Downlink (T)FHE ciphertexts compression*

Antonina Bondarchuk[1], Olive Chakraborty[1], Geoffroy Couteau[2], and Renaud Sirdey[1]

[1] Université Paris-Saclay, CEA, List, Palaiseau, France,
name.surname@cea.fr
[2] Université Paris Cité, CNRS, IRIF, Paris, France
couteau@irif.fr

Abstract. This paper focuses on the issue of reducing the bandwidth requirement for FHE ciphertext transmission. While this issue has been extensively studied from the *uplink* viewpoint (transmission of encrypted *inputs* towards a FHE calculation) where several approaches exist to essentially cancel FHE ciphertext expansion, the *downlink* case (transmission of encrypted *results* towards an end-user) has been the object of much less attention. In this paper, we address this latter issue with a particular focus on the TFHE scheme for which we investigate a number of methods including several approaches for switching to more compact linearly homomorphic schemes, reducing the precision of T(R)LWE coefficients (while maintaining acceptable probabilities of decryption errors) and others. We also investigate how to use these methods in combination, depending on the number of FHE results to transmit. We further perform extensive experiments demonstrating that the downlink FHE ciphertext expansion factor can be practically reduced to values *below 10*, depending on the setup, with little additional computational burden.

1 Introduction

Since its inception more than ten years ago, Fully Homomorphic Encryption has been the subject of a lot of research toward more efficiency and better practicality with two main issues to be dealt with: the high computational cost of homomorphic operators and the large ciphertext expansion induced by FHE schemes. This paper focuses on the latter of these two issues which leads to the two following problems, which are very different in nature depending on whether (Fig. 1):

- Encrypted *inputs*, i.e. *freshly* encrypted ciphertexts, are transmitted towards a FHE calculation (which we refer to as the *uplink* case).
- Encrypted *results*, i.e. *evaluated* ciphertexts, obtained following some FHE calculation are transmitted towards an end-user for decryption (which we refer to as the *downlink* case).

* This work was supported by the France 2030 ANR Projects ANR-22-PECY-003 SecureCompute.

Reducing the expansion factor for the uplink case has received a lot of attention over the last ten years or so. Indeed, FHE ciphertext expansion can be almost cancelled in this case by a technique usually referred to as transciphering which simply consist in transmitting data encrypted by means of a symmetric scheme (say AES) and to homorphically turn them into FHE-encrypted data by running the symmetric scheme decryption algorithm over the FHE scheme (Fig. 2). To do so, one has to pay the FHE expansion factor only to transmit a FHE encryption of the symmetric scheme key, at setup time. As a result, many works focused on the issue of designing symmetric schemes amenable to practical homomorphic execution [2, 3, 14, 23, 28–30] or on optimizing the homomorphic execution of more standard ones [5, 6, 39]. Other approaches can also be applied to reduce that expansion factor. For instance, as all practical FHE schemes are based on (R)LWE, *in the symmetric setting* (where both encryption and decryption use the secret vector or polynomial sk), it is well known [1, 36] that one can simply synchronize the sender and the receiver on a PRF to avoid sending the a term in the (R)LWE pairs. This results in an expansion factor of $\log_2 q / \log_2 t$ where q and t respectively denote the ciphertext and plaintext moduli of the scheme. For typical TFHE parameters, this approach leads to an expansion factor of "only" 8, almost for free.

Unfortunately, techniques such as the above are not applicable to the downlink case. Indeed, the dream of being able to convert FHE encrypted results back to AES form is an ill-posed problem for several reasons, the first of which being that, as transciphering requires to homomorphically execute the decryption function of the source scheme under the target scheme, the technique applies only towards an homomorphic scheme, which is of course not the case of AES. Likewise, the above synchronization technique does not apply to the downlink case as the a term in evaluated (R)LWE pairs cannot be a priori chosen. As such, compressing FHE calculation results for downlink transmission requires completly different approaches. To the best of our knowledge, the study of this issue has been initiated in [9] which was the first paper to suggest switching from an FHE scheme to a more compact linearly homomorphic scheme. For example, considering a LWE ciphertext $(a, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, the idea simply consist in executing the dot product $b - \langle a, \mathsf{sk} \rangle$ under the LHE (given some form of encryption of sk under the latter). Then one LHE ciphertext is transferred rather than $n+1$ elements in $\mathbb{Z}_q$, achieving some compression as soon as the size of an LHE ciphertext is smaller than $(n+1)\log_2 q$. Depending on the LHE at hand, several such dot products may be packed in a single LHE ciphertext in order to further enhance compression. Using this technique, they [9] further show that for GSW (with a binary plaintext domain and very specific parameters) and the Dåmgard-Jurik scheme as the LHE, it is possible to *asymptotically* achieve "rate 1" FHE (although this result is of theoretical interest at it relies on the fact that, for that scheme, the ratio between the plaintext modulus, N^k, and the ciphertext modulus, N^{k+1}, tends to 1 as k goes to infinity). In essence, our paper build on this idea in a more practically minded fashion, by focusing mainly on the non-asymptotic regime and considering several possible candidates for the

LHE: depending on their plaintext/ciphertext size ratios, the amount of "partially decrypted" messages that can packed in their plaintexts and the conditions under which they admit an efficient decryption (as some LHE require solving a discrete log during decryption). Because of all these degrees of freedom, some LHE are more appropriate than others depending on the number of FHE ciphertexts that need to be transmitted. We also investigate how this approach can be combined with other (lossy) compression techniques based on truncation of the LWE coefficients, a simple method for which we carefully analyze the induced noise in order to determine its practical usefulness.

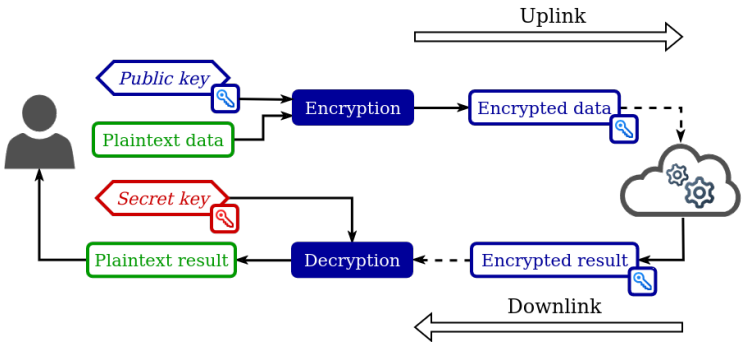

Fig. 1: *Uplink* and *downlink transmission* settings

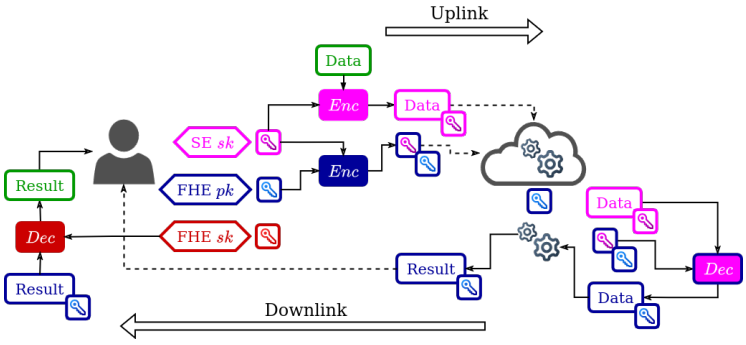

Fig. 2: Transciphering

1.1 Summary of Contributions

The contributions of this paper can be summarized as follows:

- We address the issue of compressing *evaluated* (T)FHE ciphertexts to reduce the communication burden of transferring *results* from FHE computations prior to their decryption. Contrary to prior works, we do so in the non-asymptotic regime.
- We introduce a new simple (lossy) compression technique for TLWE and TRLWE ciphertexts which consists in reducing the precision of their coefficients. We carefully analyze the resulting noise increase and show how to choose the precision loss in order to comply with a preset probability of incorrect decryption.
- We propose a new "compressed" variant of the linearly homomorphic BCP03 cryptosystem with ciphertext size reduced to $\log \mu + |m|$, where μ is the modulus of the scheme and $|m|$ denotes the bitlength of the encrypted message m. This variant, which we refer to as *compressed Paillier-ElGamal* (CPG for short) in the sequel, may also be of independent interest.
- Building on the (known) idea of executing the linear part of the decryption function for a TLWE ciphertext over a more compact LHE scheme, we investigate several candidates for the LHE (including the above) in combination with other techniques from the state-of-the-art or the present paper to achieve high compression rate of *evaluated* TFHE ciphertexts.
- We report extensive experimental results revealing the most appropriate regime for each technique (depending on parameters for TFHE as well as the number of FHE computation results that have to be transmitted).
- To the best of our knowledge, this paper is the first to demonstrate that expansion factors below 10 are practically achievable when transmitting results of FHE calculations, with limited additionnal computational burden.

1.2 Paper organization

This paper is organized as follows: Section 2 reviews the basics of TFHE (needed for understanding the paper) and gives the necessary details on the LHE that we use in this paper. Section 3 subsequently introduces known techniques that can be applied to achieve some degree of downlink compression for TLWE ciphertexts. Then, in Section 4, we present the new compression building blocks that we also propose in the paper. In, Sect. 5 we study several combinations of techniques in order to achieve high compression rates of TFHE *evaluated* ciphertexts. We then report our experimental results in Sect. 6 and conclude the paper in Sect. 7.

2 Preliminaries

2.1 General notation

In the upcoming sections, we denote vectors by bold letters, so a vector $\mathbf{x}$ of n elements is $\mathbf{x} = (x_0, \ldots, x_{n-1})$. The inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ is $\langle \mathbf{x}, \mathbf{y} \rangle$. $x \xleftarrow{\$} \mathbb{D}$ denotes sampling uniformly x from $\mathbb{D}$. $x \xleftarrow{\mathcal{N}(0,\sigma^2)} \mathbb{D}$ denotes sampling x from $\mathbb{D}$ following a Gaussian distribution of mean 0 and variance σ^2.

2.2 TFHE

The TFHE encryption scheme was proposed by Chillotti et al., in 2016 [19] and is notably implemented in the TFHE library [18]. TFHE is intrinsically a LWE scheme working over the $[0, 1)$ torus which we denote by $\mathbb{T}$. TFHE relies on three types of ciphertexts: TLWE, TRLWE and TRGSW. In this paper, we focus only on the issue of compressing TLWE and TRLWE ciphertexts as only those have to be transmitted when performing homomorphic calculations. TRGSW ciphertexts are only used temporarily within the bootstrapping procedure and never transmitted (except, offline, for transfering the bootstrapping key).

- TLWE ciphertext: a pair $(\tilde{\mathbf{a}}, \tilde{b})$ is a valid TLWE encryption of $m \in \mathbb{Z}_t$ (for plaintext modulus t), with $\tilde{\mathbf{a}} \xleftarrow{\$} \mathbb{T}^n$ and $\tilde{b} \in \mathbb{T}$ if it verifies $\tilde{b} = \langle \tilde{\mathbf{a}}, \mathbf{s} \rangle + \frac{m}{t} + \tilde{e}$, where $\mathbf{s} \xleftarrow{\$} \mathbb{B}^n$ is a TLWE secret key, and $\tilde{e} \xleftarrow{\mathcal{N}(0,\sigma^2)} \mathbb{T}$ is a noise term.

- TRLWE ciphertext: a pair $(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$ is a valid TRLWE encryption of $\mathbf{m} \in \mathbb{Z}_t[X]/(X^N+1)$, with $\tilde{\mathbf{a}} \xleftarrow{\$} \mathbb{T}_N[X]$ and $\tilde{\mathbf{b}} \in \mathbb{T}_N[X]$ if it verifies $\tilde{\mathbf{b}} = \tilde{\mathbf{a}} \cdot \mathbf{s} + \frac{\mathbf{m}}{t} + \tilde{\mathbf{e}}$, where $\mathbf{s} \xleftarrow{\$} \mathbb{B}_N[X]$ is a TRLWE secret key, and $\tilde{\mathbf{e}} \xleftarrow{\mathcal{N}(0,\sigma^2)} \mathbb{T}_N[X]$ is a noise polynomial. Here polynomials are represented by vectors of its coefficients.

Please note that, as this paper focuses on the input/output of FHE calculations, only a high-level understanding of the inner working of the TFHE scheme and in particular its bootstrapping procedure is required to understand this work. We refer the reader to [19] for further details.

