analysis of our attack under the sieving BKZ cost model we assume that the sieving oracle produces at least $\left(\frac{4}{3}\right)^{\frac{\beta}{2}}$ short lattice vectors, where β is the blocksize of BKZ (see Section 2.3). In order to check the correctness of

the assumption, we studied the distribution of vectors returned by the sieving algorithm on random lattices in several small dimensions.

In order to test our assumption, we used the implementation of the GaussSieve algorithm [36] from fplll [24] as model of the sieving oracle. We used knapsack-like random lattices generated by the program *./latticegen* from fplll as input for the GaussSieve algorithm.

First, we look at the distribution of the vectors returned by the GaussSieve algorithm on a random knapsack-like lattice of dimension 50 and volume close to 2^{1000}. Figure 3 represents the histogram of the norms of the vectors returned by the sieving algorithm. For such input lattice we assume that the size of the output list is at least $\left(\frac{4}{3}\right)^{50/2} \approx 1329$ and that the norms of the vectors are close to $\sqrt{n} \cdot \mathrm{vol}(\Lambda)^{1/n} = \sqrt{50} \cdot 2^{1000/50} \approx 7.4 \cdot 10^6$. As Figure 3 shows, GaussSieve returned 3854 different lattice vectors which is almost three times more than expected. Moreover, all the vectors are short enough for our purposes; they all have norms between $2 \cdot 10^6$ and $2.7 \cdot 10^6$ which is of the same order (and even slightly shorter) as we assume.

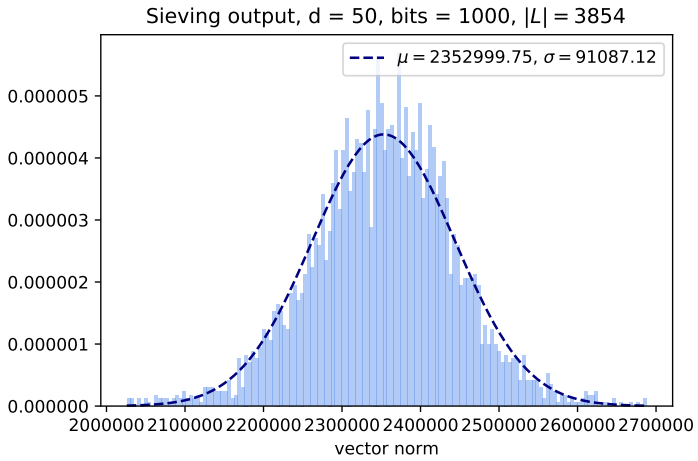


Fig. 3: The histogram of the norms of the vectors returned by the GaussSieve algorithm on a random lattice of dimension 50 and volume $\approx 2^{1000}$. $|L|$ denotes the number of vectors returned by the algorithm, μ is the average norm of the vectors, σ is the estimation of the standard devination of the norm. The dark blue line represents the density function of the normal distribution with parameters μ and σ.

We tried several other combinations of volume and dimension for input lattices and obtained essentially the same results: the size of the list was always a small constant factor bigger than expected $(4/3)^{n/2}$ and the norms of the vectors

in the list were concentrated around a value of order $\sqrt{n} \cdot \mathrm{vol}(\Lambda)^{1/n}$. Figures 4 and 5 represents our results.

In Figure 4, we fix the volume of the lattice to be 2^{1000} and see how the size of the list and the norms of the returned vectors change with the dimension.

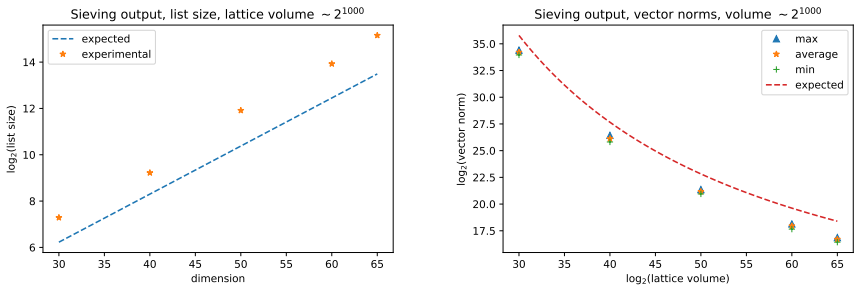


Fig. 4: Output of the GaussSieve on the lattices of dimension 30, 30, 50, 60, and 65 and of volume around 2^{1000}. The left picture represents the logarithm of the number of vectors returned by the algorithm compared to expected $\log\left((4/3)^{d/2}\right)$, the right picture represents the logarithm of the average, minimum, and maximum norms of the returned vectors compared to expected $\log\left(\sqrt{n} \cdot (\mathrm{vol}(\Lambda))^{1/d}\right)$.

