Published in E. Oswald, Ed., *Topics in Cryptology – CT-RSA 2024*, vol. 14643 of *Lecture Notes in Computer Science*, pp. 277-291, Springer, 2024.

TFHE Public-Key Encryption Revisited

Marc Joye

Zama, Paris, France
marc@zama.ai

Abstract. Fully homomorphic encryption allows directly processing encrypted data without having to decrypt it. The result of the computation is encrypted, typically under the same key. This unique feature offers a strong form of privacy. A service provider can so provide the same service but without ever seeing the user's data. Examples of application include [privacy-preserving] preventive medicine, facial recognition or voice assistants. Fully homomorphic encryption can also be used to solve the privacy issues of the blockchain.
This paper introduces a public-key variant of fully homomorphic encryption scheme TFHE. The output ciphertexts are of LWE type and compatible with TFHE. Interestingly, the public key is much shorter and the resulting ciphertexts are less noisy. The security of the scheme holds under the standard RLWE assumption. Several variations and extensions are also described. The proposed scheme has been integrated in fhEVM, a protocol enabling developers to create encrypted on-chain smart contracts.

Keywords: Fully homomorphic encryption (FHE) · Public-key encryption · Learning with errors (LWE) · Ring LWE (RLWE) · TFHE cryptosystem

1 Introduction

TFHE and its variants (e.g., [6,4]) are natively *private-key* fully homomorphic encryption schemes. The same key is used to encrypt or to decrypt messages. As already demonstrated in [8, § 6.1] (see also [12,3]), certain private-key homomorphic encryption schemes can be turned into a public-key encryption scheme by providing encryptions of zero. A more general result by Rothblum is provided in [17].

FROM PRIVATE-KEY TO PUBLIC-KEY ENCRYPTION. If $[\![\cdot]\!]_{\mathsf{sk}}$ denotes a probabilistic [private-key] homomorphic encryption algorithm, the public encryption key consists of z encryptions of 0; i.e., $\mathsf{pk} = \big(a_1 \leftarrow [\![0]\!]_{\mathsf{sk}}, \dots, a_z \leftarrow [\![0]\!]_{\mathsf{sk}}\big)$. Let $\boxplus$ denote the ciphertext addition. The public-key encryption of a plaintext m then proceeds as follows:

- Draw a random bit-string $(r_1, \dots, r_z) \xleftarrow{\$} \{0,1\}^z$;
- Compute a randomized encryption of zero as $S \leftarrow \boxplus_{i=1}^{z} r_i\, a_i$;

– Compute a trivial[1] encryption of m and get $M \leftarrow [\![m]\!]_{\mathsf{sk}}$;
– Output the ciphertext $C \leftarrow S \boxplus M$.

Noting that $C = [\![m]\!]_{\mathsf{sk}}$, the ciphertext C can be decrypted using the private key sk.

Application to TFHE. In the case of TFHE [4] (see also [10]), the private decryption key is an n-bit string $\boldsymbol{s} = (s_1, \dots, s_n)$. Let two positive integers q and t with $t < q$. With the previous construction, the matching public encryption key is

$$\left\{(\boldsymbol{a_i}, b_i) \in (\mathbb{Z}/q\mathbb{Z})^n \times \mathbb{Z}/q\mathbb{Z}\right\}_{1 \le i \le z}$$

where

$$\begin{cases} \boldsymbol{a_i} \xleftarrow{\$} (\mathbb{Z}/q\mathbb{Z})^n \\ b_i \leftarrow e_i + \sum_{j=1}^{n} (\boldsymbol{a_i})_j \, s_j \pmod q \end{cases} ;$$

e_i denotes an independent noise error term (typically drawn from a Gaussian distribution centered around 0) for security reasons and $(\boldsymbol{a_i})_j$ denotes the j-th component of vector $\boldsymbol{a_i}$. The encryption of a plaintext $m \in \mathbb{Z}/t\mathbb{Z}$ is given by $\boldsymbol{c} = (\boldsymbol{a}, b) \in (\mathbb{Z}/q\mathbb{Z})^{n+1}$ with

$$\boldsymbol{a} = \sum_{i=1}^{z} r_i \, \boldsymbol{a_i} \quad \text{and} \quad b = \sum_{i=1}^{z} r_i \, b_i + \Delta m$$

where $\Delta = q/t$. This assumes that t divides q. If not, an option is for example to define $\Delta = \lfloor q/t \rfloor$ (flooring), $\Delta = \lceil q/t \rceil$ (ceiling), or $\Delta = \lceil q/t \rfloor$ (rounding). An example of plaintexts encoded using the flooring function is given in [15, Sect. 5] for $t = 2$. Observe that $(\boldsymbol{0}, \Delta m) \in (\mathbb{Z}/q\mathbb{Z})^{n+1}$ is a trivial encryption of m.

Remark 1. Using matrix notation with vectors as column matrices, if we view the public key as the pair $\mathsf{pk} = (\mathbf{A}, \boldsymbol{b})$ with

$$\mathbf{A} = \begin{pmatrix} (\boldsymbol{a_1})_1 & \dots & (\boldsymbol{a_z})_1 \\ \vdots & & \vdots \\ (\boldsymbol{a_1})_n & \dots & (\boldsymbol{a_z})_n \end{pmatrix} \in (\mathbb{Z}/q\mathbb{Z})^{n \times z} \quad \text{and} \quad \boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_z \end{pmatrix} \in (\mathbb{Z}/q\mathbb{Z})^z$$

where $\boldsymbol{b} = \mathbf{A}^\intercal \boldsymbol{s} + \boldsymbol{e}$, then ciphertext $\boldsymbol{c}$ can be expressed as $\boldsymbol{c} = (\boldsymbol{a}, b)$ with $\boldsymbol{a} = \mathbf{A}\,\boldsymbol{r}$ and $b = \boldsymbol{b}^\intercal \boldsymbol{r} + \Delta m$ where $\boldsymbol{r} = \begin{pmatrix} r_1 & \dots & r_z \end{pmatrix}^\intercal \in (\mathbb{Z}/q\mathbb{Z})^z$.

The decryption of a ciphertext $\boldsymbol{c} = (a_1, \dots, a_n, b) \in (\mathbb{Z}/q\mathbb{Z})^{n+1}$ proceeds in two steps. The first step is to recover the corresponding phase defined as

$$\phi_{\boldsymbol{s}}(\boldsymbol{c}) = b - \sum_{j=1}^{n} a_j \, s_j \bmod q$$

[1] A "trivial" encryption is an (insecure) encryption that can be obtained without the knowledge of the private key. The so-obtained ciphertext decrypts to the input plaintext.

which represents a noisy value of plaintext m. Indeed, it turns out from the definition that $\phi_{\boldsymbol{s}}(\boldsymbol{c}) = \mathit{\Delta} m + \mathrm{Err}(\boldsymbol{c})$. The second step is to remove the noise $\mathrm{Err}(\boldsymbol{c})$ to get $\mathit{\Delta} m$ and, in turn, m.