It should further be emphasized that TFHE is fully homomorphic only over TLWE ciphertexts. Up to N TLWE ciphertexts can be packed to/unpacked from a single TRLWE ciphertext using standard techniques introduced in [10, 19, 35]. This may be useful for improving transmission efficiency (as we shall later discuss) or to perform batched homomorphic additions.

As an LWE-based scheme, TFHE decryption function is decomposed into a first linear part

$$\langle (\tilde{b}; -\tilde{\mathbf{a}}), (1; \mathbf{s}) \rangle = \tilde{\varphi} = \frac{m}{t} + \tilde{\mathbf{e}} \tag{1}$$

followed by a scale (up) and round operation

$$\lceil t\tilde{\varphi} \rfloor = m. \tag{2}$$

In TFHE terminology, $\tilde{\varphi}$ is called the *phase* or the *partial decryption* of the ciphertext $(\tilde{\mathbf{a}}; \tilde{b})$. One key observation [9] is that, given some encryption of $\mathbf{s}$ (with the appropriate form) over another LHE, (1) can be executed over the latter to get an encryption of φ under that LHE. How and when this is practically useful is investigated in the sequel.

Equivalently, the scheme is always defined relatively to a discretization of the torus by steps of $\frac{1}{q}$ where q is a power of two (typically either 2^{32} or 2^{64}). As such, TLWE or TRLWE ciphertexts are equivalently represented as LWE or

RLWE ciphertexts over $\mathbb{Z}_q$ or $\mathbb{Z}_q[X]/(X^N+1)$. When this representation is used, we will use the notations $(\mathbf{a}, b)$ and $(\mathbf{a}, \mathbf{b})$ instead of $(\tilde{\mathbf{a}}, \tilde{b})$ and $(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$ (with e.g. $\mathbf{a} = \lceil q\tilde{\mathbf{a}} \rfloor$ and similarly for the others). Note also, that the phase computation (1) over $(\mathbf{a}, b)$ returns $\varphi = \Delta m + e$ with $\Delta = \frac{q}{t}$ and decryption is finalized by outputting

$$\left\lceil \frac{\varphi}{\Delta} \right\rfloor \tag{3}$$

instead of Eq. (2).

We will use the illustrative parameters in Table 1 as a running example throughout the paper. This will allow us to provide illustrative numbers before the reader reaches Sect. 6 on experimental results where several sets of TFHE parameters are investigated.

λ	n	q	t	N	σ_0	σ_{BS}
128	550	2^{32}	2	1024	$q \cdot 2.82 \times 10^{-4}$	$q \cdot 1.69 \times 10^{-2}$

Table 1: Illustrative TFHE parameters used as a running example throughout the paper. More exhaustive parameter sets are investigated in Sect. 6.

2.3 Linear Homomorphic Encryption schemes

In this section, we provide the necessary background on the candidate LHE schemes we consider in this paper. As hinted in the previous Sect. we will consider using these schemes to evaluate the linear part of the TFHE decryption function therefore converting one (or possibly more) ciphertexts of size $(n+1)\log_2 q$ into a single ciphertext of the LHE scheme. Here we briefly recall the Paillier [34], Dåmgard-Jurik [24], Elliptic Curve ElGamal [31] and BCP03 [13] cryptosystems which are used in the sequel.

Paillier. The Paillier cryptosystem [34] is a public key encryption scheme invented by Pascal Paillier in 1999. It is a partial homomorphic scheme that notably allows to perform additions and multiplications-by-a-constant directly over its ciphertexts

- KeyGen: Choose two large prime numbers p and q randomly and independently such that $\gcd(pq, (p-1)(q-1)) = 1$. Compute $\mu = pq$, $\lambda = \varphi(\mu)$ and set $g = \mu + 1$, where $g \in \mathbb{Z}^*_{\mu^2}$, $\eta = \varphi(\mu)^{-1} \bmod \mu$. Return a public key $pk = (\mu, g)$ and a secret key $sk = (\lambda, \eta)$.

- Enc: Let m, $0 \leq m < \mu$, be a message to encrypt. Select a random $r : 0 < r < \mu$ and $\gcd(r, \mu) = 1$. Return a ciphertext $c \in \mathbb{Z}^*_{\mu^2}$ as $c = g^m \cdot r^\mu \bmod \mu^2$.

- Dec: Return $m = L(c^\lambda \bmod \mu^2) \cdot \eta \bmod \mu$, where $L(x) = \frac{x-1}{\mu}$.

In summary, for the Paillier scheme, plaintext domain is $\mathbb{Z}_\mu$ and ciphertext domain is $\mathbb{Z}_{\mu^2}$. A typical parameter choice to achieve 128-bit security or slightly

above consists in taking a modulus μ on 2048 bits. A Paillier ciphertext is therefore of size 4096 bits with a 2048 bits payload (leading to an expansion factor of 2).

Dåmgard-Jurik. Dåmgard-Jurik cryptosystem [24] is a generalization of the Paillier cryptosystem. It uses a plaintext modulus μ^y and a ciphertext modulus μ^{y+1}, where μ is an RSA modulus and $y \in \mathbb{Z}^+$. Paillier cryptosystem is the special case with $y = 1$.

- KeyGen: Choose an admissible RSA modulus $\mu = pq$ and compute $\lambda = \text{lcm}\,((p-1)(q-1))$. Return a public key $pk = (\mu)$ and a secret key $sk = (\lambda)$.

- Enc: Let an integer $m > 0$ be a message to encrypt. Choose $y : m < \mu^y$ and select a random $r \in \mathbb{Z}_\mu^*$. Return a ciphertext $c = r^{\mu^y}(1+\mu)^m \bmod \mu^{y+1}$.

- Dec: Compute $c^\lambda \bmod \mu^{y+1} = (1+\mu)^{m\lambda} \bmod \mu^{y+1}$. Compute $v = m\lambda \bmod n^y$ (apply an algorithm from Theorem 1 [24]). Return $m = v\lambda^{-1} \bmod n^y$.

In summary, for the Dåmgard-Jurik scheme, plaintext domain is $\mathbb{Z}_{\mu^y}$ and ciphertext domain is $\mathbb{Z}_{\mu^{y+1}}$. A typical parameter choice to achieve 128-bit security or slightly above consists in taking a modulus μ on 2048 bits. A Dåmgard-Jurik ciphertext is therefore of size $2048(y+1)$ bits with a $2048y$ bits payload. Note that, the choice of y has no impact on security. Interestingly, the expansion factor for the Dåmgard-Jurik scheme is asymptotically such that

$$\lim_{y\to\infty} \frac{2048(y+1)}{2048y} = 1.$$

Elliptic Curve ElGamal. Elliptic Curve ElGamal [31] is a public key additive homomorphic encryption scheme. It is a generalization of ElGamal public key cryptosystem over elliptic curve.

- KeyGen: Choose a large prime ω and an elliptic curve $E(\mathbb{F}_\omega)$. Choose a point $P \in E$, an integer $a < \text{ord}(P)$ and compute $Q = aP$. Return a public key $pk = (E, P, Q)$ and a secret key is $sk = (a)$.

- Enc: Let m, $0 \leq m < \omega$ be a message to encrypt. Express m as a point $X \in E$: $X = mP$. Choose random r and return the ciphertext $\mathbf{c} = (c_0, c_1)$ as $c_0 = rP,\ c_1 = X + rQ$.

- Dec: Compute $X = c_1 - ac_0$. Solve $X = mP$ and return m.

In summary, to setup the scheme, we choose an ellictic curve $y^2 = x^3 + ax + b \bmod \omega$ then a plaintext domain is $\mathbb{F}_\omega$ and a ciphertext domain is $\mathbb{F}_\omega^2$. A typical parameter choice to achieve 128-bit security or slightly above consists in taking an elliptic curve Curve25519. An EC-ElGamal ciphertext is therefore of

size 510 bits with a 255 bits payload. However, let us also emphasize that the decryption function requires solving an elliptic curve discrete logarithm problem over Curve25519 which is practical only when a small upper bound is known for the decrypted message. This means that only $\mathfrak{p} \leq 50$ bits are truly usable from the plaintext domain.

BCP03. BCP03 [13] is a another public key additive homomorphic cryptosystem, that is an ElGamal-style variant of the Paillier scheme. The scheme composes of three algorithms defined below.

- KeyGen: Let $\mu = pq$ be an RSA modulus. Choose a random $\alpha \in \mathbb{Z}^*_{\mu^2}$, a random value $d \in [1, \mathrm{ord}(\mathbb{G})]$. Set $g = \alpha^2 \bmod \mu^2$ and $h = g^d \bmod \mu^2$. Return a public key $pk = (\mu, g, h)$ and a secret key $sk = (d)$.

- Enc: For a given message $m \in \mathbb{Z}_\mu$, a random pad $r \xleftarrow{\$} \mathbb{Z}_{\mu^2}$ return a ciphertext $\mathbf{c} = (c_0, c_1)$ such that $c_0 = g^r \bmod \mu^2,\ c_1 = h^r(1+\mu)^m \bmod \mu^2$.

- Dec: Compute $c = c_1(c_0)^{-d} \bmod \mu^2$ and return $m = \frac{c-1}{\mu}$.

In summary, for the *vanilla* variant of this scheme, a plaintext domain is $\mathbb{Z}_\mu$ and a ciphertext domain is $\mathbb{Z}^2_{\mu^2}$. A typical parameter choice to achieve 128-bit security or slightly above consists in taking a modulus μ on around 2048 bits. A BCP03 ciphertext is therefore of size 8192 bits with a 2048 bits payload (leading to an expansion factor of 4). As such this scheme might not appear competitive with the other LHE presented in this section, we will however propose in Sect. 5.4 a more advanced variant, which we will refer to as *compressed Paillier-ElGamal* (CPG), in which c_0 is compressed onto 2048 bits and c_1 onto $\log_2 U$ bits (where U is an upper bound on the encrypted message). As such, this new variant will be much competitive.

Cryptosystem	Plaintext domain	Ciphertext domain	Plaintext size (bits)	Ciphertext size (bits)	Expansion factor
Paillier	$\mathbb{Z}_\mu$	$\mathbb{Z}_{\mu^2}$	$\log_2 \mu$	$2\log_2 \mu$	2
Dåmgard-Jurik	$\mathbb{Z}_{\mu^y}$	$\mathbb{Z}_{\mu^{y+1}}$	$y \log_2 \mu$	$(y+1)\log_2 \mu$	$1 + \frac{1}{y}$
EC ElGamal	$\mathbb{F}_\omega$	$\mathbb{F}^2_\omega$	$\mathfrak{p}$	$2\log_2 \omega$	$\frac{2\log_2 \omega}{\mathfrak{p}}$
BCP03	$\mathbb{Z}_\mu$	$\mathbb{Z}^2_{\mu^2}$	$\log_2 \mu$	$4\log_2 \mu$	4
CPG (Sect. 5.4)	$\mathbb{Z}_\mu$	$\mathbb{Z}^2_{\mu^2}$	$\log_2 \mu$	$\log_2 \mu + \log_2 U$	$1 + \frac{\log_2 U}{\log_2 \mu}$

Table 2: Summary of the main characteristics of the LHE schemes presented in Sect. 2.3.

3 Existing *downlink* compression techniques for FHE

3.1 Positionning with respect to [9]

The idea to switch from FHE scheme with linear decryption to LHE scheme with a smaller expansion factor to compress FHE data is introduced in [9]. Also, as the present paper, that paper is focused on techniques to compress post-evaluation FHE ciphertexts (i.e., in the *downlink* case) and does not discuss compression techniques for the *uplink* case. In that setting, it introduces a general approach to build a rate-1 FHE by switching from GSW-style schemes [4, 12, 26, 32] to the Dåmgard-Jurik cryptosystem or to a linearly homomorphic scheme with a packed Regev encryption that supports ciphertext shrinking.

To switch n ciphertexts from GSW style FHE to Dåmgard-Jurik, the authors [9] define a "compression key" ck, which is an encryption of the FHE secret key $\mathbf{s}$ under the LHE scheme, and a packed linear decrypt-and-multiply function L^* (in a matrix form) for packed n GSW style ciphertexts. The idea then is to evaluate L^* using ck and get an encryption of $\sum_{i=1}^{n} 2^{t+1} \cdot m_i + e_i$ ($2^t > nB$, where B is a noise bound), i.e. the packed n messages under the Dåmgard-Jurik scheme. The authors claim that a rate-1 expansion factor is achievable asymptotically (for very specific GSW parameters), as for the Dåmgard-Jurik cryptosystem the rate $\frac{\log(\mu^y)}{\log(\mu^{y+1})} = 1 - \frac{1}{y+1}$ approaches 1 as y grows. In this case, however the asymptotic growth of y also increases the computational cost for running the scheme.