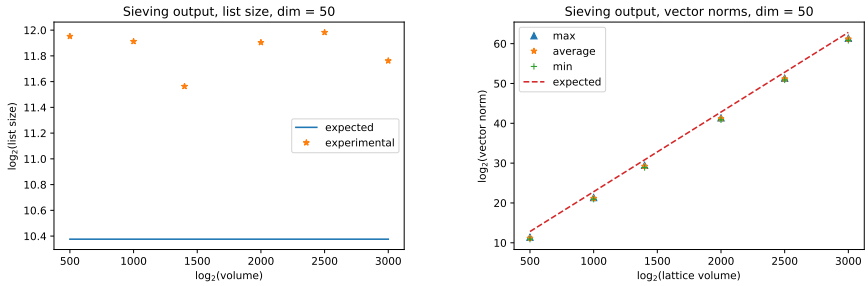


Fig. 5: Output of the GaussSieve on the lattices of dimension 50 and of volumes several volumes from 2^{500} to 2^{3000}. The left picture represents the logarithm of the number of vectors returned by the algorithm compared to expected $\log\left((4/3)^{d/2}\right)$, the right picture represents the logarithm of the average, minimum, and maximum norms of the returned vectors compared to expected $\log\left(\sqrt{n} \cdot (\mathrm{vol}(\Lambda))^{1/d}\right)$.

In Figure 5, we fix the dimension of the lattice to be 50 and see how the size of the list and the norms of the returned vectors change when volume changes.

Acknowledgments

We thank the anonymous reviewers for valuable comments on this work, as well as Alexandre Wallet and Paul Kirchner for interesting discussions. This work has been supported in part by the European Union's H2020 Programme under grant agreement number ERC-669891.

References

1. M. Ajtai. The shortest vector problem in L2 is NP-hard for randomized reductions. In *ACM symposium on Theory of computing*, 1998.
2. M. R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2017.
3. M. R. Albrecht, C. Cid, J.-C. Faugere, R. Fitzpatrick, and L. Perret. On the complexity of the BKW algorithm on LWE. *DCC*, 2015.
4. M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer. Estimate all the {LWE, NTRU} schemes! In *International Conference on Security and Cryptography for Networks*, 2018.
5. M. R. Albrecht, R. Fitzpatrick, and F. Göpfert. On the efficacy of solving LWE by reduction to unique-SVP. In *International Conference on Information Security and Cryptology*, 2013.
6. M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of LWE *Journal of Mathematical Cryptology*, 2015.
7. S. Bai and S. D. Galbraith. Lattice decoding attacks on binary LWE. In *Australasian Conference on Information Security and Privacy*, 2014.
8. A. Becker, L. Ducas, N. Gama, and T. Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, 2016.
9. A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM*, 2003.
10. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM TOCT*, 2014.
11. Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *ACM symposium on Theory of computing*, 2013.
12. Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, 2011.
13. J. Buchmann, F. Göpfert, R. Player, and T. Wunderer. On the hardness of LWE with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack. In *International Conference on Cryptology in Africa*, 2016.
14. H. Chen, K. Laine, and R. Player. Simple encrypted arithmetic library-SEAL v2. 1. In *International Conference on Financial Cryptography and Data Security*, 2017.
15. Y. Chen. *Réduction de réseau et sécurité concrete du chiffrement completement homomorphe*. PhD thesis, 2013.

16. Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology Security*, 2011.
17. J. H. Cheon, M. Hhan, S. Hong, and Y. Son. A hybrid of dual and meet-in-the-middle attack on sparse and ternary secret LWE. *IEEE Access*, 2019.
18. J. H. Cheon and D. Stehlé. Fully homomophic encryption over the integers revisited. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2015.
19. I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *International Conference on the Theory and Application of Cryptology and Security*, 2016.
20. I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for tfhe. In *International Conference on the Theory and Application of Cryptology and Security*, 2017.
21. I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène. Tfhe: fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 2020.
22. L. Ducas and D. Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2015.
23. J. Fan and F. Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*, 012.
24. The FPLLL development team. fplll, a lattice reduction library. Available at https://github.com/fplll/fplll, 2016.
25. N. Gama, P. Q. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2010.
26. C. Gentry et al. Fully homomorphic encryption using ideal lattices. In *STOC*, 2009.
27. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.
28. C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors In *Annual Cryptology Conference*, 2013.
29. S. Halevi and V. Shoup. Bootstrapping for HElib. In *Annual International conference on the theory and applications of cryptographic techniques*, 2015.
30. G. Hanrot, X. Pujol, and D. Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. In *Annual Cryptology Conference*, 2011.
31. N. Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In *Annual International Cryptology Conference*, 2007.
32. K. Laine, R. Player. Simple encrypted arithmetic library-SEAL (v2. 0). *Technical report, Microsoft Research*, 2016.
33. A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 1982.
34. R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In *Cryptographers' Track at the RSA Conference*, 2011.
35. M. Liu and P. Q. Nguyen. Solving BDD by enumeration: An update. In *Cryptographers' Track at the RSA Conference*, 2013.
36. D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, 2010.
37. D. Micciancio and M. Walter. Practical, predictable lattice basis reduction. In *Annual International Conference on the Theory and Applications*, 2016.

38. O. Regev. On lattices, learning with errors, random linear codes, and cryptography, 2005. In *STOC*, 2005.
39. Y. Son and J. H. Cheon. Revisiting the hybrid attack on sparse and ternary secret LWE. *IACR ePrint Archive*, 2019.
40. N. Gama et al. Github repository. TFHE: Fast fully homomorphic encryption library over the torus. https://github.com/tfhe/tfhe, 2016.
41. T. Wunderer. Revisiting the hybrid attack: Improved analysis and refined security estimates. *IACR ePrint Archive*, 2016.

6 Supporting material

6.1 Proof of Lemma 3

Proof. Under Assumption 21, the coordinates of $\mathbf{w}_q$ are independent and distributed according to the Gaussian distribution with expectation 0 and standard deviation $\delta^{n+m}/\sqrt{n+m}$. Since $\mathbf{w}_q = (q\cdot\mathbf{x} \,||\, q^{-n/m}\cdot\mathbf{v})^t$, the coordinates of vectors $\mathbf{x}$ and $\mathbf{v}$ also have centered Gaussian distribution, but with different standard deviations. Let

$$\sigma_{\mathbf{x}} = \frac{1}{q}\cdot\frac{\delta^{m+n}}{\sqrt{m+n}} \quad \text{and} \quad \sigma_{\mathbf{v}} = q^{n/m}\cdot\frac{\delta^{m+n}}{\sqrt{m+n}}$$

be the standard deviation of coordinates of $\mathbf{x}$ and of $\mathbf{v}$ correspondingly. Consider the distribution of

$$\mathbf{v}^t\mathbf{b} = \mathbf{x}^t\mathbf{s} + \mathbf{v}^t\mathbf{e} = \sum_{i=1}^{n} x_i\cdot s_i + \sum_{i=1}^{m} v_i\cdot e_i.$$

$\mathbf{v}^t\mathbf{b}$ is a sum of $m+n$ independent random variables and, therefore, its distribution can be approximated by a Gaussian distribution according to the Central Limit Theorem. In order to learn the parameters of the Gaussian, we need to obtain the expectations and variances of $x_i\cdot s_i$ and $v_i\cdot e_i$.

First, consider the distribution of $x_i\cdot s_i$. As s_i is drawn uniformly in B, $x_i s_i$ is a random variable from the distribution that can be obtained by sampling $b\in B$ with probability B^{-1} and then sampling from a Gaussian distribution with mean 0 and variance $b^2\sigma_{\mathbf{x}}^2$. Therefore, $\mathbb{E}(x_i\cdot s_i) = 0$ and $\mathrm{Var}(x_i\cdot s_i) = S^2\sigma_{\mathbf{x}}^2$ for S^2 being $\mathrm{Var}(B) = \frac{1}{|B|}\sum_{b\in B} b^2$.