Remark 2. The above description makes use of the ring $\mathbb{Z}/q\mathbb{Z}$. TFHE and the likes can similarly be defined over the discretized torus $\mathbb{T}_q = \frac{1}{q}\mathbb{Z}/\mathbb{Z}$.

Parameter selection. In order to have a sufficient security margin, the left-over hash lemma teaches that the value of z should verify

$$z = (n+1)\,|q|_2 + \kappa\ ;$$

the additional term κ, where κ is the security parameter, accounts for the corresponding subset-sum problems. $|\cdot|_2$ denotes the binary length.

Performance analysis. For a random variable X, its expectation is denoted by $\mathbb{E}[X]$ and its variance by $\mathrm{Var}(X)$; see Section A. Assuming that the noise e_i ($1 \le i \le n$) is Gaussian, centered around zero and that its variance is bounded by the same threshold $\sigma^2 = \mathrm{Var}(e_i)$, the noise variance in an output ciphertext—where $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^z$—is of $\frac{1}{2} z\sigma^2$. In the worst case, $\boldsymbol{r} = (1, 1, \dots, 1)$ and $\mathrm{Var}(\mathrm{Err}(\boldsymbol{c})) = z\sigma^2$.

Proof. Let $\boldsymbol{c}$ denote the output ciphertext. It is easy to check that $\phi_{\boldsymbol{s}}(\boldsymbol{c}) = \sum_{i=1}^{z} r_i\, e_i + \mathit{\Delta} m$ and thus $\mathrm{Err}(\boldsymbol{c}) = \sum_{i=1}^{z} r_i\, e_i$. Noting that for a uniform bit b in $\{0,1\}$, $\mathbb{E}[b] = 1/2$ and $\mathrm{Var}(b) = 1/4$, it follows that $\mathrm{Var}(\mathrm{Err}(\boldsymbol{c})) = \sum_{i=1}^{z} \mathrm{Var}(r_i\, e_i) = \sum_{i=1}^{z} \big(\frac{1}{4}\,\sigma^2 + \frac{1}{4}\,0 + \sigma^2\,(\frac{1}{2})^2\big) = z\,\frac{1}{2}\sigma^2$. If $\boldsymbol{r} = (1, 1, \dots, 1)$ then $\mathrm{Var}(\mathrm{Err}(\boldsymbol{c})) = \sum_{i=1}^{z} \mathrm{Var}(e_i) = z\sigma^2$. □

Further, assuming for more efficiency that the masks $\boldsymbol{a_i}$ are derived from a random seed $\vartheta \in \{0,1\}^\kappa$ where κ is the security parameter, the size of the public encryption key is of

$$\kappa + \big((n+1)|q|_2 + \kappa\big)\,|q|_2$$

bits.

As an illustration, at the 128-bit security level, with parameters $n = 1024$, $q = 2^{64}$ and $\sigma = 2^{-25}q = 2^{39}$,[2] it follows that $z = 65728 \approx 2^{16}$. This leads to an increase of the noise variance in an output ciphertext by an expected factor of 2^{15}. With $\sigma = 2^{39}$, the standard deviation of the noise in an output ciphertext is of $2^{46.5}$. It also results that the public encryption key takes 4206720 bits, that is, about 526 kB.

Our contributions. The noise increase and, more importantly, the large size of the public key can be prohibitive for certain applications. In this paper, we

[2] Parameters were obtained from the Lattice Estimator available at https://github.com/malb/lattice-estimator.

replace the normal public-key encryption methodology for TFHE via a methodology which goes via ring LWE. This significantly reduces the size of the associated public key. While broadly applicable, the technique is essentially tailored for TFHE for two main reasons:

- The output format of the public-key encrypted messages is of the LWE type, exactly as in TFHE. In particular, the bootstrapping applies in the same way.
- The security of the resulting scheme relies on the RLWE assumption. This assumption is already required for the bootstrapping in TFHE and its programmable version [5].

This is achieved by astutely convoluting vectors in the key generation and for the associated encryption. The decryption process remains unchanged. We present an abstract version of our basic scheme that allows for more flexibility in the parameter selection and efficiency trade-offs. Furthermore, we describe efficient packing techniques when multiple plaintexts need to be encrypted. Again, the output format is of LWE type.

Outline of the paper. The rest of this paper is organized as follows. The next section introduces a specialized convolution operator and builds therefrom an efficient public-key FHE scheme. It establishes its semantic security under the RLWE assumption and analyzes its performance. This basic scheme is then abstracted and generalized Section 3. The generalized construction offers more flexibility in the parameter selection. Several variants are also presented. Section 4 studies the case of multiple plaintexts. It describes how sharing the randomness allows for more compact ciphertexts. The corresponding conversion to regular LWE ciphertexts is also presented. Finally, Section 5 concludes the paper.

2 Smaller Public Keys, Less Noisy Ciphertexts

It is useful to introduce a new vector operator. The *reverse negative wrapped convolution* of two vectors $\boldsymbol{u} = (u_1, \dots, u_n), \boldsymbol{v} = (v_1, \dots, v_n) \in \mathbb{Z}^n$ is the vector $\boldsymbol{w} = \boldsymbol{u} \circledast \boldsymbol{v} = (\boldsymbol{u} \circledast_1 \boldsymbol{v}, \dots, \boldsymbol{u} \circledast_n \boldsymbol{v}) \in \mathbb{Z}^n$ defined by

$$w_i = \boldsymbol{u} \circledast_i \boldsymbol{v} = \textstyle\sum_{j=1}^{i} u_j\, v_{n+j-i} - \sum_{j=i+1}^{n} u_j\, v_{j-i} \;\; .$$

For example, over $\mathbb{Z}$, $(1, 2, 3, 4) \circledast (5, 6, 7, 8)$ is the vector $(-48, -16, 24, 70)$.

Remark 3. For a vector $\boldsymbol{v} \in \mathbb{Z}^n$, $\overleftarrow{\boldsymbol{v}}$ denotes vector $\boldsymbol{v}$ in reverse order; i.e., if $\boldsymbol{v} = (v_1, \dots, v_n)$ then $\overleftarrow{\boldsymbol{v}} = (v_n, \dots, v_1)$. The above convolution bears its name from the classical negative wrapped convolution (a.k.a. skew circular convolution or negacyclic convolution) defined by $\boldsymbol{w} = \boldsymbol{u} * \boldsymbol{v}$ where $w_i = \sum_{j=1}^{i} u_j\, v_{i+1-j} - \sum_{j=i+1}^{n} u_j\, v_{n+1+i-j}$. Indeed, it turns out that $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \overleftarrow{\boldsymbol{v}}$.

The main properties of the reverse negative wrapped convolution are captured by the next lemma.

Lemma 1. *Given three vectors $\boldsymbol{t}, \boldsymbol{u}, \boldsymbol{v} \in \mathbb{Z}^n$, it holds that*

1. $\boldsymbol{u} \circledast \boldsymbol{v} = \overleftarrow{\boldsymbol{v}} \circledast \overleftarrow{\boldsymbol{u}}$ *;*

2. $\boldsymbol{u} \circledast_n \boldsymbol{v} = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$;
3. $\langle \boldsymbol{t} \circledast \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{t} \circledast \boldsymbol{v}, \boldsymbol{u} \rangle$.