As in our work we are focused on TFHE ciphertexts compression, we cannot use the proposed switching algorithm from [9] for GSW style schemes. We introduce a $\mathsf{TFHEtoLHE}$ algorithm (Alg. 2) to pack several TLWE ciphertexts in a LHE one (Section 5). In Alg. 2 we use an encryption of the TFHE secret key under the target LHE and define, how to evaluate a linear part of TFHE decryption to get a resulting ciphertext encrypted under the packed target LHE. In our approach, we use a slot-by-slot packing approach i.e., Alg. 2 returns an LHE encryption of $m_0 + e_0 || \ldots || m_{n-1} + e_{n-1}$. In Sect. 5 we carefully define a slot size for TFHE ciphertexts packing. Additionally, we show that it is possible to reduce the slot size by applying a lossy compression technique (Sect. 4.1) to TFHE ciphertexts before switching to the target LHE. In our work we do not limit ourselves to only Dåmgard-Jurik cryptosystem as a candidate LHE, but use several other LHE schemes such as Paillier, EC ElGamal and a new compressed variant BCP03. We provide clear dimensioning details about the maximum number of TFHE ciphertexts it is possible to switch and resulting expansion factors for all the listed LHE schemes. We fix standard ciphertext space modulus for each LHE. If we need to switch more TFHE ciphertexts than can be switched into a single LHE ciphertext with a given ciphertext space modulus, we assume switching to several LHE ciphertexts (i.e. we do not increase the ciphertext space modulus to switch all TFHE ciphertexts into a single LHE ciphertext).

3.2 TLWE packing to one TRLWE

In this Section we discuss the *well-known* idea of packing (up to) $N > n$ TLWE samples into a single TRLWE sample in order to amortize the $\mathbf{a}$ vectors of

TLWE samples. Doing that we would need to transmit just one $\mathbf{a}$ vector for N TLWE samples rather then one $\mathbf{a}$ vector for each of N TLWE samples. This TLWE samples packing technique is called TFHE Public Functional Key Switching, which was introduced in [19]. We use the TFHE Public Functional Key Switching with a function being an identity function (Alg. 1), which allows to pack N TLWE samples by means of the $\mathbb{Z}-$module isomorphism into a single TRLWE sample whereby N TLWE messages $m_0, \ldots, m_{N-1} \in \mathbb{T} \mapsto m(X) = \sum_{i=0}^{N-1} m_i X^i \in \mathbb{T}_N[X]$.

Algorithm 1 TLWEtoTRLWE [19]

Input: p TLWE ciphertexts $\mathbf{c}^{(z)} = (\mathbf{a}^{(z)}, b^{(z)}) \in \mathrm{TLWE}_{\mathbf{s}}(m_z)$, for $z = 0, \ldots, p-1$ and $\mathrm{KS}_{i,j} \in \mathrm{TRLWE}_{\mathbf{S}}\left(s_i / B_{\mathrm{KS}}^{j}\right)$.
Output: a TRLWE sample $\mathbf{C} \in \mathrm{TRLWE}_{\mathbf{S}}(m_0, \ldots, m_{p-1})$.
1: **for** $i \in [0, n-1]$ **do**
2: Let $a_i = a_i^{(0)} + a_i^{(1)} X + \cdots + a_i^{(p-1)} X^{p-1}$
3: Let $\hat{a}_i$ be the closest multiple of $1/B_{\mathrm{KS}}^{t_{\mathrm{d}}}$ to a_i, thus $\|\hat{a}_i - a_i\|_\infty < B_{\mathrm{KS}}^{-(t_{\mathrm{d}}+1)}$
4: Let $\hat{a}_i = \sum_{j=1}^{t_{\mathrm{d}}} \hat{a}_{i,j} \cdot B_{\mathrm{KS}}^{-j}$, where $\hat{a}_{i,j} \in \mathbb{B}_N[X]$
5: **end for**
6: **return** $(0, \ldots, b^{(0)} + b^{(1)} X + \cdots + b^{(p-1)} X^{p-1}) - \sum_{i=0}^{n-1} \sum_{j=1}^{t_{\mathrm{d}}} \hat{a}_{i,j} \cdot \mathrm{KS}_{i,j}$

Noise Analysis: Because additional noise increases the probability of erroneous decryption, it is important to characterize the noise variance σ^2_{pack} in a TRLWE sample obtained from TLWEtoTRLWE packing of N TLWE samples with *independant* noises of variance bounded by σ^2_{BS}. From [19], the error variance σ^2_{pack} satisfies:

$$\sigma^2_{\mathrm{pack}} \leq R^2 \sigma^2_{\mathrm{BS}} + n t_{\mathrm{d}} N \sigma^2_{\mathrm{TRLWE}} + \frac{n}{12} B_{\mathrm{KS}}^{-2t_{\mathrm{d}}}, \tag{4}$$

where $R = 1$[3] and $\sigma^2_{\mathrm{TRLWE}}$[4] is the variance of the error in the Key-switching key (KS).

If $K < N$ TLWE samples have to be transmitted, it is possible to pack just $K < N$ TLWE samples into a single TRLWE sample assuming a *not-full packing* to avoid transmiting a part of resulting TRLWE sample coefficients. It means that we "fill" the first K slots of TRLWE with TLWEs and keep $N - K$ slots empty. To transmit a not-fully packed TRLWE sample on the downlink, we transmit $\mathbf{a}$ and the first K coefficients of $\mathbf{b}$: $b_0, \ldots, b_{K-1}$. To decrypt this TRLWE sample, we perform a decryption of the first K slots of TRLWE:

[3] We use the identity function f as a public R-Lipschitz morphism, thus $R = 1$.
[4] Note that the keyswitch key KS is a fresh TRLWE sample hence, the variance of the error in KS is $\sigma^2_{\mathrm{TRLWE}}$.

$$\begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{K-1} \end{pmatrix} - \begin{pmatrix} a_0 & -a_{N-1} & \dots & -a_1 \\ a_1 & a_0 & \dots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{K-1} & a_{K-2} & \dots & -a_K \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{N-1} \end{pmatrix} = \begin{pmatrix} m_0 + e_0 \\ m_1 + e_1 \\ \vdots \\ m_{K-1} + e_{K-1} \end{pmatrix}$$

Now, let's compute the expansion factor for TLWEtoTRLWE packing in the downlink case. Let a value $r_2 = \lfloor K/N \rfloor$ defines how many fully packed TRLWE samples we get after packing K TLWE samples, and a value $r_1 = K \bmod N$ defines how many TLWE samples are left to pack in a last not-fully packed TRLWE sample. It means that we need to transmit r_2 fully packed TRLWE samples and one not-fully packed TRLWE sample with r_1 (> 0) packed coefficients, then the size of the resulting ciphertexts is $2r_2 N \log_2 q + (N + r_1) \log_2 q$. At the same time, the resulting ciphertexts encrypt $r_2 N + r_1$ plaintext messages of size $\log_2 t$; thus, for $r_1 > 0$ the expansion factor is

$$\frac{(2r_2 N + N + r_1) \log_2 q}{(r_2 N + r_1) \log_2 t}. \tag{5}$$

and for $r_1 = 0$

$$\frac{2 \log_2 q}{\log_2 t}. \tag{6}$$

Then, for example, for $t = 2$ and $K = N = 1024$ the expansion factor is 64.

3.3 Shrinking

The, so-called shrinking technique, first proposed in [9], allows for compressing ciphertexts from a FHE scheme that supports linear decrypt-and-multiply, like the GSW scheme [27]. It can also be applied directly to compress BFV-style ciphertexts as well as TRLWE ones. In a nutshell, given a TRLWE sample $(\mathbf{a}, \mathbf{b})$, shrinking consists of computing two values, a 'helper' $r \in \mathbb{Z}_q$ and a value $w \in \mathbb{Z}_t$ such that the decryption of the original ciphertext (which is not necessarily correct) can be recovered *exactly*[5] from r and w (as well as, of course, the knowledge of the secret key). Shrinking is interesting because a single helper value 'r' can be used to cover the N LWE samples assembled in a RLWE ciphertext.

In a nutshell, shrinking works as follows. Let $\mathbf{c} = (\mathbf{a}, \mathbf{b})$ be a TRLWE sample we need to compress and let $\mathbf{s}$ be a TRLWE secret key. To shrink the TRLWE sample we parse $\mathbf{b} = (b_0, \dots, b_{N-1}) \in \mathbb{Z}_q^N$ and compute the union of intervals $U \subseteq \mathbb{Z}_q$, where $B = \sigma_{\text{TLWE}}$ is a TLWE noise bound.

$$U = \bigcup_{i=0}^{N-1} \left(\left[\frac{\Delta}{2} - b_i - B, \frac{\Delta}{2} - b_i + B \right] \cup \left[-\frac{\Delta}{2} - b_i - B, -\frac{\Delta}{2} - b_i + B \right] \right)$$

[5] This is important for BFV-style schemes which tend to induce large noise variances in evaluated ciphertexts.

where $\Delta = q/t$. Then we pick any $r \in \mathbb{Z}_q \setminus U$ and for $i = [0, N-1]$ compute $w_i = \lceil b_i + r \rfloor_t$, where $\lceil x \rfloor_t = \left\lfloor x \cdot \frac{t}{q} \right\rfloor \mod t$ is a rounding function. The resulting shrunk sample is $\tilde{\mathbf{c}} = (r, \mathbf{a}, w_0, \ldots, w_{N-1})$. This therefore leads to an overhead of

$$\frac{(N+1)\log_2 q + N \log_2 t}{N \log_2 t} \tag{7}$$

i.e., for $N = 1024$ ($t = 2$) it is ≈ 33 or for $N = 2048$ ($t = 16$) ≈ 9. If only $K < N$ LWE samples are to be transmitted (we assume not-full packing for TRLWE), then, trivially from (7), the expansion factor is

$$\frac{(N+1)\log_2 q + K \log_2 t}{K \log_2 t}.$$

To decrypt the shrunk TRLWE sample on the downlink, we do the following: we compute $\mathbf{v} = \mathbf{s} \cdot \mathbf{a}$ and parse $\mathbf{v} = (v_0, \ldots, v_{N-1})$. For $i \in [0, N-1]$ we then compute $m'_i = (w_i - \lceil v_i \rfloor_t) \mod t$ and output $m' = (m'_0, \ldots, m'_{N-1})$. The equivalence between this decryption function and the original one then follows from Lemma 1 of [9]. For more details, we refer the reader to Appendix A.1.

Let us emphasize however that when used for compressing several TLWE ciphertexts, Shrinking affects the probability of erroneous decryption as we can apply Shrinking only to TRLWE ciphertexts. As discussed in the previous Sect. packing several TLWE increases the noise deviation.

4 New compression building-blocks for evaluated TFHE ciphertexts

4.1 ℓ-truncation

Now we turn our attention to a basic (lossy) compression technique which consists in dropping least significant bits in LWE or RLWE pairs coefficients. The main question is whether the additional noise that it finally induces may be small enough to allow significant compression without prohibitively increasing the probability for decryption errors to occur. The goal is then to carefully choose the number of discarded bits in order to comply with a preset probability of incorrect decryption, e.g., $\epsilon = 2^{-k}$ for $k = 40$, 64 or 128.

On one hand, TLWE ℓ-truncation can be combined with the LHE switching technique of Sect. 5 to increase the number of TLWE partial decryptions that can be packed into a single LHE ciphertext. On the other hand, TRLWE ℓ-truncation is useful as a stand-alone method: we can apply it after performing TLWEtoTRLWE packing (Sect. 3.2) to reduce the size of the coefficient of a TRLWE sample before transmission[6].

[6] It is worth mention that RLWE ℓ-truncation technique can be applied to BGV [11], BFV [7, 25] and CKKS [17] RLWE ciphertexts. As these FHE cryptosystems use composite ciphertext modulus, it is natural to think of reducing it via rescaling before applying ℓ-truncation and sending the final result to the client. We leave this idea as a perspective for future explorations.

We first study how ℓ-truncation may be applied to a TLWE ciphertext.