Then, consider v_1e_1. As $\mathbf{v}$ and $\mathbf{e}$ are independent and $\mathbb{E}(v_1) = \mathbb{E}(e_1) = 0$, $\mathbb{E}(v_1e_1) = \mathbb{E}(v_1)\mathbb{E}(e_1) = 0$ and $\mathrm{Var}(v_1e_1) = \mathrm{Var}(v_1)\cdot\mathrm{Var}(e_1) = \alpha^2\sigma_{\mathbf{v}}^2$.

Thus, the distribution of $\mathbf{v}^t\mathbf{b}$ is close to the Gaussian distribution with expectation 0 and variance

$$\sigma^2 = n\,\mathrm{Var}(x_1s_1)+m\,\mathrm{Var}(v_1e_1) = nS^2\sigma_{\mathbf{x}}^2+m\alpha^2\sigma_{\mathbf{v}}^2 = \frac{\delta^{2(m+n)}}{m+n}\Big(\frac{nS^2}{q^2}+m\alpha^2q^{2n/m}\Big). \tag{25}$$

Our goal is to obtain a distribution that is as concentrated around zero as possible. Hence we choose parameters m and q in order to minimize variance of $\mathbf{v}^t\mathbf{b}$.

First, we find the optimal value of q by differentiation of Equation (25) :

$$\frac{\partial \sigma^2}{\partial q} = \frac{\delta^{2(m+n)}}{m+n} \cdot \Big(-\frac{2nS^2}{q^3} + \frac{2n}{m} \cdot m\alpha^2 q^{\frac{2n}{m}-1} \Big) = 0 \quad \rightarrow \quad q_{\text{opt}} = \Big(\frac{S}{\alpha}\Big)^{\frac{m}{m+n}}.$$

After replacing q by q_{opt} in Equation (25) we obtain:

$$\sigma^2 = \left(S\delta^{m+n} \Big(\frac{\alpha}{S}\Big)^{\frac{m}{m+n}} \right)^2. \tag{26}$$

Also, for $\sigma_{\mathbf{x}}$ and $\sigma_{\mathbf{v}}$ we obtain the following relation

$$\frac{\sigma_{\mathbf{x}}}{\sigma_{\mathbf{v}}} = \frac{q^{-n/m}}{q} = \frac{\alpha}{S}. \tag{27}$$

Then, we find the optimal value of m by differentiating $\ln(\sigma)$:

$$\frac{\partial \ln(\sigma)}{\partial m} = \ln(\delta) + n \ln\Big(\frac{\alpha}{S}\Big) \cdot \frac{1}{(m+n)^2} = 0 \quad \rightarrow \quad m_{\text{opt}} = \sqrt{n \cdot \frac{\ln(S/\alpha)}{\ln(\delta)}} - n \tag{28}$$

Now, replacing m by m_{opt} in Equation (26), we find:

$$\sigma(\delta, n, S, \alpha) = \sigma(\hat{m}, \delta, n, S, \alpha) = \alpha \cdot \exp\big(2\sqrt{n \ln(S/\alpha) \ln(\delta)}\big).$$

The distance between the distribution of $\mathbf{v}^t\mathbf{b}$ and the Gaussian distribution with mean 0 and variance σ^2 can be estimated by the Berry-Esseen inequality (see Theorem 1). To use this inequality, we need to compute the third absolute moments of $x_1 s_1$ and $v_1 e_1$.