Proof. The first property is immediate. Since $*$ is commutative, it follows that $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \overleftarrow{\boldsymbol{v}} = \overleftarrow{\boldsymbol{v}} * \boldsymbol{u} = \overleftarrow{\boldsymbol{v}} \circledast \overleftarrow{\boldsymbol{u}}$.

Now, write $\boldsymbol{t} = (t_1, \ldots, t_n)$, $\boldsymbol{u} = (u_1, \ldots, u_n)$, and $\boldsymbol{v} = (v_1, \ldots, v_n)$. From the definition, denoting [pred] $= 1$ if some predicate pred is true and [pred] $= 0$ otherwise, we can express $\boldsymbol{u} \circledast_i \boldsymbol{v}$ compactly as

$$\sum_{j=1}^{n} (-1)^{[j>i]}\, u_j\, v_{[j \le i]n+j-i} \ .$$

Plugging $i = n$, we so get $\boldsymbol{u} \circledast_n \boldsymbol{v} = \sum_{j=1}^{n} u_j\, v_j = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$.

Likewise, we also get

$$\begin{aligned}
\langle \boldsymbol{t} \circledast \boldsymbol{u}, \boldsymbol{v} \rangle &= \sum_{i=1}^{n} \Bigl(\textstyle\sum_{j=1}^{n} (-1)^{[j>i]}\, t_j\, u_{[j \le i]n+j-i} \Bigr)\, v_i \\
&= \sum_{j=1}^{n} t_j \, \Bigl(\textstyle\sum_{i=1}^{n} (-1)^{[i<j]}\, u_{[i \ge j]n+j-i}\, v_i \Bigr) \\
&= \sum_{j=1}^{n} t_j \, \Bigl(-\textstyle\sum_{i=1}^{j-1} u_{j-i}\, v_i + \sum_{i=j}^{n} u_{n+j-i}\, v_i \Bigr) \\
&= \sum_{j=1}^{n} t_j \, \Bigl(-\textstyle\sum_{i=1}^{j-1} v_{j-i}\, u_i + \sum_{i=j}^{n} v_{n+j-i}\, u_i \Bigr) \\
&= \langle \boldsymbol{t} \circledast \boldsymbol{v}, \boldsymbol{u} \rangle
\end{aligned}$$

by symmetry. □

2.1 Description

Equipped with the $\circledast$ operator, we can now present a public-key cryptosystem. Interestingly, the encryption algorithm outputs regular LWE-type ciphertexts [15]. As a consequence, the original decryption algorithm is unchanged.

A public-key LWE-type scheme

KeyGen(1^κ) On input security parameter κ, define an integer $n = 2^\eta$ for some $\eta > 0$, select positive integers t and q with $t \mid q$, let $\Delta = q/t$, and define two discretized error distributions $\hat{\chi}_1$ and $\hat{\chi}_2$ over $\mathbb{Z}$.
Sample uniformly at random a vector $\boldsymbol{s} = (s_1, \ldots, s_n) \xleftarrow{\$} \{0,1\}^n$.
Using $\boldsymbol{s}$, select uniformly at random a vector $\mathfrak{a} \xleftarrow{\$} (\mathbb{Z}/q\mathbb{Z})^n$ and form the vector $\mathfrak{b} = \mathfrak{a} \circledast \boldsymbol{s} + \boldsymbol{e} \in (\mathbb{Z}/q\mathbb{Z})^n$ with $\boldsymbol{e} \leftarrow \hat{\chi}_1{}^n$.

The plaintext space is $\mathcal{M} = \{0, 1, \ldots, t-1\}$. The public parameters are $\mathsf{pp} = \{n, \hat{\chi}_1, \hat{\chi}_2, t, q, \Delta\}$, the public key is $\mathsf{pk} = (\mathfrak{a}, \mathfrak{b})$, and the private key is $\mathsf{sk} = \boldsymbol{s}$.

Encrypt$_{\mathsf{pk}}(m)$ The public-key encryption of a plaintext $m \in \mathcal{M}$ is given by $\boldsymbol{c} = (\boldsymbol{a}, b) \in (\mathbb{Z}/q\mathbb{Z})^{n+1}$ with

$$\begin{cases} \boldsymbol{a} = \mathfrak{a} \circledast \boldsymbol{r} + \boldsymbol{e_1} \\ b = \langle \mathfrak{b}, \boldsymbol{r} \rangle + \Delta m + e_2 \end{cases}$$

for a random vector $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^n$, and where $\boldsymbol{e_1} \leftarrow \hat{\chi}_1{}^n$ and $e_2 \leftarrow \hat{\chi}_2$.

Decrypt$_{\mathsf{sk}}(\boldsymbol{c})$ To decrypt $\boldsymbol{c} = (\boldsymbol{a}, b)$, using secret decryption key $\boldsymbol{s}$, return

$$\lceil (\mu^* \bmod q)/\Delta \rfloor \bmod t$$

where $\mu^* = b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle$.

2.2 Correctness

Let $\boldsymbol{c} = (\boldsymbol{a}, b) \leftarrow \text{Encrypt}_{\mathsf{pk}}(m)$. Then, by Lemma 1, we have $b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = \langle \mathfrak{a} \circledast \boldsymbol{s} + \boldsymbol{e}, \boldsymbol{r} \rangle + \Delta m + e_2 - \langle \mathfrak{a} \circledast \boldsymbol{r} + \boldsymbol{e_1}, \boldsymbol{s} \rangle = \Delta m + e_2 + \langle \boldsymbol{e}, \boldsymbol{r} \rangle - \langle \boldsymbol{e_1}, \boldsymbol{s} \rangle + \langle \mathfrak{a} \circledast \boldsymbol{s}, \boldsymbol{r} \rangle - \langle \mathfrak{a} \circledast \boldsymbol{r}, \boldsymbol{s} \rangle = \Delta m + E$ where $E = e_2 + \langle \boldsymbol{e}, \boldsymbol{r} \rangle - \langle \boldsymbol{e_1}, \boldsymbol{s} \rangle$. Decryption correctness thus requires that $|E| < \Delta/2$.

2.3 Security

We state the semantic security [9] of the proposed cryptosystem under the RLWE assumption [12] in $\mathbb{Z}_{n,q}[X] := (\mathbb{Z}/q\mathbb{Z})[X]/(X^n + 1)$.