4.2 TLWE ℓ-truncation

Let $\mathbf{c} = (a_0, \ldots, a_{n-1}, b = a_n)$ denotes a TLWE encryption of m. Given $\ell < \lceil \log_2(q) \rceil$, we define the following three operations:

- $\mathsf{PartialDec}(\mathbf{c}, \mathbf{s})$: return $a_n - \langle \mathbf{a}, \mathbf{s} \rangle = \Delta m + e$, with $\Delta = \frac{q}{t}$.
- $\mathsf{Trunc}(\mathbf{c}, \ell)$: compute $a'_i = \left\lfloor \frac{a_i}{2^\ell} \right\rceil$ for $i \in \{0, ..., n\}$ and return $\mathbf{c}' = (a'_0, \ldots, a'_n)$.
- $\mathsf{Rescale}(\mathbf{c}', \ell)$: compute $a''_i = 2^\ell a'_i$, for $i \in \{0, ..., n\}$ and return $\mathbf{c}'' = (a''_0, \ldots, a''_n)$.

From these definitions, it follows that when $\mathbf{c}$ is a TLWE encryption of m with noise e (i.e. $\mathsf{PartialDec}(\mathbf{c}, \mathbf{s}) = \Delta m + e$), then $\mathbf{c}''$ is an encryption of m with noise

$$e'' = e - \sum_{i=0}^{n-1} e''_i s_i + e''_n \tag{8}$$

where $e''_i = -(a_i \mod 2^\ell)$.

Now, given a preset probability of decryption error $\epsilon = 2^{-k}$ and a noise variance σ for e, we would like to be able to choose ℓ such that an ℓ-truncated ciphertext decrypts correctly with probability at least $1 - \epsilon$, i.e. following Eq. (8)

$$\Pr\left(|e''| < \frac{\Delta}{2} \right) \geq 1 - \epsilon.$$

We then have the following proposition which provides us with a first lower bound for the choice of ℓ.

Proposition 1. *Let $\mathbf{c}$ denote a TLWE encryption of m subject to a centered Gaussian noise e with variance σ^2 and let*

$$\mathbf{c}' = \mathit{Trunc}(\mathbf{c}, \ell_0)$$

with

$$\ell_0 \leq \left\lfloor \log_2 \left(\frac{1}{n+1} \left(\frac{\Delta}{2} - \sigma \sqrt{2k \log 2} \right) + 1 \right) \right\rfloor. \tag{9}$$

Then, $\left\lceil \frac{1}{\Delta} \mathit{PartialDec}(\mathit{Rescale}(\mathbf{c}', \ell_0), \mathbf{s})) \right\rfloor = m$ with probability at least $1 - 2^{-k}$.

Proof. Let us start by bounding the probability that $\mathbf{c}'' = \mathsf{Rescale}(\mathbf{c}', \ell)$ incorrectly decrypts, i.e., following (8), that

$$\begin{aligned} \Pr\left(|e''| \geq \frac{\Delta}{2} \right) &= \Pr\left(\left| e - \sum_{i=0}^{n-1} e''_i s_i + e''_n \right| \geq \frac{\Delta}{2} \right), \\ &\leq \Pr\left(|e| \geq \frac{\Delta}{2} - (n+1)(2^\ell - 1) \right), \end{aligned}$$

as, by definition, $|e_i''| \leq 2^\ell - 1$. Assuming e follows a (centered) Gaussian distribution of variance σ^2, the Chernoff bound[7] tells us that

$$\Pr\left(|e''| \geq \frac{\Delta}{2}\right) \leq e^{-\frac{\left(\frac{\Delta}{2} - (n+1)(2^\ell - 1)\right)^2}{2\sigma^2}}.$$

Which, letting

$$e^{-\frac{\left(\frac{\Delta}{2} - (n+1)(2^\ell - 1)\right)^2}{2\sigma^2}} = 2^{-k},$$

leads to,

$$2^\ell = \frac{1}{n+1}\left(\frac{\Delta}{2} - \sigma\sqrt{2k\log 2}\right) + 1,$$

hence the claim. □

In general, we use this proposition for evaluated ciphertext taking σ^2 as the post-bootstrapping variance σ^2_{BS}. To give an intuition, with our running TFHE parameter example (Table 1), for $\epsilon = 2^{-40}$, Prop. 1 leads to $\ell_0 = 19$ meaning a ciphertext size reduction of around 60%.

Although it does not gives us a closed-form formula for ℓ, the following proposition is a bit more precise than Prop. 1.

Proposition 2. *Let* $\mathbf{c}$ *denote a TLWE encryption of* m *subject to a centered Gaussian noise* e *with variance* σ^2 *and let*

$$\mathbf{c}' = \mathsf{Trunc}(\mathbf{c}, \ell)$$

then $\left\lceil \frac{1}{\Delta}\mathsf{PartialDec}(\mathsf{Rescale}(\mathbf{c}', \ell_0), \mathbf{s}))\right\rfloor = m$ *with probability at least*

$$1 - \exp\left(-\frac{\left(\frac{\Delta}{2} - \frac{1}{2}(n-1)(2^\ell - 1)\right)^2}{2\left(\sigma^2 + \frac{1}{12}(n+1)(2^{2\ell} - 1)\right)}\right).$$

Proof. Recall Eq. (8), we have

$$e'' = e + \sum_{i=0}^{n-1} \underbrace{(a_i \mod 2^\ell)}_{-e_i''} s_i - \underbrace{(a_n \mod 2^\ell)}_{-e_n''}.$$

Under the assumption than q is a power of 2, $a_i \bmod 2^\ell$ ($i \in \{0, ..., n-1\}$) is uniformly distributed over $\{0, ..., 2^\ell - 1\}$ as the associated a_i's are uniformly distributed in $\mathbb{Z}_q$. Additionally, $a_n \bmod 2^\ell$ is also uniformly distributed over $\{0, ..., 2^\ell - 1\}$ as, following the LWE assumption, $b = a_n$ is indistinguishable from a uniform deviate over $\mathbb{Z}_q$. Let $n' = \sum_i s_i$, from the CLT, we can thus assume that e'' follows a Gaussian distribution with expectation

$$E[e''] = \frac{1}{2}(n'-1)(2^\ell - 1) \leq \frac{1}{2}(n-1)(2^\ell - 1),$$

[7] Recall that for a (centered) Gaussian deviates, the Chernoff bound is such that $P(|X| \geq \alpha) \leq e^{-\frac{\alpha^2}{2\sigma^2}}$.

and variance[8]

$$V[e''] = \sigma^2 + \frac{1}{12}(n'+1)(2^{2\ell}-1) \leq \sigma^2 + \frac{1}{12}(n+1)(2^{2\ell}-1).$$

The claim follows from applying the Chernoff bound[9] to $\Pr\left(|e''| \geq \frac{\Delta}{2}\right)$. □

Using the latter Proposition in conjunction with Prop. 1 usually allows to slightly increase the number of bits that may be dropped off. For example, as discussed just above, with our running TFHE parameter set example, Prop. 1 tells that for $\epsilon = 2^{-40}$, $l_0 = 19$. Prop. 2 however tells us that the probability of decryption error is bounded by $2^{2^{-117.50}}$ for that value (part of that gap is explained by the ceiling that occur in (9), as we can only drop a integer number of bits). Then, if we choose $\ell = 20$, the bound drops to $2^{-83.21}$, still above our 2^{-40} target. As this is the cutoff value, we can finally settle on $\ell = 20$ meaning a ciphertext size reduction of 62.5%. Overall, the expansion factor goes from 8816 down to 3306, i.e. is reduced by a factor of 2.66.

As we shall later see, ℓ-truncation can be further combined with other methods to improve their compression rate. This is so because the phase computation (1) can be performed over $\mathbf{c}'$ rather than $\mathbf{c}''$ and as such, less bit is required to represent the phase of $\mathbf{c}'$ than that of $\mathbf{c}$ (still with some control kept on the probability of erroneous decryption).

4.3 TRLWE ℓ-truncation

ℓ-truncation can equally be applied to TRLWE ciphertext, by dropping ℓ least bits on all the coefficients of the $\mathbf{b}$ polynomial. Because, this is very similar to the TLWE case, we do not provide further details. For TRLWE, we also have the analogous of Proposition 1 with Eq. (9) replaced by (recall from Sect. 3.2 that, by default, we pack N n-dimensional TLWE samples in a single degree-N TRLWE)

$$\ell_0 \leq \left\lfloor \log_2 \left(\frac{1}{n+1} \left(\frac{\Delta}{2} - \sqrt{-2\sigma^2 \log(1 - \sqrt[N]{1-2^{-k}})} \right) + 1 \right) \right\rfloor. \qquad (10)$$

which provides the guarantees that

$$\Pr\left(||\mathbf{e}''||_\infty \geq \frac{\Delta}{2} \right) \leq 2^{-k}.$$

However, let us emphasize that the above equation bounds the probability that an ℓ_0-truncated TRLWE ciphertext decrypts incorrectly meaning that *a decryption error occurs in at least one slot of the message polynomial*. Since the present

[8] Recall that the variance of the discrete uniform distribution over $\{a, ..., b\}$ is $\frac{1}{12}((b-a+1)^2-1)$, leading to $\frac{1}{12}(2^{2l}-1)$ when $a=0$ and $b=2^\ell-1$.

[9] For a Gaussian deviate of expectation $\mu \geq 0$ and variance σ^2, it holds that $P(|X| \geq \alpha) \leq e^{-\frac{(\alpha-\mu)^2}{2\sigma^2}}$. Furthermore, for $\mu' \geq \mu$ and $\sigma' \geq \sigma$, $e^{-\frac{(\alpha-\mu)^2}{2\sigma^2}} \leq e^{-\frac{(\alpha-\mu')^2}{2\sigma'^2}}$.

work is focusing on TFHE and is therefore TLWE-centric, we are only using TRLWE ciphertexts as a mean to more efficiently transmit TLWE ciphertexts and, in fine, it is the decryption error probability of these TLWE ciphertexts which is important to us. So despite of the fact that we may use TRLWE ciphertexts for transmission, we stick to the tools provided in the previous section in the sequel.

However, in using Prop. 1 and 2 to choose the number of bits that can be dropped from coefficients of a TLWE pair embedded in a TRLWE ciphertext, we have to take into account the extra variance induced by packing following Eq. (4). Then with our running TFHE parameter set example, Prop. 1 tells that for $\epsilon = 2^{-40}$, $l_0 = 19$. Prop. 2 however tells us that the probability of decryption error is bounded by $2^{-101.98}$ for that value. Then, if we choose $\ell = 20$, the bound drops to $2^{-72.29}$, still above our 2^{-40} target. As this is the cutoff value, we can finally settle on $\ell = 20$ (as in the previous Sect. meaning that TRLWE packing does not affect how much we can truncate in the present setting). Assuming one fully packed TRLWE, we get a ciphertext size reduction of 56.25%. Overall, the expansion factor goes from 64 (Sect. 3.2) down to 24, i.e. is reduced by a factor of 2.66.

4.4 Compressed Paillier-ElGamal (CPG)

In this section, we recall a standard variant of the BCP encryption scheme, which is commonly called Paillier-ElGamal [33, 37] and features smaller ciphertext compared to the traditional BCP encryption scheme ($3\log\mu$ bits versus $4\log\mu$). Then, we show that this variants additionally supports an efficient *compression* procedure, that allows to further reduce its ciphertext size down to $\log\mu + |m|$, where $|m|$ denotes the bitlength of the encrypted message m. This compression procedure is incompatible with the homomorphic features of the scheme, but can be used once all homomorphic operations have been computed, before sending the final ciphertext to the owner of the secret key.

A variant of BCP with shorter ciphertexts. The following variant of BCP is well-known and has roots in [21, 22]. It builds upon the fact that

- KeyGen: Let $\mu = pq$ be an RSA modulus. Choose a random $\alpha \in \mathbb{Z}^*_{\mu^2}$, a random value $d \in [1, \mathrm{ord}(\mathbb{G})]$. Set $g = \alpha^2 \bmod \mu$ and $h = g^{\mu \cdot d} \bmod \mu^2$. Return a public key $pk = (\mu, g, h)$ and a secret key $sk = d$.

- Enc: For a given message $m \in \mathbb{Z}_\mu$, a random pad $r \xleftarrow{\$} \mathbb{Z}_{\mu^2}$ return a ciphertext $\mathbf{c} = (c_0, c_1)$ such that $c_0 = g^r \bmod \mu,\ c_1 = h^r(1+\mu)^m \bmod \mu^2$.

- Dec: Compute $c = c_1(c_0)^{-\mu \cdot d} \bmod \mu^2$ and return $m = \frac{c-1}{\mu}$.