We start with $x_1 s_1$. As x_1 and s_1 are independent,

$$\mathbb{E}\{|x_1 s_1|^3\} = \mathbb{E}\{|x_1|^3\}\mathbb{E}\{|s_1|^3\}.$$

By Lemma 2, $\mathbb{E}\{|x_1|^3\} = 2\sqrt{2/\pi}\sigma_x^3$. As s_1 has the Bernoulli distribution with parameter S^2, $\mathbb{E}\{|s_1|^3\} = \mathbb{E}\{s_1\} = S^2$. Putting two parts together, we get

$$\rho_{x_1 s_1} = \mathbb{E}\{|x_1 s_1|^3\} = 2\sqrt{2/\pi} S^2 \sigma_{\mathbf{x}}^3. \tag{29}$$

In the same way, we obtain

$$\rho_{v_1 e_1} \mathbb{E}\{|v_1 e_1|^3\} = \frac{8}{\pi} \alpha^3 \sigma_{\mathbf{v}}^3. \tag{30}$$

Denote the cumulative distribution function of $\mathbf{v}^t\mathbf{b}$ by $F_{\mathbf{v}^t\mathbf{b}}$, and denote the cumulative distribution function of the Gaussian distribution with mean 0 and variance σ^2 by Φ_σ. By the Berry-Esseen inequality, there exists a constant C_0 such that

$$\sup_{x \in \mathbb{R}} |F_{\mathbf{v}^t\mathbf{b}}(x) - \Phi_\sigma(x)| \leqslant C_0 \cdot \frac{n\rho_{x_1 s_1} + m\rho_{v_1 e_1}}{(nS^2\sigma_{\mathbf{x}}^2 + m\alpha^2\sigma_{\mathbf{v}}^2)^{3/2}}. \tag{31}$$

Then, using Equations (27) and (29) to (31), for the distance between the distributions we get:

$$\sup_{x \in \mathbb{R}} |F_{\mathbf{v}^t\mathbf{b}}(x) - \Phi_\sigma(x)| \leqslant C_0 \sqrt{\frac{8}{S^2\pi}} \cdot \frac{n + mS\sqrt{8/\pi}}{(m+n)^{3/2}} \leqslant C_0 \cdot \frac{8}{\pi S} \cdot \frac{1}{\sqrt{m+n}}. \tag{32}$$

6.2 Security of TFHE parameters from the papers [19, 21]

In Table 5 we present the results of our estimates for the parameters from [19]; in Table 6, we present the security estimates for more recent parameters from [21].

Table 5: Security of the parameters of the TFHE scheme from [19] (the same as from [21, Table 3]) against dual attack (as described in Section 3) and hybrid dual attack (as described in Section 4). λ denotes security in bits, δ and n_1 are the optimal parameters for the attacks. "-" means that the distinguishing attack doesn't have the parameter n_1.

BKZ model	**switching key** $n = 500$, $\alpha = 2.54 \cdot 10^{-5}$				**bootstrapping key** $n = 1024$, $\alpha = 3.73 \cdot 10^{-9}$			
	attack	λ	δ	n_1	attack	λ	δ	n_1
delta-squared	dual	169	1.0052	-	dual	204	1.0046	-
	new attack	**119**	1.0059	406	**new attack**	**160**	1.0051	889
sieving	dual	102	1.0054	-	dual	117	1.0048	-
	new attack	**94**	1.0058	455	**new attack**	**112**	1.005	972
enumeration	dual	195	1.0052	-	dual	230	1.0046	-
	new attack	**137**	1.0062	388	**new attack**	**180**	1.0052	868

6.3 Attack results on FHE schemes

In this appendix we present the various practical results obtained on the schemes SEAL and HElib.

6.4 Heatmaps for comparing our hybrid dual attack and the dual attack as described in Section 3

In this section, we present the results of comparison of our hybrid dual attack with the dual attack described in Section 3 under three different BKZ cost models. In Figure 6, the left heatmap represents the logarithm of the time complexity of the dual attack while the right heatmap represents the logarithm of the time complexity of our attack. Figure 6 shows that for the same sets of parameters the cost of our attack is always less than or equal to the cost of the dual distinguishing attack and that the difference between the costs of the attacks is bigger when the dimension n and the noise parameter α is bigger.

Figures 7 and 8 represent the similar results for the enumeration and delta-squared BKZ cost models.

Figure 9 presents results similar to Figure 1, but under the enumeration and delta-squared BKZ cost models.

Table 6: Security of the parameters of the TFHE scheme from [21, Table 4]) against dual attack (as described in Section 3), hybrid dual attack (as described in Section 4), and the uSVP attack (denoted as "primal"). λ denotes security in bits, δ and n_1 are the optimal parameters for the attacks. "-" means that the distinguishing attack doesn't have the parameter n_1.