Definition 1 (RLWE Assumption). *Given a security parameter κ, let $n, q \in \mathbb{N}$ with n a power of 2 and let $\mathscr{s} \xleftarrow{\$} \mathbb{B}[X]/(X^n + 1)$ where $\mathbb{B} = \{0, 1\}$. Let also $\hat{\chi}$ be an error distribution over $\mathbb{Z}[X]/(X^n + 1)$; namely, over polynomials of $\mathbb{Z}[X]/(X^n + 1)$ with coefficients drawn according to $\hat{\chi}$. The* ring learning with errors (RLWE) problem *is to distinguish samples chosen according to the following distributions:*

$$\text{dist}_0(1^\kappa) = \left\{(\mathscr{a}, \mathscr{b}) \mid \mathscr{a} \xleftarrow{\$} \mathbb{Z}_{n,q}[X], \mathscr{b} \xleftarrow{\$} \mathbb{Z}_{n,q}[X]\right\}$$

and

$$\text{dist}_1(1^\kappa) = \left\{(\mathscr{a}, \mathscr{b}) \mid \mathscr{a} \xleftarrow{\$} \mathbb{Z}_{n,q}[X], \mathscr{b} = \mathscr{a}\,\mathscr{s} + \mathscr{e} \in \mathbb{Z}_{n,q}[X], \mathscr{e} \leftarrow \hat{\chi}\right\} \ .$$

The RLWE assumption *posits that for all probabilistic polynomial-time algorithms $\mathcal{R}$, the advantage*

$$\left|\Pr\left[\mathcal{R}(\mathscr{a}, \mathscr{b}) = 1 \mid (\mathscr{a}, \mathscr{b}) \xleftarrow{\$} \text{dist}_0(1^\kappa)\right] - \Pr\left[\mathcal{R}(\mathscr{a}, \mathscr{b}) = 1 \mid (\mathscr{a}, \mathscr{b}) \xleftarrow{\$} \text{dist}_1(1^\kappa)\right]\right|$$

is negligible in κ.

We identify polynomials in $\mathbb{Z}_{n,q}[X]$ with their coefficient vectors in $(\mathbb{Z}/q\mathbb{Z})^n$, and conversely. A vector $\boldsymbol{u} = (u_1, \dots, u_n) \in (\mathbb{Z}/q\mathbb{Z})^n$ corresponds to polynomial $\mathcal{u} = \sum_{i=0}^{n-1} u_{j+1}\, X^j \in \mathbb{Z}_{n,q}[X]$; the correspondence is written $\boldsymbol{u} \cong \mathcal{u}$.

The next lemma relates the corresponding operations.

Lemma 2. *Let* $\boldsymbol{u} = (u_1, \dots, u_n)$ *and* $\boldsymbol{v} = (v_1, \dots, v_n) \in (\mathbb{Z}/q\mathbb{Z})^n$. *Let also* $\mathcal{u} = \sum_{j=0}^{n-1} u_{j+1}\, X^j$ *and* $\mathcal{v} = \sum_{j=0}^{n-1} v_{j+1}\, X^j \in \mathbb{Z}_{n,q}[X]$. *Then*

$$\boldsymbol{u} \circledast \overleftarrow{\boldsymbol{v}} = \boldsymbol{v} \circledast \overleftarrow{\boldsymbol{u}} \cong \mathcal{u} \cdot \mathcal{v} \quad .$$

Proof. From Remark 3, if $*$ denotes the negative wrapped convolution, it turns out that $\boldsymbol{w} = (w_1, \dots, w_n) := \boldsymbol{u} \circledast \overleftarrow{\boldsymbol{v}} = \boldsymbol{u} * \boldsymbol{v}$ with $w_i = \sum_{j=1}^{i} u_j\, v_{i+1-j} - \sum_{j=i+1}^{n} u_j\, v_{n+1+i-j}$. Now looking at the corresponding polynomials $\mathcal{u}$ and $\mathcal{v}$, it is easily seen that their multiplication in $\mathbb{Z}_{n,q}[X] = (\mathbb{Z}/q\mathbb{Z})[X](X^n + 1)$ yields polynomial $\mathcal{w} = \sum_{j=0}^{n-1} w_{j+1}\, X^j$. Hence, we have $\boldsymbol{w} \cong \mathcal{w}$ or, equivalently, $\boldsymbol{u} \circledast \overleftarrow{\boldsymbol{v}} \cong \mathcal{u} \cdot \mathcal{v}$. The equality $\boldsymbol{u} \circledast \overleftarrow{\boldsymbol{v}} = \boldsymbol{v} \circledast \overleftarrow{\boldsymbol{u}}$ follows from Lemma 1. $\square$

Back to the encryption scheme, it is instructive to observe that the public key $\mathsf{pk} = (\mathfrak{a}, \mathfrak{b} = \mathfrak{a} \circledast \boldsymbol{s} + \boldsymbol{e})$ corresponds to a (polynomial) RLWE sample under secret key $\sum_{j=0}^{n-1} s_{n-j}\, X^j \cong \overleftarrow{\boldsymbol{s}} = (s_n, \dots, s_1)$. Under the RLWE assumption, the public key as output by the key generation algorithm is therefore pseudo-random; i.e., indistinguishable from uniform. Regarding a ciphertext $\boldsymbol{c} = (\boldsymbol{a}, b)$ with $\boldsymbol{a} = \mathfrak{a} \circledast \boldsymbol{r} + \boldsymbol{e_1}$ and $b = \langle \mathfrak{b}, \boldsymbol{r} \rangle + \Delta m + e_2$, consider the vector $\boldsymbol{b} := \mathfrak{b} \circledast \boldsymbol{r} + \boldsymbol{e_2}$ for some $\boldsymbol{e_2} \in \hat{\chi}_2{}^n$ such that $(\boldsymbol{e_2})_n = e_2$. Again, it is worth noting that the pairs $(\mathfrak{a}, \boldsymbol{a} = \mathfrak{a} \circledast \boldsymbol{r} + \boldsymbol{e_1})$ and $(\mathfrak{b}, \boldsymbol{b} = \mathfrak{b} \circledast \boldsymbol{r} + \boldsymbol{e_2})$ correspond respectively to two (polynomial) RLWE samples under 'secret key' $\sum_{j=0}^{n-1} r_{n-j}\, X^j \cong \overleftarrow{\boldsymbol{r}}$ and thus appear to be pseudo-random. The same is true for $\langle \mathfrak{b}, \boldsymbol{r} \rangle + e_2$ since, from Lemma 1, this turns out to be the n^{th} component of vector $\mathfrak{b} \circledast \boldsymbol{r} + \boldsymbol{e_2}$: $\langle \mathfrak{b}, \boldsymbol{r} \rangle + e_2 = \mathfrak{b} \circledast_n \boldsymbol{r} + (\boldsymbol{e_2})_n$. It is also important that the randomness can be re-used in multiple ciphertexts provided they are all encrypted under different keys. This follows from [2]. Indeed, when the randomness is given explicitly in a ciphertext, it is readily verified that the "reproducibility" criterion [1, Definition 9.3] is satisfied.

The semantic security under the RLWE assumption now follows by a series of hybrid games where the different RLWE samples are successively replaced with uniform samples.

2.4 Performance

The public key expands to $2n\,|q|_2$ bits. If the component $\mathfrak{a}$ of the public key is generated from a random seed, the public key only requires $n\,|q|_2 + \kappa$ bits for its storage or transmission. With the example parameters of Section 1, this amounts to 65664 bits, or about 8.2 kB.