We make a few remarks on the above scheme. Compared with BCP, h is now computed as a μ-th power. This implies that $c_1 = h^r(1+\mu)^m = (g^{rd})^\mu(1+$

$\mu)^m \bmod \mu^2$ (without the component c_0) is actually a valid Paillier encryption of m. Second, the component c_0 is now given modulo μ, reducing the ciphertext size by 25%. This is without loss of security, as one can easily check that $[g^r \bmod \mu]^\mu = g^{\mu r} \bmod \mu^2$.

Eventually, security-wise, the scheme can be proven IND-CPA secure under the DCR assumption, hence it achieves identical security guarantees compared to Paillier. This follows from a sequence of straightforward game hops. We believe this proof to be essentially folklore. However, to our knowledge it has not been explicitely described anywhere, and we include it for convenience:

- **Hybrid 1:** replace $h = g^{\mu d} \bmod \mu^2$ with a uniformly random $h \xleftarrow{\$} \mathbb{Z}^*_{\mu^2}$. Under the DCR assumption, this hybrid is indistinguishable from the real key generation algorithm.
- **Hybrid 2:** replace α with a random μ-th power $\alpha \leftarrow \beta^\mu \bmod \mu^2$ for $\beta \xleftarrow{\$} \mathbb{Z}_{\mu^2}$, and $g \leftarrow \alpha^2 \bmod \mu^2$. Under the DCR assumption, this hybrid is indistinguishable from the previous one.

Then, observe that in Hybrid 2, the message m is statistically hidden given a ciphertext (c_0, c_1). Indeed, the uniform distribution over $\mathbb{Z}_{\mu^2}$ is statistically close to the uniform distribution over $\mathbb{Z}_{\mu\cdot\phi(\mu)}$, because $\mu - \phi(\mu) = p + q - 1$ is of the order of $\sqrt{\mu}$. By the chinese remainder theorem, as μ is coprime to $\phi(\mu)$, $\mathbb{Z}_{\mu\cdot\phi(\mu)}$ is isomorphic to $\mathbb{Z}_\mu \times \mathbb{Z}_{\phi(\mu)}$. Write $h = (1+n)^a g^b \bmod \mu^2$ for some (a, b) (all elements admit such a decomposition), where $a \neq 0$ and is coprime to μ with overwhelming probability. Then, observe that since g generates a subgroup of order $\phi(\mu)/4$, the ciphertext c_0 leaks only the value $r_0 = [r \bmod \phi(\mu)/4]$. In contrast, $c_1 = h^r(1+\mu)^m = (1+\mu)^{ar_1 + m \bmod \mu} g^{br_0 \bmod \phi(\mu)/4} \bmod \mu^2$, where $r_1 = [r \bmod \mu]$ is statistically indistinguishable from random given r_0 (by coprimality of μ and $\phi(\mu)$). Hence, the value $ar_1 + m \bmod \mu$ statistically hides m. This concludes the proof.

In the following, in line with previous works, we call the above scheme Paillier-ElGamal.

Distributed discrete logarithm. The scheme above enjoys shorter ciphertexts than BCP, but still larger than Paillier ($3\log\mu$ versus $2\log\mu$). In this section, we recall the *distributed discrete logarithm* procedure introduced in [33,37]. At a high level, this procedure allows two parties, given divisive shares of $(1+\mu)^m \bmod \mu^2$ over $\mathbb{Z}^*_{\mu^2}$, to non-interactively derive *substractive shares* of m over $\mathbb{Z}_\mu$. We outline the procedure DDLog_μ below.

Input. An element $u \in \mathbb{Z}^*_{\mu^2}$.
Output. A value $v \in \mathbb{Z}_\mu$.
Procedure. Write $u = u_0 + \mu \cdot u_1$, where $u_0, u_1 \in \mathbb{Z}_\mu$ denote the base-μ decomposition of u. Return $v = u_1/u_0 \bmod \mu$.

We now explain why this procedure has the intented behavior. Let u, u' denote two divisive shares over $\mathbb{Z}^*_{\mu^2}$ of $(1+\mu)^m \bmod \mu^2$; that is, $u'/u = (1+\mu)^m =$

$1 + \mu m \bmod \mu^2$. Writing $u = u_0 + \mu \cdot u_1$ and $u' = u'_0 + \mu \cdot u'_1$, we obtain

$$u'_0 + \mu \cdot u'_1 = (u_0 + \mu \cdot u_1) \cdot (1 + \mu \cdot m) \bmod \mu^2.$$

The above equation yields $u_0 = u'_0 \bmod \mu$ and $u'_1 = u_1 + u_0 m \bmod \mu$. Therefore, $m = u'_1/u'_0 - u_1/u_0 \bmod \mu$: u'_1/u'_0 and u_1/u_0 form substractive shares of m over $\mathbb{Z}_\mu$, as intended.

Compressing ciphertexts via DDLog_μ. The distributed discrete logarithm procedure implies a simple and efficient compression mechanisms for Paillier-ElGamal. The key observation is that given $c_0 = g^r \bmod \mu$, the holder of the secret key d can locally compute $u = c_0^{\mu \cdot d} = h^r \bmod \mu^2$. Then, u and c_1 form divisive shares of $c_1/u = (1 + \mu m) \bmod \mu^2$. This immediatly yields the following compression mechanism:

- $\mathsf{Compress}(c_0, c_1)$: run $v' \leftarrow \mathsf{DDLog}_\mu(c_1)$. Output (c_0, v').
- $\mathsf{Dec}'(c_0, v')$: compute $u \leftarrow c_0^{\mu \cdot d} \bmod \mu^2$ and $v \leftarrow \mathsf{DDLog}_\mu(u)$. Output $m = v' - v \bmod \mu$.

The resulting compressed ciphertext size is $2 \log \mu$, down from $3 \log \mu$, matching the size of a standard Paillier ciphertext. However, if m is known to be smaller than a bound $B < \mu/2^\lambda$ (where λ denotes a statistical security parameter), we can do better. The main observation (which is not new, the same observation was used in [33, 37]) is that with overwhelming probability, v', v form substractive shares of m *over the integers*. This stems from the following facts:

- Over the randomness of r, the value $v' = \mathsf{DDLog}_\mu(c_1) = \mathsf{DDLog}(h^r \cdot (1 + \mu m) \bmod \mu^2)$ is uniformly distributed over $\mathbb{Z}_\mu$.
- Then, if $m \leq B$, the probability that $v' - m$ causes a wrapperound modulo μ is at at most $B/\mu \leq 1/2^\lambda$, hence $v' - v = m$ over the integers.

This observation allows to further reduce the compressed ciphertext size by reducing v' modulo B:

- $\mathsf{Compress}(c_0, c_1)$: run $v' \leftarrow \mathsf{DDLog}_\mu(c_1)$ and set $v'' \leftarrow [v' \bmod B]$. Output (c_0, v'').
- $\mathsf{Dec}'(c_0, v')$: compute $u \leftarrow c_0^{\mu \cdot d} \bmod \mu^2$ and $v \leftarrow \mathsf{DDLog}_\mu(u)$. Output $m = v'' - v \bmod B$.

With this last optimization, the ciphertext size went down to $\log \mu + \log B$ bits. When B is small (e.g. $B \approx 2^{40}$ as in our application), this yields an almost twofold size improvement over a standard Paillier encryption.

We note that a similar procedure has been previously described in the context of ElGamal encryption [8]. The Paillier-ElGamal variant which we outline here has the advantage of being extremely efficient, as compression amounts only to an inversion and a product modulo μ followed by a modular reduction.

5 Switching to LHE

In this section, we investigate several approaches to switch from TLWE homomorphic ciphertexts to (more compact) linearly homomorphic ones by executing the linear part of the (T)LWE decryption function, $b - \langle \mathbf{a}, \mathbf{s} \rangle$ (recall Eq. (1)), under the target linearly homomorphic scheme. As a result of this operation, we obtain encryptions of *partial decryptions* of TFHE ciphertexts, under the target LHE. We consider several candidate LHE cryptosystems such as Paillier, Dåmgard-Jurik, EC ElGamal and our compressed variant of BCP03 (Sect. 4.4). When the properties of the LHE allows it, we also consider packing the partial decryptions of *several* TFHE ciphertexts in a single LHE ciphertext, in order to achieve better transmission efficiency when several evaluated TFHE ciphertexts have to be transmitted. To reduce the number of bits needed to encode a partial decryption (and hence be able to pack more partial decryptions per LHE ciphertext) we also investigate the use of this technique in conjunction to the ℓ-truncation technique which we introduced in Sect. 4.1.

Overall, the most appropriate choice for the LHE depends on several factors such as its plaintext/ciphertext ratio size, how many partial decryptions can be packed in its plaintexts and the conditions under which it can decrypt efficiently (as some LHE require solving a discrete log in their decryption function). Because of all these degrees of freedom, some LHE are more appropriate than others for the purpose of transmitting a given number of evaluated TFHE ciphertexts.

5.1 A generic switching algorithm

Let $\mathcal{E}_H$ denote an instance of TFHE, and $\mathcal{E}_L$ a target LHE. Let μ denote the plaintext modulus of $\mathcal{E}_L$ and v denote the number of bits necessary to represent a partial decryption (because μ is generally much greater than q, the partial decryptions are *not* computed modulo q and we have to cope for the carries occurring in their computation). Then up to

$$M = \left\lfloor \frac{\lfloor \log_2 \mu \rfloor}{v} \right\rfloor \tag{11}$$

partial decryptions can be packed into a single LHE ciphertext. Switching then works as follows. Let $s \in \mathbb{B}^n$ denote $\mathcal{E}_H$'s secret key and let

$$c_s^{(i)} = \mathcal{E}_L.\mathsf{Enc}(s_i),$$

for $i \in \{0, ..., n-1\}$ denote encryptions of its coefficients under $\mathcal{E}_L$. We further denotes by $(\mathbf{a}^{(j)}, b^{(j)})$, $j \in \{0, ..., M-1\}$, the M TLWE pairs that we wish to convert. The conversion algorithm then start with ciphertext

$$c = \mathcal{E}_L.\mathsf{Enc}(0).$$

For $i = 0$ to $n-1$, we then perform,

$$c := c \oplus \left(- \sum_{j=0}^{M-1} 2^{jv} a_i^{(j)} \right) \odot c_s^{(i)}$$

where $\oplus$ and $\odot$ respectively denote the addition and the multiplication-by-a-constant operator of $\mathcal{E}_L$ (remark that $-\sum_{j=0}^{M-1} 2^{jv} a_i^{(j)}$ lives in the clear domain of $\mathcal{E}_L$). Lastly, switching is finalized by doing[10]

$$c := c \oplus \mathcal{E}_L.\mathsf{Enc}\left(\sum_{j=0}^{M-1} 2^{jv} b^{(j)}\right). \tag{12}$$

Algorithm 2 summarizes the above. As such, the algorithm terminates with an encryption of $\sum_{j=0}^{M-1} 2^{jv}\left(b^{(j)} - \langle \mathbf{a}^{(j)}, s\rangle\right)$ but *without the modulo q which is implicit in Eq.* (1). After LHE decryption, one may recover the j-th partial decryption by doing

$$\varphi^{(j)} = \left(\left\lfloor \frac{\mathcal{E}_L.\mathsf{Dec}(c)}{2^{jv}} \right\rceil \mod 2^v\right) \mod q, \tag{13}$$

and decryption is finalized using Eq. (3). Lastly, (13) has to be slightly modified as follows when ℓ-truncation is applied,

$$\varphi^{(j)} = 2^{\ell}\left(\left\lfloor \frac{\mathcal{E}_L.\mathsf{Dec}(c)}{2^{jv'}} \right\rceil \mod 2^{v'}\right) \mod q,$$

where $v' < v$ is used instead of v in Algorithm 2.

Algorithm 2 TLWEtoLHE

Input: Encryptions of $\mathcal{E}_H$'s secret key coefficients under $\mathcal{E}_L$, $c_s^{(i)} = \mathcal{E}_L.\mathsf{Enc}(s_i)$, M TLWE pairs $(\mathbf{a}^{(j)}, b^{(j)})$.
Output: $c \in \mathcal{E}_L.\mathcal{C}$ such that c is an encryption of $\sum_{j=0}^{M-1} 2^{jv}\left(b^j - \langle \mathbf{a}^j, s\rangle\right)$.
1: $c = \mathcal{E}_L.\mathsf{Enc}(0)$
2: **for** $i = 0$, $i < n$, $i{+}{+}$ **do**,
3: $\quad c := c \oplus \left(-\sum_{j=0}^{M-1} 2^{jv} a_i^{(j)}\right) \odot c_s^{(i)}$,
4: **end for**
5: **return** $c := c \oplus \mathcal{E}_L.\mathsf{Enc}\left(\sum_{j=0}^{M-1} 2^{jv} b^{(j)}\right)$.