BKZ model	**switching key** $n = 612$, $\alpha = 2^{-15}$				**bootstrapping key** $n = 1024$, $\alpha = 2^{-26}$			
	attack	λ	δ	n_1	attack	λ	δ	n_1
delta-squared	dual	194	1.0045	-	dual	191	1.0045	-
	primal	198	1.0042	-	primal	203	1.0043	-
	new attack	**169**	1.0051	474	**new attack**	**179**	1.0049	871
sieving	dual	144	1.0043	-	dual	134	1.0043	-
	primal	127	1.0045	-	primal	123	1.0043	-
	new attack	**118**	1.0048	559	**new attack**	**120**	1.0047	970
enumeration	dual	239	1.0043	-	dual	222	1.0045	-
	primal	219	1.0045	-	primal	210	1.0043	-
	new attack	**185**	1.0053	457	**new attack**	**179**	1.0049	871

6.5 Comparison with primal uSVP attack

The security of the recent parameters from TFHE's implementation is evaluated using the LWE estimator from [4, 6]. As the results of this estimation suggest, under the sieving BKZ cost model, the best attack against the current parameters of the TFHE scheme among the attacks presented in the LWE estimator is the primal uSVP attack [7] (see also [6, Section 6.3] for the description of the attack). Therefore, it is interesting to compare our hybrid dual attack with the primal uSVP attack on a wider range of parameters.

In order to compare our attack with the primal uSVP attack, we estimate the time complexity of both attacks for each pair of the parameters (n, α) from the following set: $(n, -\log(\alpha)) \in \{200, 250, \ldots, 1450\} \times \{10, 12, \ldots, 48\}$. We evaluate

Table 7: Security of the SEAL v2.0 library parameters against the dual attack from [2] and against our hybrid dual attack. n denotes the dimension, q is the modulus, the standard deviation is given by $\sigma = 3.2$ for sets of parameters.

SEAL params.	**dual attack from [2]**	**our attack**
$n = 1024$, $q = 2^{47.5}$	68	67
$n = 2048$, $q = 2^{95.4}$	69	68
$n = 4096$, $q = 2^{192}$	68	67

Table 8: Security of the HElib library parameters (80-bit) against the dual attack from [2] and against our hybrid dual attack. n denotes the dimension, q is the modulus, the standard deviation is given by $\sigma = 3.2$ for sets of parameters.

HElib params.	**dual attack from [2]**	**our attack**
$n = 1024$, $q = 2^{85.2}$	61	64
$n = 2048$, $q = 2^{85.2}$	65	65
$n = 4096$, $q = 2^{85.3}$	67	69

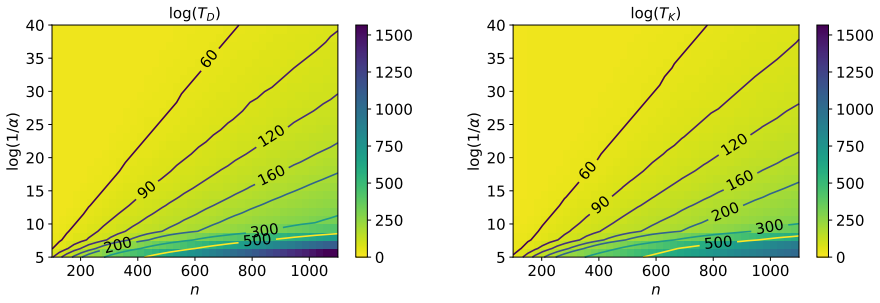
Table 9: Security of the HElib library parameters (120-bit) against the dual attack from [2] and against our hybrid dual attack. Notation is as in Table 8

HElib params.	**dual attack from [2]**	**our attack**
$n = 1024$, $q = 2^{38}$	73	77
$n = 2048$, $q = 2^{70}$	77	79
$n = 4096$, $q = 2^{134}$	81	85

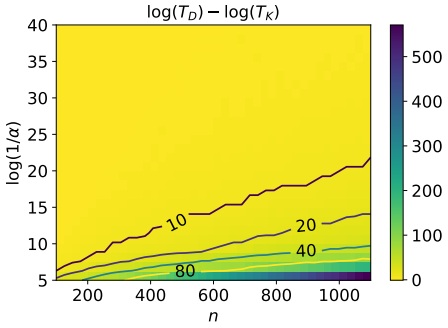
the cost of the primal uSVP attack using the LWE estimator [4, 6]. For this comparison, we consider two BKZ cost models: sieving and enumeration. The results of our estimation are presented in Figures 10 and 11.