Suppose $\hat{\chi}_i = \mathcal{N}(0, {\sigma_i}^2)$ for $i \in \{1, 2\}$. For a ciphertext $\boldsymbol{c}$ output by the encryption algorithm, from Section 2.2, the noise variance satisfies $\mathrm{Var}(\mathrm{Err}(\boldsymbol{c})) = \mathrm{Var}(e_2 + \langle \boldsymbol{e}, \boldsymbol{r} \rangle - \langle \boldsymbol{e_1}, \boldsymbol{s} \rangle) = \mathrm{Var}(e_2) + \sum_{j=1}^{n} \mathrm{Var}((\boldsymbol{e})_i\, r_i) + \sum_{j=1}^{n} \mathrm{Var}((\boldsymbol{e_1})_j\, s_j) = {\sigma_2}^2 + 2n\big({\sigma_1}^2\, \frac{1}{4} + {\sigma_1}^2\, (\frac{1}{2})^2 + \frac{1}{4}\, 0\big) = {\sigma_2}^2 + n\, {\sigma_1}^2$. Again, with the example parameters of Section 1, for ${\sigma_1}^2 = {\sigma_2}^2$, this translates in an increase of $n+1 \approx 2^{10}$ in the noise variance. With $\sigma_1 = \sigma_2 = 2^{39}$, the standard deviation of the noise in an output ciphertext is of 2^{44}. Larger values for ciphertext modulus q lead to larger gains compared to the direct approach using encryptions of 0 for the public key (Section 1).

3 Generalization

3.1 General construction

Let $\mathscr{p}$ be a monic (irreducible) polynomial of degree n. Let also $\mathfrak{R}$ and $\mathfrak{R}_q$ denote the polynomial rings $\mathbb{Z}[X]/(\mathscr{p}(X))$ and $\mathfrak{R}/(q) = (\mathbb{Z}/q\mathbb{Z})[X]/(\mathscr{p}(X))$, respectively. A polynomial $\mathscr{a} \in \mathfrak{R}$ (resp. $\mathscr{a} \in \mathfrak{R}_q$) of degree less than n and given by $\mathscr{a}(X) = \sum_{i=0}^{n-1} a_i\, X^i$ with $a_i \in \mathbb{Z}$ (resp. $a_i \in \mathbb{Z}/q\mathbb{Z}$) can be identified with its coefficient vector $\boldsymbol{a} := (a_0, a_1, \dots, a_{n-1}) \in \mathbb{Z}^n$ (resp. $\in (\mathbb{Z}/q\mathbb{Z})^n$). Over $\mathfrak{R}_q$, we let Υ_q denote the corresponding coefficient-embedding map

$$\Upsilon_q \colon \mathfrak{R}_q \xrightarrow{\sim} (\mathbb{Z}/q\mathbb{Z})^n, \mathscr{a} = \textstyle\sum_{i=0}^{n-1} a_i\, X^i \longmapsto \Upsilon_q(\mathscr{a}) = (a_0, a_1, \dots, a_{n-1}) \enspace .$$

This one-to-one correspondence defines the convolution $*$ between two vectors in $(\mathbb{Z}/q\mathbb{Z})^n$. Given $\boldsymbol{a}, \boldsymbol{b} \in (\mathbb{Z}/q\mathbb{Z})^n$, their convolution is defined as

$$\boldsymbol{a} * \boldsymbol{b} = \Upsilon_q\big({\Upsilon_q}^{-1}(\boldsymbol{a}) \cdot {\Upsilon_q}^{-1}(\boldsymbol{b})\big) \in (\mathbb{Z}/q\mathbb{Z})^n$$

where $\cdot$ denote the polynomial multiplication in $\mathfrak{R}_q$.

Interestingly, the convolution operator allows expressing an RLWE ciphertext with vectors. One advantage of RLWE-type encryption is that it comes with an efficient public-key variant. For example, adapting [7, Sect. 3.2] following [13] (see also [12]), an RLWE public-key encryption scheme can be abstracted as follows. The key generation draws at random a small secret key $\mathscr{s} \in \mathfrak{R}$ and forms the matching public key $(\mathscr{A}, \mathscr{B}) \in (\mathfrak{R}_q)^2$ where $\mathscr{A}$ is a random polynomial in $\mathfrak{R}_q$ and $\mathscr{B} = \mathscr{A} \cdot \mathscr{s} + \mathscr{e}$ for a small random noise error $\mathscr{e} \in \mathfrak{R}$. Let $t \mid q$ and $\Delta = q/t$. The public-key encryption of a plaintext $\mathscr{m} := \mathscr{m}(X) = \sum_{i=0}^{n-1} m_i\, X^i \in \mathfrak{R}_t$ is given by the pair of polynomials $(\mathscr{a}, \mathscr{b}) \in (\mathfrak{R}_q)^2$ with

$$\begin{cases} \mathscr{a} = \mathscr{A} \cdot \mathscr{r} + \mathscr{e}_1 \\ \mathscr{b} = \mathscr{B} \cdot \mathscr{r} + \Delta\, \mathscr{m} + \mathscr{e}_2 \end{cases}$$

for some small random polynomial $\mathscr{r} \in \mathfrak{R}$ and small random noise errors $\mathscr{e}_1, \mathscr{e}_2 \in \mathfrak{R}$. The decryption of ciphertext $(\mathscr{a}, \mathscr{b})$, using secret key $\mathscr{s}$, proceeds in two steps: (i) compute in $\mathfrak{R}_q$ the phase $\mathscr{b} - \mathscr{a} \cdot \mathscr{s} = \Delta\, \mathscr{m} + \mathscr{E}$ with $\mathscr{E} := \mathscr{e} \cdot \mathscr{r} + \mathscr{e}_2 - \mathscr{e}_1 \cdot \mathscr{s} \in \mathfrak{R}$, and (ii) remove $\mathscr{E}$ to get $\Delta\, \mathscr{m}$ and, in turn, $\mathscr{m} \in \mathfrak{R}_t$.

Using the convolution operator as defined above, we get the corresponding formulation using vectors. The secret key is a small vector $\boldsymbol{s} \in \mathbb{Z}^n$ and the public key is a pair of vectors $(\boldsymbol{A}, \boldsymbol{B})$ where $\boldsymbol{A}$ is a random vector in $(\mathbb{Z}/q\mathbb{Z})^n$ and

$\boldsymbol{B} = \boldsymbol{A} * \boldsymbol{s} + \boldsymbol{e} \pmod q$ for some small random vector $\boldsymbol{e} \in \mathbb{Z}^n$. Then encryption of a plaintext $\boldsymbol{m}$ seen as a vector in $(\mathbb{Z}/t\mathbb{Z})^n$ is given by the pair of vectors $(\boldsymbol{a}, \boldsymbol{b})$ in $(\mathbb{Z}/q\mathbb{Z})^n$ where

$$\begin{cases} \boldsymbol{a} = \boldsymbol{A} * \boldsymbol{r} + \boldsymbol{e_1} \\ \boldsymbol{b} = \boldsymbol{B} * \boldsymbol{r} + \Delta\, \boldsymbol{m} + \boldsymbol{e_2} \end{cases} \tag{1}$$

for some small random vector $\boldsymbol{r} \in \mathbb{Z}^n$ and small random noise errors $\boldsymbol{e_1}, \boldsymbol{e_2} \in \mathbb{Z}^n$. Next, given ciphertext $(\boldsymbol{a}, \boldsymbol{b})$, plaintext $\boldsymbol{m}$ can be recovered using secret key $\boldsymbol{s}$ from the phase $\boldsymbol{b} - \boldsymbol{a} * \boldsymbol{s} = \Delta\, \boldsymbol{m} + \boldsymbol{E} \pmod q$ where $\boldsymbol{E} := \boldsymbol{e} * \boldsymbol{r} + \boldsymbol{e_2} - \boldsymbol{e_1} * \boldsymbol{s} \in \mathbb{Z}^n$.