Considering our running TFHE parameters example of Table 1, with $q = 2^{32}$ we need $v = 42$ bits[11] to be able represent a partial decryption prior to its reduction modulo q. This number goes down to 22 bits, if we apply ℓ-truncation as proposed in Sect. 4.2 and drop 20 least significant bits in the coefficients of the LWE pairs prior to switching them (leading to a probability of erroneous decryption of 2^{-40}). We will use these numbers for illustration purpose in the next subsections.

[10] When the target LHE provides an addition-by-a-constant operator, the latter can be used in Eq. (12) instead of invoking the encryption function of the scheme.

[11] As we have to sum $n+1$ numbers (uniformly distributed) in $\{0, ..., q\}$, in the worst-case, we get $(n+1)q$ which is requires $\lceil \log_2(n+1) + \log_2 q \rceil$ bits. With n between 512 and 1024 and $q = 2^{32}$, 42 bits are conservatively required.

5.2 Switching to Paillier

Recall Sect. 2.3. As the Paillier scheme has respective plaintext and ciphertext domains $\mathbb{Z}_\mu$ and $\mathbb{Z}_{\mu^2}$ where μ is a RSA modulus. Following the previous Sect. we can pack up to $M = \left\lfloor \frac{\lfloor \log_2 \mu \rfloor}{v} \right\rfloor$ partial decryption in a single Paillier ciphertext. Let K denote a number of partial decryptions that need to be transmitted and let $r_1 = K \bmod M$ and $r_2 = \lfloor K/M \rfloor$, then we have to pack these partial decryptions M-by-M (using Algorithm 2) using r_2 Paillier ciphertexts when $r_1 = 0$ or $r_2 + 1$ such ciphertexts otherwise. When $r_1 = 0$ the expansion factor is then

$$\frac{\lceil 2 \log_2 \mu \rceil}{M \log_2 t} \tag{14}$$

(this is also the asymptotic expansion factor when $K \to \infty$) and, otherwise, the expansion factor is given by

$$\frac{(r_2 + 1)\lceil 2 \log_2 \mu \rceil}{K \log_2 t}.$$

Considering an RSA modulus on 2048 bits (the usual recommendation to achieve 128 bits security) and our running example of TFHE parameters (Table 1), we can pack around $2048/42 \approx 48$ TLWE partial decryptions per Paillier ciphertext. For $K \leq 48$ we then get an expansion factor of $4096/K$, i.e. 4096 for $K = 1$ and around 85 for $K = 48$ (which is also the asymptotic expension factor). If we apply, ℓ-truncation then we can now pack around $2048/22 \approx 93$ partial decryptions in a single Paillier ciphertext. For $K \leq 93$ we obtain an expansion factor between 4096 ($K = 1$) and 44 ($K = 93$), this later also being the asymptotic expansion factor. *Keep in mind that these latter numbers must be compared with the raw expansion factor of 32800 induced by TLWE ciphertexts: for $K = 93$ (with ℓ-truncation) the expansion thus becomes 745 times smaller.* We explore more TFHE parameters in Sect. 6.

5.3 Switching to Dåmgard-Jurik

As an alternative to Paillier (and as initially considered in [9]), we may consider using the Damgård-Jurik scheme (recall definition in Sect. 2.3) which generalizes the former scheme with $\mathbb{Z}_{\mu^y}$ and $\mathbb{Z}_{\mu^{y+1}}$ ($y > 1$) respectively as plaintext and ciphertext domains. Because the plaintext modulus is larger than for Paillier, it is possible to pack more TLWE partial decryptions into a single Dåmgard-Jurik ciphertext. Indeed, following Sect. 5.1 we can now pack up to $M = \left\lfloor \frac{\lfloor y \log_2 \mu \rfloor}{v} \right\rfloor$ partial decryption in a single Dåmgard-Jurik ciphertext. As in the previous Sect., let K denote the number of partial decryptions that need to be transmitted and let $r_1 = K \bmod M$ as well as $r_2 = \lfloor K/M \rfloor$, then we have to pack these partial decryptions M-by-M (using Algorithm 2) using r_2 Dåmgard-Jurik ciphertexts when $r_1 = 0$ or $r_2 + 1$ such ciphertexts otherwise. When $r_1 = 0$ the expansion factor is then

$$\frac{\lceil (y + 1) \log_2 \mu \rceil}{M \log_2 t} \tag{15}$$

(*for a fixed* y, this is also the asymptotic expansion factor when $K \to \infty$) and, otherwise, the expansion factor is given by

$$\frac{(r_2+1)\lceil (y+1)\log_2 \mu \rceil}{K \log_2 t}.$$

Interestingly, Dåmgard-Jurik asymptotically achieves the lowest expansion factor when both K and y increase to ∞. Indeed, (15) can be approximated by $\frac{v(y+1)\log_2 \mu}{y \log_2 \mu \log_2 t} = \frac{v(y+1)}{y \log_2 t}$ leading to

$$\lim_{y \to \infty} \frac{v(y+1)}{y \log_2 t} = \frac{v}{\log_2 t},$$

which is the optimal rate achievable with the LHE switching technique.

Returning to our favorite running TFHE parameter example and considering, as for Paillier, a 2048 bit RSA modulus with $y = 2$, we can pack up to $M = 4096/42 \approx 97$ TLWE partial decryptions in a single Dåmgard-Jurik ciphertext. Then for $K = 97$ we illustratively obtain an expansion factor of around 63, this is also the asymptotic expansion factor (for fixed $y = 2$) which can then be compared with the value 85 obtained for Paillier. Lastly, putting ℓ-truncation into the picture, leads to $M = 4096/22 \approx 186$ and an asymptotic expansion factor (again for fixed $y = 2$) of around 33. We compare the different methods and explore more TFHE parameters in Sect. 6.

5.4 Switching to compressed Paillier-ElGamal

We now consider packing TLWE partial decryptions within ciphertexts of the compressed Paillier-ElGamal (CPG) scheme that we introduced in Sect. 4.4. As for Paillier (and Dåmgard-Jurik), this scheme has a plaintext modulus μ. However, the size of a ciphertext with an l-bits payload ($l \leq \log_2(\mu)$) is only $l + \lceil \log_2 \mu \rceil$. As a consequence, this scheme is best used when small numbers of TLWE partial decryptions have to be transmitted. As in the Paillier case, we can pack up to $M = \left\lfloor \frac{\lfloor \log_2 \mu \rfloor}{v} \right\rfloor$ partial decryption in a single CPG ciphertext and the two schemes achieve the same asymptotic expansion factor. However, when we wish to pack only $K \leq M$ partial decryptions in a CPG ciphertext, the resulting expansion factor is

$$\frac{Kv + \lceil \log_2 \mu \rceil}{K \log_2 t}$$

compared to the Paillier case which, recall Eq. (14), gives $\frac{2 \log_2 \mu}{K \log_2 t}$.

Returning again to our running TFHE parameters example of Table 1 (and reusing the numbers from the end of Sect. 5.2) we can pack up to 48 (w/o ℓ-truncation) or 93 (with ℓ-truncation) partial decryptions in a a single CPG ciphertext. Without ℓ-truncation, for $K = 5$, 10, 40 we then obtain respective expansion factors of 451, 246 and 93 (versus 819, 409 and 102 for Paillier). With ℓ-truncation into the picture, again for $K = 5$, 10, 40, we respectively end up with 431, 226 and 73 (also versus 819, 409 and 102 for Paillier). More comparison are provided in Sect. 6 on various parameter sets.

5.5 Switching to EC ElGamal

For completeness, we also briefly consider packing partial decryptions in EC ElGamal ciphertexts (recall the definition. in Sect. 2.3). This scheme is the most compact that we consider in this work but this compacity comes with the price of having to solve a discrete logarithm in the scheme decryption function. As such, we can only use it to encrypt small payloads (say, with a length of around or slightly above 40 bits). As a consequence, it is not possible to pack many TLWE partial decryptions in an EC ElGamal ciphertext and this reduces its applicability to settings when no more than one or two evaluated TLWE ciphertexts have to be transmitted (using ℓ-truncation to decrease the number of bits needed to represent their partial decryptions). Still, in such cases, it is competitive with other approaches (transferring only one TLWE ciphertext always leads the worst expansion factor as the size of the LHE ciphertext is not amortized).

Indeed, when a single TLWE ciphertext $(\mathbf{a}, b)$ has to be transmitted, switching to EC (exponential) ElGamal by executing $b - \langle \mathbf{a}, \mathbf{s} \rangle$ over that cryptosystem will lead to a ciphertext of size around $\log_2 \omega$ bits. Thus leading an expansion factor of

$$\frac{\lceil \log_2 \omega \rceil}{K \log_2(t)} \tag{16}$$

For our favorite running TFHE parameters example, with $t = 2$ and $K = 1$ we thus get 512 (which is the smallest expansion factor we obtain when transferring a single TLWE partial decryption, all the others LHE being in the thousands in that case). If we apply ℓ-truncation we can either accelerate the decryption function (only a discrete log with an upper bound of 22 bits then needs to be solved) or attempts to pack two partial decryptions (needing 44 bits in total) in a single ciphertext. In that latter case, the expansion factor gets down to 256.

6 Experimental results and comparisons

6.1 TFHE parameters

In our experimental analysis, we consider the two TFHE parameter sets given in Table 3. The first set is identical to the running example we have used so far for illustrative purpose. This first parameter set is consistent with the "standard TFHE gate bootstrapping" approach where TFHE is configured for performing operations over binary plaintexts (i.e. $t = 2$). This first parameter set achieves an error probability for bootstrapping of 2^{-154}. Our second parameter set is for $t = 16$ meaning that TFHE ciphertexts now have a 4-bits payload. This set is the most interesting because, as we shall see, the increased payload length will consistently lead to the smallest expansion factors in our experiments. Furthermore, several recent works, notably [38, 39], hint that $t = 16$ may achieve an optimal tradeoff between the bootstrapping time (which increases with t) and the number of operations (which decreases with t) required when executing

(useful) algorithms over TFHE. However, for $t = 16$, the bootstrapping error probability is increased to 2^{-46} and cannot be significantly lowered unless one is willing to increase q to 2^{64} (and also mechanically increase n). We do not consider this latter option as it would result in a large performance hit for the FHE calculation themselves. Our two parameter sets achieve 128-bits security according to the lattice-estimator and have been obtained following the methodology in [20].

Note that, since we have used our parameter set for the case where $t = 2$ has a running example throughout the paper, the present section focuses essentially on the case where $t = 16$.

t	q	n	N	l	t_{d}	B_{g}	B_{KS}	σ_{TLWE}	σ_{TRLWE}
2	2^{32}	550	1024	2	1	256	1024	$q \cdot 2.82 \times 10^{-4}$	$q \cdot 5.6 \times 10^{-8}$
16	2^{32}	750	2048	2	2	1024	1024	$q \cdot 7.7 \times 10^{-6}$	$q \cdot 9.6 \times 10^{-11}$

t	σ_{BR}	σ_{KS}	σ_{BS}	σ_{pack}
2	$q \cdot 1.12 \times 10^{-2}$	$q \cdot 1.27 \times 10^{-2}$	$q \cdot 1.69 \times 10^{-2}$	$q \cdot 1.81 \times 10^{-2}$
16	$q \cdot 3.62 \times 10^{-4}$	$q \cdot 4.92 \times 10^{-4}$	$q \cdot 6.1082 \times 10^{-4}$	$q \cdot 6.1087 \times 10^{-4}$

Table 3: Our TFHE parameter sets for $t = 2$ (binary ciphertext payload) and 16 (4-bits payloads). The full set of parameters is provided for completion and reproductibility of our results but we do not detail the meaning of them all. As of the second of the above tables, it provides the post-bootstrapping noise standard deviation σ_{BS} (which characterize the noise present in *evaluated* ciphertexts) and the post-packing standard deviation (obtained after packing N, n-dimensionnal *evaluated* TLWE ciphertexts, hence with a noise deviation of σ_{BS}, in a *single* degree-N TRLWE ciphertext). The other two deviation are post Blind Rotation (BR) and post KeySwitch (KS) but we did not need to detail these operations in this paper.