Figures 10 and 11 show that under both BKZ cost models, it is not so that one attack is better than another for all the sets of the parameters. Under both BKZ cost models, the primal uSVP attack outperforms the hybrid dual attack when the dimension is high (i.e., $n > 800$) and the noise parameter is small (i.e., $\alpha < 2^{-35}$). For the rest of the parameters that we consider, the hybrid dual attack outperforms the primal uSVP attack. The difference in the cost of the attacks depends on the chosen BKZ cost model; for the enumeration BKZ cost model the difference between attacks in more significant than for the sieving model.

In particular, as reported in Table 1, for the practical security parameters of TFHE, the hybrid dual attack we propose in this work is slightly better than the primal attack technique.

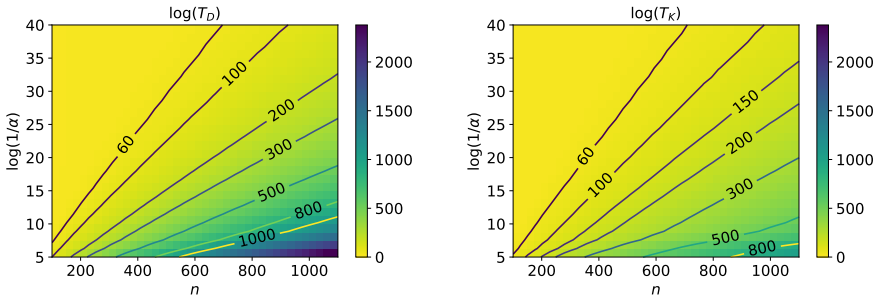


(a) dual attack (b) our attack



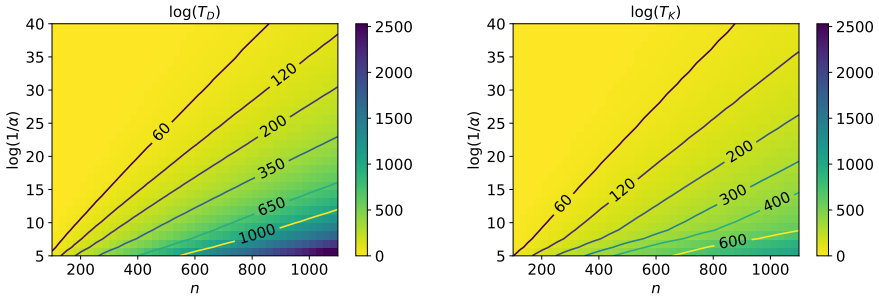
(c) difference

Fig. 6: Comparison of the costs of the attacks under the sieving BKZ cost model. Here, n and α denote the dimension and the standard deviation of the noise of LWE samples, T_D denotes the time complexity of the dual distinguishing attack, T_K denotes the time complexity of our key recovery attack.



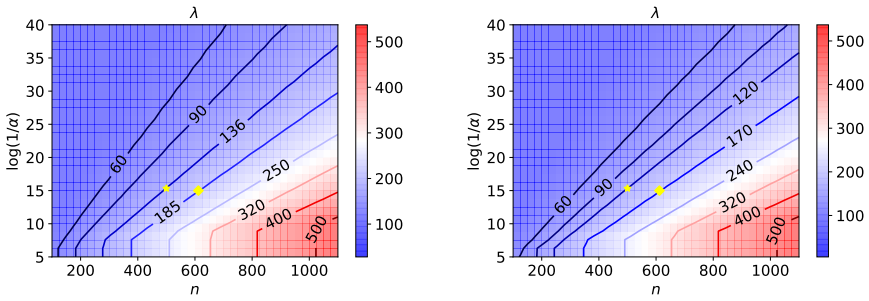
(a) dual attack (b) our attack

Fig. 7: Comparison of the costs of the attacks under the enumeration BKZ cost model.