Three important observations are in order:

1. If b_i (resp. m_i) denotes the i-th component of vector $\boldsymbol{b}$ (resp. $\boldsymbol{m}$) in (1) then the pair $(\boldsymbol{a}, b_i)$ is an LWE-type encryption of message $m_i \in \mathbb{Z}/t\mathbb{Z}$ provided that
$$b_i - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = \Delta\, m_i + (\text{small noise}) \enspace .$$
In particular, we have
$$\begin{aligned} b_i - \langle \boldsymbol{a}, \boldsymbol{s} \rangle &= (\boldsymbol{B} * \boldsymbol{r})_i + \Delta\, m_i + (\boldsymbol{e_2})_i - \langle \boldsymbol{A} * \boldsymbol{r} + \boldsymbol{e_1}, \boldsymbol{s} \rangle \\ &= \big((\boldsymbol{A} * \boldsymbol{s} + \boldsymbol{e}) * \boldsymbol{r}\big)_i + \Delta\, m_i + (\boldsymbol{e_2})_i - \langle \boldsymbol{A} * \boldsymbol{r} + \boldsymbol{e_1}, \boldsymbol{s} \rangle \\ &= \Delta\, m_i + (\boldsymbol{A} * \boldsymbol{s} * \boldsymbol{r})_i - \langle \boldsymbol{A} * \boldsymbol{r}, \boldsymbol{s} \rangle \\ &\qquad + (\boldsymbol{e} * \boldsymbol{r})_i + (\boldsymbol{e_2})_i - \langle \boldsymbol{e_1}, \boldsymbol{s} \rangle \enspace . \end{aligned}$$
As a consequence, if the condition
$$(\boldsymbol{A} * \boldsymbol{s} * \boldsymbol{r})_i \approx \langle \boldsymbol{A} * \boldsymbol{r}, \boldsymbol{s} \rangle \tag{2}$$
is satisfied, one ends up with an LWE-type ciphertext for plaintext $m_i \in \mathbb{Z}/t\mathbb{Z}$.
2. If the public key is replaced with $(\boldsymbol{A}, \boldsymbol{B} = \boldsymbol{A} * \varphi_1(\boldsymbol{s}) + \boldsymbol{e}) \in (\mathbb{Z}/q\mathbb{Z})^n \times (\mathbb{Z}/q\mathbb{Z})^n$ for some (bijective) map $\varphi_1 \colon (\mathbb{Z}/q\mathbb{Z})^n \to (\mathbb{Z}/q\mathbb{Z})^n$ then Condition (2) relaxes to
$$(\boldsymbol{A} * \varphi_1(\boldsymbol{s}) * \boldsymbol{r})_i \approx \langle \boldsymbol{A} * \boldsymbol{r}, \boldsymbol{s} \rangle \enspace . \tag{3}$$
3. Further, the above encryption scheme is unchanged if vector $\boldsymbol{r}$ is replaced with vector $\varphi_2(\boldsymbol{r})$ for some (bijective) map $\varphi_2 \colon (\mathbb{Z}/q\mathbb{Z})^n \to (\mathbb{Z}/q\mathbb{Z})^n$. In particular, taking $\varphi_2 = \varphi_1$ and letting $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \varphi_1(\boldsymbol{v})$, Condition (3) can be written as
$$(\boldsymbol{A} \circledast \boldsymbol{s} \circledast \boldsymbol{r})_i = (\boldsymbol{A} \circledast \boldsymbol{r} \circledast \boldsymbol{s})_i \approx \langle \boldsymbol{A} \circledast \boldsymbol{r}, \boldsymbol{s} \rangle \enspace . \tag{4}$$

We argue that one can find a map φ_1 such that Condition (4) is strictly verified. Define $\boldsymbol{C} = \boldsymbol{A} \circledast \boldsymbol{r} = (C_1, \dots, C_n)$ and write $\varphi_1(\boldsymbol{s}) = (s'_1, \dots, s'_n)$. Then

$$\begin{aligned} &(\boldsymbol{C} \circledast \boldsymbol{s})_i := (\boldsymbol{C} * \varphi_1(\boldsymbol{s}))_i = \langle \boldsymbol{C}, \boldsymbol{s} \rangle \iff \\ &\qquad \Big(\Upsilon_q\Big(\big(\textstyle\sum_{j=1}^n C_j\, X^{j-1}\big) \cdot \big(\sum_{j=1}^n s'_j\, X^{j-1}\big)\Big)\Big)_i = \textstyle\sum_{j=1}^n C_j\, s_j \pmod q \enspace . \end{aligned} \tag{5}$$

The left-hand side of the last equation can be rewritten as

$$\sum_{j=1}^{n} C_j \left(\textstyle\sum_{k=1}^{n} \alpha_{j,k}\, s'_k\right) \tag{6}$$

for some $\alpha_{j,k} \in \mathbb{Z}/q\mathbb{Z}$ given by the multiplication $\cdot$ in $\mathfrak{R}_q$. Equating each multiplier of C_j yields a system of n equations, $\sum_{k=1}^{n} \alpha_{j,k}\, s'_k = s_j$ (for $1 \leq j \leq n$), from which values for $s'_1, \dots, s'_n$ can be derived and, in turn, map φ_1.

This leads to the following public-key encryption scheme. For security and efficiency reasons, we restrict quotient polynomial $\mathcal{p}(X)$ to cyclotomic polynomials $\Phi_M(X)$. We so have $\mathfrak{R}_q = (\mathbb{Z}/q\mathbb{Z})[X]/(\Phi_M(X))$ with $n = \deg(\Phi_M)$. The multiplication in $\mathfrak{R}_q$ is denoted by $\cdot$ and the corresponding convolution in $(\mathbb{Z}/q\mathbb{Z})^n$ by $*$. The 'specialized' convolution operator in $(\mathbb{Z}/q\mathbb{Z})^n$ is denoted by $\circledast$. For any two vectors $\boldsymbol{u}, \boldsymbol{v} \in (\mathbb{Z}/q\mathbb{Z})^n$, we define $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \varphi_1(\boldsymbol{v})$. With this corresponding definition of φ_1, it holds by construction that $\boldsymbol{u} \circledast_i \boldsymbol{v} = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$; see Equation (5).