6.2 Relationship between ℓ and k

Recall Eq. (9) on page 13 which tells the number of bits ℓ that we can drop in the coefficients of a TLWE pair in function of a target probability of decryption error 2^{-k}. Since ℓ is influenced by $\log_2 k$ and because a flooring occurs in the formula (as only an integer number of bits can be dropped) we can expect that a given choice for ℓ covers a wide range of decryption error probabilities.

We illustrate this in Tables 4 (for TLWE ℓ-truncation) and 5 (for TRLWE ℓ-truncation). These tables also show that in both cases, ℓ-truncation does not prevent to achieve a $\mathrm{negl}(\lambda)$ probability of erroneous decryption although of course less bits have to be dropped than for larger probabilities of error. Following the discussion in Sect. 6.1 the probability of bootstrapping error has to be set consistently with the probability of erroneous decryption induced by ℓ-truncation. For example, since our parameter set for $t = 2$ induces a probability of bootstrapping error of 2^{-154}, Table 4 tells us that we can drop up to 18 bits

per coefficients and still achieve an *overall* probability of erroneous decryption less than 2^{-128}, i.e. FHE correctness with overwhelming probability[12].

ϵ	2^{-40}	2^{-64}	2^{-128}
t	2		
$\log_2\left(\frac{1}{n+1}\left(\frac{\Delta}{2}-\sigma\sqrt{2k\log 2}\right)+1\right)$	19.881411	19.428192	17.539266
ℓ (from Prop. 2)	20	20	18
t	16		
$\log_2\left(\frac{1}{n+1}\left(\frac{\Delta}{2}-\sigma\sqrt{2k\log 2}\right)+1\right)$	17.220401	17.153781	17.012206
ℓ (from Prop. 2)	18	18	17

Table 4: Maximum value for ℓ when ℓ-truncating an evaluated (i.e bootstrapped) TLWE ciphertext, for $t = 2, 16$ and several values for the probability of erroneous decryption ϵ.

ϵ	2^{-40}	2^{-64}	2^{-128}
t	2		
$\log_2\left(\frac{1}{n+1}\left(\frac{\Delta}{2}-\sigma\sqrt{2k\log 2}\right)+1\right)$	19.769217	19.227989	15.901458
ℓ (from Prop. 2)	20	20	16
t	16		
$\log_2\left(\frac{1}{n+1}\left(\frac{\Delta}{2}-\sigma\sqrt{2k\log 2}\right)+1\right)$	17.220382	17.153756	17.012168
ℓ (from Prop. 2)	18	18	17

Table 5: Maximum value for ℓ when ℓ-truncating a degree-N TRLWE ciphertext in which N n-dimensional *evaluated* TLWE ciphertexts have been packed, for $t = 2, 16$ and several values for the probability of erroneous decryption ϵ.

6.3 Expansion factors

We now turn our attention to the expansion factor metric. First, Tables 6 and 7 summarize the expansion factor formulas respectively for the TLWE/TRLWE-based compression techniques and the LHE-based ones. All there formulas can be found in the respective sections describing these methods.

Figure 3 provide a comparison between the expansion factor obtained by the different methods for $t = 16$ in function of K, the number of evaluated TLWE ciphertexts that have to be transmitted. Additionally, Figure 4 provides a more focused view of the same for the smaller values $K \leq 300$. Complementarily, Table 8 provides a some of the numbers behind these two Figures.

[12] This remark is important has ensuring correctness is a natural countermeasures against the recent CPA^D attacks against TFHE and other schemes [15, 16].

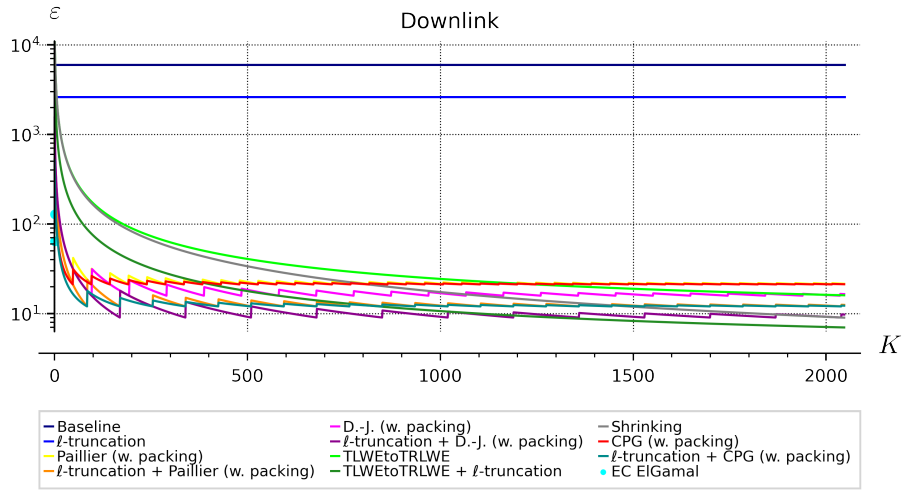

Fig. 3: Comparison of expansion factors for the downlink ciphertext compression methods studied in this paper ($t = 16$).

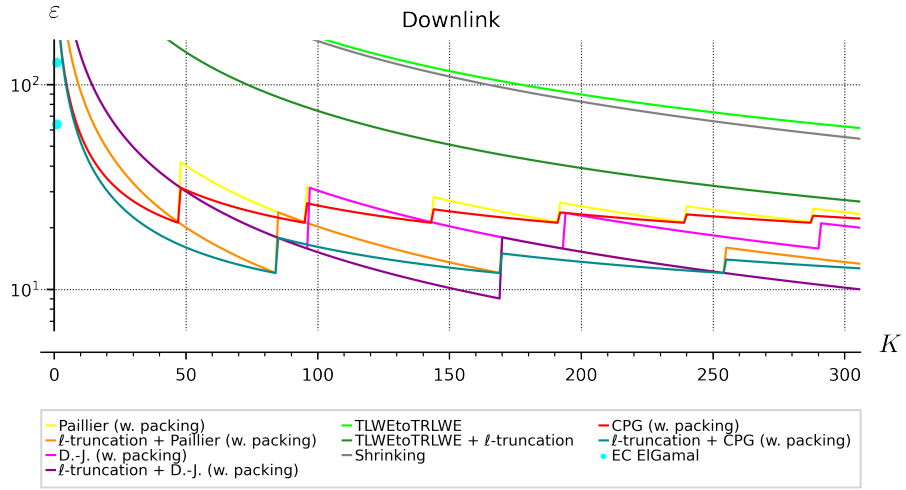

Fig. 4: Comparison of expansion factors for the downlink ciphertext compression methods studied in this paper for $K \leq 300$ ($t = 16$).

Method	Exp. factor
TLWE	$\frac{(n+1)\log_2 q}{\log_2 t}$
TLWE ℓ-truncation	$\frac{(n+1)(\log_2 q-\ell)}{\log_2 t}$
Shrinking	$\frac{(N+1)\log_2 q+K\log_2 t}{K\log_2 t}$
TLWEtoTRLWE	$\frac{(2r_2N+N+r_1)\log_2 q}{K\log_2 t}$
TLWEtoTRLWE + ℓ-truncation	$\frac{(2r_2N+N+r_1)(\log_2 q-\ell)}{K\log_2 t}$

Table 6: Expansion factor formula for the different methods based on TLWE/TRLWE when K *evaluated* TLWE ciphertexts have to be transmitted on the downlink. For RLWE-based methods, $r_1 = K \bmod N$ and $r_2 = \lfloor K/N \rfloor$ when the K n-dimensionnal TLWE are packed N-by-N in degree-N TRLWE ciphertexts.

Overall, again for $t = 16$, when $K = 1$ or 2, EC ElGamal achieves the smallest expansion factors (128 and 64 respectively). In the range $2 \leq K \leq 80$, the lowest expansion factor is achieved by performing ℓ-truncation and then switching to CPG (achieving for example an expansion factor of around 16 for $K = 50$). Up to $K \leq 1000$, Dåmgard-Jurik (with $y = 2$) leads the lowest factors, achieving for example an expansion factor of around 12 and 9 respectively for $K = 250$ and 500. For $K \geq 1000$, TRLWE packing followed by ℓ-truncation becomes the best option and leads to an expansion factor of around 7. This is summarized in Table 9. As discussed in Sect. 5.3, if we forget practicality for a moment and let the y parameter of Dåmgard-Jurik increase and apply ℓ-truncation with $\ell = 18$ (Table 4), we will eventually achieve the smallest possible expansion factor of $\frac{32-18}{4} = 3.5$. As already emphasized, *all these expansion factors must be compared to the raw expansion factor of 32800 induced by TLWE ciphertexts.*

6.4 Remarks on timings

From a timing perspective we give only a few illustrative numbers. For example, packing 1024 550-TLWE ciphertexts ($t = 2$) into a single TRLWE ciphertext takes 0.4 secs when implemented by means of TFHELib. For comparison, switching 93 20-truncated partial decryptions ($t = 2$) to a Paillier ciphertext with a 20248 bits RSA modulus takes 2.46 secs with a custom C++ Paillier implementation on a single core. Although we do not report timings for all the LHE we have considered in this paper, these numbers hint that our *downlink* compression techniques are competitive to the transciphering techniques that are used (and applicable only) on the *uplink*. As a matter of example, the most optimized implementation of AES over TFHE still run in around 1 minute.

Compression method	M	Exp. factor
Paillier (w. packing)	$\frac{\lfloor \log_2 \mu \rfloor}{v}$	$\frac{(r_2+1)\lceil 2\log_2 \mu \rceil}{K \log_2 t}$
ℓ-truncation + Paillier (w. packing)	$\frac{\lfloor \log_2 \mu \rfloor}{v-\ell}$	
Dåmgard-Jurik (w. packing)	$\frac{\lfloor y \log_2 \mu \rfloor}{v}$	$\frac{(r_2+1)\lceil (y+1)\log_2 \mu \rceil}{K \log_2 t}$
ℓ-truncation + D.-J. (w. packing)	$\frac{\lfloor y \log_2 \mu \rfloor}{v-\ell}$	
CPG (w. packing)	$\frac{\lfloor \log_2 \mu \rfloor}{v}$	$\frac{Kv+(r_2+1)\lceil \log_2 \mu \rceil}{K \log_2 t}$
ℓ-truncation + CPG (w. packing)	$\frac{\lfloor \log_2 \mu \rfloor}{v-\ell}$	$\frac{K(v-\ell)+(r_2+1)\lceil \log_2 \mu \rceil}{K \log_2 t}$
EC ElGamal	2	$\frac{\lceil \log_2 \omega \rceil}{K \log_2 t}$

Table 7: Expansion factor formula for the different LHE-based methods when K *evaluated* TLWE ciphertexts have to be transmitted on the downlink. Above, M is the number of TLWE partial decryptions which can be packed in a single LHE ciphertext (and v the number of bits required to represent one such partial decryption). Let $r_1 = K \bmod N$ and $r_2 = \lfloor K/N \rfloor$ when the K TLWE partial decryptions are packed M-by-M in LHE ciphertexts.

K	1	50	150	250	500	∞
TLWE	6008	6008	6008	6008	6008	6008
TLWE ℓ-truncation	2628.5	2628.5	2628.5	2628.5	2628.5	2628.5
Shrinking	16393	328.8	110.2	66.5	33.7	9
TLWEtoTRLWE	16392	335.6	117.2	73.5	40.7	16
TLWEtoTRLWE + ℓ-truncation	7171.5	146.8	51.2	32.1	17.8	**7**
Paillier (w. packing)	1024	40.9	27.3	24.5	22.5	21.3
ℓ-truncation + Paillier (w. packing)	1024	20.4	13.6	**12.2**	12.2	12
Dåmgard-Jurik (w. packing)	1536	30.7	20.4	18.4	18.4	15.8
ℓ-truncation + D.-J. (w. packing)	1536	30.7	**10.2**	**12.2**	**9.2**	9
CPG (w. packing)	522.5	30.9	24.1	22.7	21.7	21.1
ℓ-truncation + CPG (w. packing)	518.0	**16.2**	12.8	12.1	12.1	12
EC ElGamal	**128**	–	–	–	–	–

Table 8: Example expansion factors for each of the methods considered in this paper ($t = 16$).