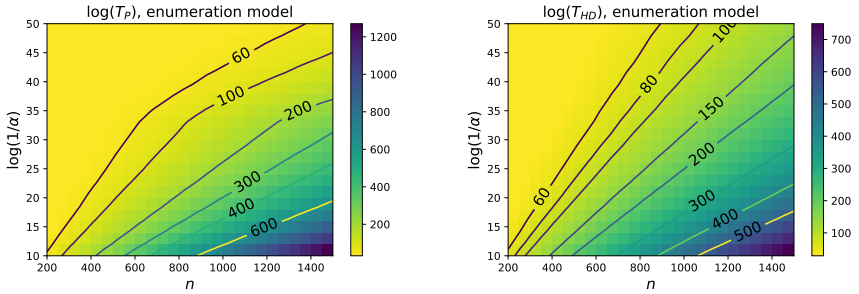
(a) dual attack (b) our attack

Fig. 8: Comparison of the costs of the attacks under the delta-squared BKZ cost model.



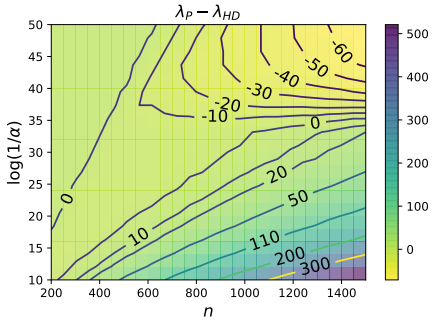
(a) enumeration (b) delta-squared

Fig. 9: Bit-security as a function of the LWE parameters n and α under the enumeration and delta-squared BKZ cost models.



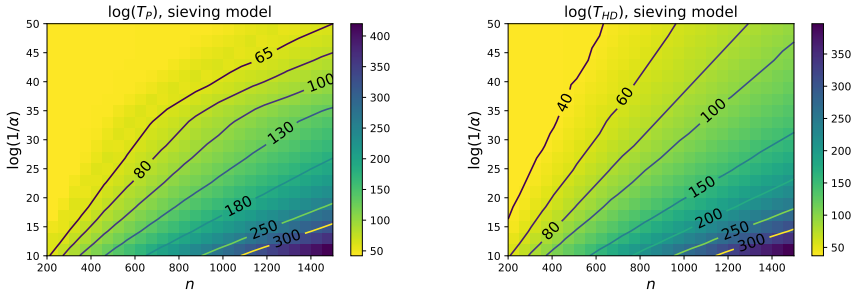
(a) primal uSVP attack

(b) our attack



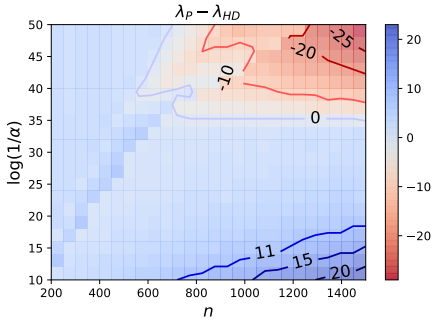
(c) difference

Fig. 10: Comparison of the costs of the hybrid dual attack and primal uSVP attack from [7] under the enumeration BKZ cost model. Here, n and α denote the dimension and the standard deviation of the noise of LWE samples, T_P denotes the time complexity of the primal uSVP attack, T_{HD} denotes the time complexity of our hybrid dual attack, $\lambda_P - \lambda_{HD} := \log(T_P) - \log(T_{HD})$.



(a) primal uSVP attack

(b) our attack



(c) difference

Fig. 11: Comparison of the costs of the hybrid dual attack and primal uSVP attack from [7] under the sieving BKZ cost model. Here, n and α denote the dimension and the standard deviation of the noise of LWE samples, T_P denotes the time complexity of the primal uSVP attack, T_{HD} denotes the time complexity of our hybrid dual attack, $\lambda_P - \lambda_{HD} := \log(T_P) - \log(T_{HD})$.