A public-key LWE-type scheme (General case)

KeyGen(1^κ) On input security parameter κ, define an integer $n = \phi(M)$ for some integer M and where ϕ denotes Euler's totient function, select positive integers t and q with $t \mid q$, let $\Delta = q/t$, and define two discretized error distributions $\hat{\chi}_1$ and $\hat{\chi}_2$ over $\mathbb{Z}$.
Sample uniformly at random a vector $\boldsymbol{s} = (s_1, \dots, s_n) \xleftarrow{\$} \{0,1\}^n$. Using $\boldsymbol{s}$, select uniformly at random a vector $\mathfrak{a} \xleftarrow{\$} (\mathbb{Z}/q\mathbb{Z})^n$ and form the vector $\mathfrak{b} = \mathfrak{a} \circledast \boldsymbol{s} + \boldsymbol{e} \in (\mathbb{Z}/q\mathbb{Z})^n$ with $\boldsymbol{e} \leftarrow {\hat{\chi}_1}^n$.
The plaintext space is $\mathcal{M} = \{0, 1, \dots, t-1\}$. The public parameters are $\mathsf{pp} = \{n, \hat{\chi}_1, \hat{\chi}_2, t, q, \Delta\}$, the public key is $\mathsf{pk} = (\mathfrak{a}, \mathfrak{b})$, and the private key is $\mathsf{sk} = \boldsymbol{s}$.

Encrypt$_{\mathsf{pk}}(m)$ The public-key encryption of a plaintext $m \in \mathcal{M}$ is given by $\boldsymbol{c} = (\boldsymbol{a}, b) \in (\mathbb{Z}/q\mathbb{Z})^{n+1}$ with

$$\begin{cases} \boldsymbol{a} = \mathfrak{a} \circledast \boldsymbol{r} + \boldsymbol{e_1} \\ b = \langle \mathfrak{b}, \boldsymbol{r} \rangle + \Delta m + e_2 \end{cases}$$

for a random vector $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^n$, and where $\boldsymbol{e_1} \leftarrow {\hat{\chi}_1}^n$ and $e_2 \leftarrow \hat{\chi}_2$.

Decrypt$_{\mathsf{sk}}(\boldsymbol{c})$ To decrypt $\boldsymbol{c} = (\boldsymbol{a}, b)$, using secret decryption key $\boldsymbol{s}$, return

$$\lceil (\mu^* \bmod q)/\Delta \rfloor \bmod t$$

where $\mu^* = b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle$.

3.2 Basic scheme

Applied to the basic scheme given in Section 2, this corresponds to $M = 2^{\eta+1}$, $\mathcal{p}(X) = X^n + 1$ with $n = 2^\eta$ and, letting $\boldsymbol{s} = (s_1, \dots, s_n)$, $\varphi_1(\boldsymbol{s}) = (s_n, \dots, s_1)$.

Indeed, for $i = n$ and $\mathscr{p}(X) = X^n + 1$, left-hand side of Equation (5) becomes

$$\Bigl(\Upsilon_q\Bigl(\bigl(\textstyle\sum_{j=1}^n C_j\, X^{j-1}\bigr) \cdot \bigl(\sum_{j=1}^n s'_j\, X^{j-1}\bigr)\Bigr)\Bigr)_n = \sum_{j=1}^n C_j\, s'_{n+1-j}$$

that is, comparing with Equation (6),

$$(\alpha_{j,k})_{\substack{1\le j\le n\\1\le k\le n}} = \begin{pmatrix} 0\,0 \dots 0\,1\\ 0\,0 \dots 1\,0\\ \vdots\,\vdots\, \iddots \;\; \vdots\\ 0\,1 \dots 0\,0\\ 1\,0 \dots 0\,0 \end{pmatrix} .$$

Equating each multiplier of C_j with those of $\sum_{j=1}^n C_j\, s_j$ yields $s'_{n+1-j} = s_j$ or, equivalently, $(s'_1, \dots, s'_n) = (s_n, \dots, s_1)$; and thus $\varphi_1(\boldsymbol{s}) = (s_n, \dots, s_1)$.

The map φ_1 in the basic scheme of Section 2 is obtained by selecting $i = n$; namely, $\varphi_1(\boldsymbol{s}) = (s_n, \dots, s_1)$. However, another vector convolution operator that is 'compatible' with the multiplication in $\mathfrak{R}_q = \mathbb{Z}_{n,q}[X] \coloneqq (\mathbb{Z}/q\mathbb{Z})/(X^n + 1)$ can be used. An alternative therefore consists in choosing another value for i. For a general value for $i \neq n$, the vector $\boldsymbol{s} = (s_1, \dots, s_n)$ is mapped to

$$\begin{aligned}\varphi_1(\boldsymbol{s}) &= (s_i, \dots, s_1, -s_n, \dots, -s_{i+1})\\ &= \bigl((-1)^{[j>i]}\, s_{1+(i-j \bmod n)}\bigr)_{1\le j\le n} .\end{aligned}$$

For example, for $i = n - 1$, we get $\varphi_1(\boldsymbol{s}) = (s_{n-1}, \dots, s_1, -s_n)$. The matching specialized convolution operator is defined as $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \varphi_1(\boldsymbol{v})$ for any two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, where $*$ denotes the classical negative wrapped convolution operator.

3.3 Higher-order convolutions and more

The general construction presents the advantage that the condition n being a power of two can be relaxed. For quotient polynomial $\mathscr{p}(X) = \Phi_M(X)$, the corresponding value for n is given by the Euler's totient function of M. For example, if $M = 3^w$ then $n = 2 \cdot 3^{w-1} = 2M/3$ and $\mathscr{p}(X) = X^n + X^{n/2} + 1$. For $i = n$, $*$ corresponds to the multiplication in $(\mathbb{Z}/q\mathbb{Z})[X]/(X^n + X^{n/2} + 1)$ and

$$\begin{aligned}\varphi_1(\boldsymbol{s}) = (&s_n + s_{n/2}, s_{n-1} + s_{n/2-1}, \dots, s_{n-(n/2-1)} + s_{n/2-(n/2-1)},\\ &s_{n/2}, s_{n/2} - 1, \dots, s_{n/2-(n/2-1)})\\ = \bigl(&s_{n+1-j} + [j \le n/2]\, s_{1+(n/2-j \bmod n)}\bigr)_{1\le j\le n} .\end{aligned}$$

Again, by construction, letting $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \varphi_1(\boldsymbol{v})$, it holds that $\boldsymbol{u} \circledast_n \boldsymbol{v} = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$ for any two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. The process generalizes to any cyclotomic Φ_M.

The general construction can also be extended to the general case of an irreducible polynomial $\mathscr{p}$. The hardness of the corresponding (R)LWE-like assumptions is studied in [16] and [14]. A potential drawback of moving away from cyclotomics is a loss of efficiency.We refer the reader to [11] for an adaptation of the (programmable) bootstrapping to quotient polynomials beyond power-of-two cyclotomics.