K	Most compressive method
$1 \leq K \leq 2$	Switching to EC ElGamal
$2 < K \leq 85$	ℓ-truncation + switching to CPG (w. packing)
$85 < K \leq 170$	ℓ-truncation + switching to Dåmgard-Jurik (w. packing)
$170 < K \leq 255$	ℓ-truncation + switching to CPG (w. packing)
$255 < K \leq 1190$	ℓ-truncation + switching to Dåmgard-Jurik (w. packing)
$K > 1190$	TLWEtoTRLWE + ℓ-truncation

Table 9: Most appropriate compression methods in function of K, the number of evaluated TLWE ciphertexts that have to be transmitted.

7 Conclusion

In this paper, we have proposed and experimentally studied a versatile and practical toolbox to address the issue of compressing *evaluated* (T)FHE ciphertexts, i.e. encrypted *results* obtained following the FHE evaluation of some useful function, to minimize their *downlink* transmission footprint towards decryption. To the best of knowledge, while this issue is very important to FHE practice, it has so far been largely overshadowed in the literature by the issue of compressing *input* FHE ciphertexts, i.e. encrypted *inputs* towards the FHE evaluation of some useful function, to minimize their *uplink* transmission footprint from encryption. Still, the two issues are very different in nature and their solutions require different corpus of techniques and tools.

As key takeaways, we have revealed the regimes in which the techniques we have studied are best applicable, leading to the following concrete recommendations:

- Switching to EC ElGamal is the most compact options for transmitting a single evaluated TFHE ciphertext.
- Switching to our new compressed variant of BCP03 (with several partial decryptions packed in each ciphertext) is the most communication efficient option for transmitting up to around 100 evaluated TFHE ciphertexts.
- Above this value, we recommend switching to Damgard-Jurik (also with several partial decryptions packed in each ciphertext) for compressing larger number of evaluated TFHE ciphertexts although this approach may eventually become too computationally costly and, in that case, TRLWE packing will provide relatively similar compression factors with a much lower computational cost.

Additionnally, all these approaches can be combined with a simple precision reduction technique which, we have shown, still allows to keep a manageable probability of erroneous decryption. This technique allows to pack more partial decryptions in each LHE ciphertext and thus further enhance compression.

Compared to the previous works which have essentially focused on asymptotic rates from a more theoretical viewpoint, all these approaches are practically applicable.

As a concluding remark, let us emphasize that the techniques developed in this paper are applicable and beneficial only to LWE-based schemes such as TFHE. This is so for several reasons: first the LHE that we are considering in this paper have a plaintext domain which is too small to absorb the large N typically used for RLWE schemes such as BFV or BGV (which then achieve relatively low expansion factors of $2\log_2 q/\log_2 t$ for both uplink and downlink[13]) and by default fall in the large K regime in the terminology of Sect. 6). This is also true for the ℓ-truncation technique we introduced. Indeed, as it significantly increases the ciphertext noise, it can be applied only to schemes with an efficient bootstrapping procedure (as TFHE) which then allows to apply it on evaluated ciphertexts with a sufficient noise margin. Trying to apply this technique in the SHE setting for BFV or BGV would then require larger parameters and would most likely cancel the benefits of using ℓ-truncation in the first place. As a perspective, developing compression techniques practically applicable to partially-filled *evaluated* RLWE ciphertexts is an interesting follow up research question.

References

1. Ajtai, M.: Representing hard lattices with $o(n \log n)$ bits. In: STOC. pp. 94–103 (2005)
2. Albrecht, M.R., Grassi, L., Perrin, L., Ramacher, S., Rechberger, C., Rotaru, D., Roy, A., Schofnegger, M.: Feistel structures for mpc, and more. In: Sako, K., Schneider, S.A., Ryan, P.Y.A. (eds.) Computer Security - ESORICS 2019 - 24th European Symposium on Research in Computer Security, Luxembourg, September 23-27, 2019, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11736, pp. 151–171. Springer (2019). https://doi.org/10.1007/978-3-030-29962-0_8, https://doi.org/10.1007/978-3-030-29962-0_8
3. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: EUROCRYPT. pp. 430–454 (2015)
4. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. A major revision of an IACR publication in CRYPTO 2014 (2014)
5. Bendoukha, A., Boudguiga, A., Sirdey, R.: Revisiting stream-cipher-based homomorphic transciphering in the TFHE era. In: Proceedings of the 14th International Symposium on Foundations and Practice of Security, Paris, France, December 7-10, 2021. pp. 19–33
6. Bendoukha, A., Clet, P., Boudguiga, A., Sirdey, R.: Optimized stream-cipher-based transciphering by means of functional-bootstrapping. In: Proceedings of the 37th Annual IFIP WG 11.3 Conference DBSec, Sophia-Antipolis, France, July 19-21, 2023. pp. 91–109
7. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical GapSVP. Advances in Cryptology – CRYPTO 2012
8. Brakerski, Z., Branco, P., Döttling, N., Garg, S., Malavolta, G.: Constant ciphertext-rate non-committing encryption from standard assumptions. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 58–87. Springer, Cham (Nov 2020). https://doi.org/10.1007/978-3-030-64375-1_3

[13] Provided of course that these ciphertexts have no or only few unused slots.

9. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. Theory of Cryptography: 17th International Conference, TCC 2019, Nuremberg, Germany, 1-5 December 2019
10. Brakerski, Z., Gentry, C., Halevi, S.: Packed ciphertexts in LWE-based homomorphic encryption. International Conference on Theory and Practice of Public Key Cryptography (2013)
11. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. ITCS'12: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, 8-10 January 2012
12. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. TCS '14: Proceedings of the 5th conference on Innovations in theoretical computer science, 12-14 January 2014
13. Bresson, E., Catalano, D., Pointcheval, D.: A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. Advances in Cryptology - ASIACRYPT 2003
14. Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., Sirdey, R.: Stream Ciphers: A Practical Solution for Efficient Homomorphic-Ciphertext Compression. Journal of Cryptology **31**(3), 885–916 (Jul 2018). https://doi.org/10.1007/s00145-017-9273-9, https://hal.inria.fr/hal-01650012
15. Checri, M., Sirdey, R., Boudguiga, A., Bultel, J.P.: On the practical cpad security of "exact" and threshold FHE schemes. In: CRYPTO (2024)
16. Cheon, J.H., Choe, H., Passelègue, A., Stehlé, D., Suvanto, E.: Attacks against the IND-CPAD security of exact FHE schemes. Tech. Rep. 127, IACR ePrint (2024)
17. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. Advances in Cryptology – ASIACRYPT 2017
18. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption library (August 2016), https://tfhe.github.io/tfhe/
19. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. Advances in Cryptology – ASIACRYPT 2016
20. Clet, P.E., Boudguiga, A., Sirdey, R., Zuber, M.: ComBo: A Novel Functional Bootstrapping Method for Efficient Evaluation of Nonlinear Functions in the Encrypted Domain, pp. 317–343 (2023)
21. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Berlin, Heidelberg (Apr / May 2002). https://doi.org/10.1007/3-540-46035-7_4
22. Damgård, I., Groth, J., Salomonsen, G.: The theory and implementation of an electronic voting system. Secure Electronic Voting pp. 77–99 (2003)
23. Dobraunig, C., Grassi, L., Helminger, L., Rechberger, C., Schofnegger, M., Walch, R.: Pasta: A case for hybrid homomorphic encryption. Cryptology ePrint Archive (2021)
24. Dåmgard, I., Jurik, M., Nielsen, J.B.: A generalization of paillier's public-key system with applications to electronic voting. International Journal of Information Security, Volume 9 (2010)
25. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Paper 2012/144 (2012)
26. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. Advances in Cryptology – CRYPTO 2013

27. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 75–92. Springer (2013)

28. Hoffmann, C., Méaux, P., Ricosset, T.: Transciphering, using filip and TFHE for an efficient delegation of computation. In: Bhargavan, K., Oswald, E., Prabhakaran, M. (eds.) Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12578, pp. 39–61. Springer (2020). https://doi.org/10.1007/978-3-030-65277-7_3, https://doi.org/10.1007/978-3-030-65277-7_3

29. Méaux, P., Carlet, C., Journault, A., Standaert, F.: Improved filter permutators for efficient FHE: better instances and implementations. In: Hao, F., Ruj, S., Gupta, S.S. (eds.) Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11898, pp. 68–91. Springer (2019). https://doi.org/10.1007/978-3-030-35423-7_4, https://doi.org/10.1007/978-3-030-35423-7_4

30. Méaux, P., Journault, A., Standaert, F., Carlet, C.: Towards stream ciphers for efficient FHE with low-noise ciphertexts. In: EUROCRYPT. pp. 311–343 (2016)

31. Menezes, A.: Elliptic curve public key cryptosystems. Springer Science and Business Media, Volume 234 (1993)

32. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. Advances in Cryptology – EUROCRYPT 2016

33. Orlandi, C., Scholl, P., Yakoubov, S.: The rise of paillier: Homomorphic secret sharing and public-key silent OT. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 678–708. Springer, Cham (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_24

34. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. Advances in Cryptology — EUROCRYPT '99

35. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. Advances in Cryptology - CRYPTO 2008

36. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (2005)

37. Roy, L., Singh, J.: Large message homomorphic secret sharing from DCR and applications. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part III. LNCS, vol. 12827, pp. 687–717. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84252-9_23

38. Trama, D., Clet, P.E., Boudguiga, A., Sirdey, R.: Designing a general-purpose 8-bit (T)FHE processor abstraction. Tech. Rep. 20241201, IACR ePrint (2024)

39. Trama, D., Clet, P., Boudguiga, A., Sirdey, R.: A homomorphic AES evaluation in less than 30 seconds by means of TFHE. In: Proceedings of the 11th Workshop on Encrypted Computing & Applied Homomorphic Cryptography, Copenhagen, Denmark, 26 November 2023. pp. 79–90

A Appendix

A.1 Proof of Correctness of Shrinking for TRLWE

Following [9], we provide further details on the Shrinking approach (Sect. 3.3) for TRLWE ciphertexts compression. We know that $\mathbf{b}-\mathbf{s}\cdot\mathbf{a} = \mathbf{m}+\mathbf{e}$, where $\mathbf{m}+\mathbf{e} = \left(\frac{q}{t}m_0 + e_0, \ldots, \frac{q}{t}m_{N-1} + e_{N-1}\right)$ and, as $e_i \sim \mathcal{N}\left(0, \sigma^2_{\text{TLWE}}\right)$ for $i \in [0, N-1]$, the $\mathbf{e}$ can be bounded $\mathbf{e} \leq N\sigma_{\text{TLWE}}$. Then since the TRLWE decryption is performed slot-by-slot, we get $b_i - v_i = \frac{q}{t}m_i + e_i$. This implies that $b_i = v_i + \frac{q}{t}m_i + e_i$ and given that $b_i + r \notin \left[\frac{\Delta}{2} - B, \frac{\Delta}{2} + B\right] \cup \left[-\frac{\Delta}{2} - B, -\frac{\Delta}{2} + B\right]$ and $e_i \in [-B, B]$, it holds that

$$\begin{aligned}\lceil b_i + r \rfloor_t &= \lceil b_i + r - e_i \rfloor_t \\ &= \left\lceil v_i + \frac{q}{t}m_i + r \right\rfloor_t \\ &= (\lceil v_i + r \rfloor_t + m_i) \mod t.\end{aligned}$$

Then it holds that $(\lceil b_i + r \rfloor_t - \lceil v_i + r \rfloor_t) \mod t = m_i$. For $i \in [0, N-1]$ the decryption of m_i is correct if $b_i + r \notin \left[\frac{\Delta}{2} - B, \frac{\Delta}{2} + B\right] \cup \left[-\frac{\Delta}{2} - B, -\frac{\Delta}{2} + B\right]$ and $r \notin \left[\frac{\Delta}{2} - b_i - B, \frac{\Delta}{2} - b_i + B\right] \cup \left[-\frac{\Delta}{2} - b_i - B, -\frac{\Delta}{2} - b_i + B\right]$. Given that the set U of all forbidden choices of r has less than q elements, we can find an $r \in \mathbb{Z}_q$ which satisfies all constraints as $|U| \leq N \cdot 2tB$ and since $q > 2tNB$, holds that $\mathbb{Z}_q \setminus U \neq \emptyset$. For example, it is true for $q = 2^{32}$ as for $N = 1024$, $B = 2^{17}$ $(t = 2)$: $\log_2(2tNB) = 29$ and for $N = 2048$, $B = 2^{15}$ $(t = 16)$: $\log_2(2tNB) = 31$.