3.4 Variants

There are a number of possible variants. Instead of selecting $t \mid q$, plaintext modulus t can be more generally chosen as an arbitrary positive integer $< q$. In this case, a plaintext m is for example encrypted as $\boldsymbol{c} = (\boldsymbol{a}, b)$ with $\boldsymbol{a} = \boldsymbol{\mathfrak{a}} \circledast \boldsymbol{r} + \boldsymbol{e_1}$ and $b = \langle \boldsymbol{\mathfrak{b}}, \boldsymbol{r} \rangle + \lfloor q/t \rfloor m + e_2$; see Section 1.

Another variant is to select private key $\boldsymbol{s}$ and/or randomizer $\boldsymbol{r}$ at random in e.g. $\{-1, 0, 1\}^n$, or in any small subset of $\mathbb{Z}/q\mathbb{Z}$.

4 Encrypting Multiple Plaintexts

When multiple plaintexts need to be encrypted, the natural way is to encrypt them individually. For Z plaintexts this requires $Z \cdot (n+1)\,|q|_2$ bits for the corresponding ciphertexts. We show in this section how to only make use of $(\lceil Z/n \rceil\, n + Z)\,|q|_2$ bits. This saves

$$(Z - \lceil Z/n \rceil) \cdot n\,|q|_2$$

bits.

Given an LWE dimension n and a convolution operator $*$ operating on n-dimensional vectors, fix an integer $i \in \{1, \dots, n\}$. This integer i defines a map φ_1 and, in turn, the matching specialized convolution operator $\circledast$ as $\boldsymbol{u} \circledast \boldsymbol{v} = \boldsymbol{u} * \varphi_1(\boldsymbol{v})$ for any two n-dimensional vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. As detailed in the previous sections, this operator $\circledast$ gives rise to a public-key encryption scheme. With the previous notations, a plaintext m is encrypted under public key $(\boldsymbol{\mathfrak{a}}, \boldsymbol{\mathfrak{b}}) \in (\mathbb{Z}/q\mathbb{Z})^{2n}$ as

$$\begin{cases} \boldsymbol{a} = \boldsymbol{\mathfrak{a}} \circledast \boldsymbol{r} + \boldsymbol{e_1} \\ b = \langle \boldsymbol{\mathfrak{b}}, \boldsymbol{r} \rangle + \Delta m + e_2 \end{cases}$$

for some $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^n$, $\boldsymbol{e_1} \leftarrow \hat{\chi}_1{}^n$, and $e_2 \leftarrow \hat{\chi}_2$. Part $\boldsymbol{a}$ is called the mask of the ciphertext and part b is called the body of the ciphertext.

When Z plaintexts, $m_1, \dots, m_Z$, need to be encrypted, they are first put in $\lceil Z/n \rceil$ bins so that each bin contains at most n plaintexts. Next, for each bin:

1. A fresh mask $\boldsymbol{a}$ is generated from a fresh randomizer $\boldsymbol{r} \xleftarrow{\$} \{0,1\}^n$ and a fresh noise vector $\boldsymbol{e_1} \leftarrow \hat{\chi}_1{}^n$ as $\boldsymbol{a} \leftarrow \boldsymbol{\mathfrak{a}} \circledast \boldsymbol{r} + \boldsymbol{e_1}$;
2. The first plaintext, say m_1, is encrypted as above; namely, by adding the body $b \coloneqq b_1 \leftarrow \langle \boldsymbol{\mathfrak{b}}, \boldsymbol{r} \rangle + \Delta m_1 + e_{2,1}$ for a fresh random noise $e_{2,1} \leftarrow \hat{\chi}_2$;
3. The remaining plaintexts in the bin (if any), say $m_2, \dots, m_L$ for some $L \leq n$, are represented by pairs of the form

$$\{(\boldsymbol{a}, b_\ell)\}_{2 \leq \ell \leq L}$$

where $\boldsymbol{a}$ is the mask generated in 1 and

$$b_\ell \leftarrow (\mathfrak{b} \circledast \boldsymbol{r})_{j_\ell} + \Delta m_\ell + e_{2,\ell} \quad (\text{for } 2 \le \ell \le L)$$

for a fresh random noise $e_{2,\ell} \leftarrow \hat{\chi}_2$ and distinct indexes $j_\ell \in \{1, \dots, n\} \setminus \{i\}$. (Note that, by construction, $(\mathfrak{b} \circledast \boldsymbol{r})_i = \langle \mathfrak{b}, \boldsymbol{r} \rangle$.)

Ciphertext $(\boldsymbol{a}, b_1)$ is an LWE-type ciphertext but ciphertexts in $\{(\boldsymbol{a}, b_\ell)\}_{2 \le \ell \le L}$ are not. To turn them into LWE-type ciphertexts the common mask $\boldsymbol{a}$ needs first to be converted into the corresponding mask $\Psi_{j_\ell}(\boldsymbol{a})$ to get the LWE-type ciphertext $(\Psi_{j_\ell}(\boldsymbol{a}), b_{j_\ell})$ for some map $\Psi_{j_\ell} : (\mathbb{Z}/q\mathbb{Z})^n \to (\mathbb{Z}/q\mathbb{Z})^n$. There is always such a map. For instance, map Ψ_{j_ℓ} can be chosen as a linear map satisfying

$$(\boldsymbol{C} \circledast \boldsymbol{s})_{j_\ell} \approx \langle \Psi_{j_\ell}(\boldsymbol{C}), \boldsymbol{s} \rangle$$

for any vector $\boldsymbol{C} = (C_1, \dots, C_n)$. An expression for ψ_{j_ℓ} can be obtained in a way similar to what is done to derive map φ_1; see Section 3.

For example, for $i = n$ and n a power of two as in Section 2.1, for a vector $\boldsymbol{x} = (x_1, \dots, x_n)$, we can define

$$\Psi_{j_\ell}(\boldsymbol{x}) = \left((-1)^{[k \le n - j_\ell]} \, x_{1 + (k + j_\ell - 1 \bmod n)} \right)_{1 \le k \le n} \ .$$

For such a choice for Ψ_{j_ℓ}, it can be verified that $(\Psi_{j_\ell}(\boldsymbol{a}), b_{j_\ell})$ is an LWE-type ciphertext encrypting plaintext m_ℓ; that is, that

$$b_{j_\ell} - \langle \Psi_{j_\ell}(\boldsymbol{a}), \boldsymbol{s} \rangle = \Delta \, m_\ell + (\text{small noise}) \ .$$

This choice of the map Ψ_{j_ℓ} therefore ensures correct decryption. It is also interesting to observe that when $i = n$, replacing j_ℓ by i yields $\Psi_i(\boldsymbol{x}) = (x_k)_{1 \le k \le n} = (x_1, \dots, x_n)$; namely, Ψ_i is the identity map.

5 Conclusion

This paper introduced a public-key variant of TFHE with ciphertexts as LWE samples. The scheme significantly improves the public-key size and lowers the noise in the resulting ciphertexts, departing from the direct approach defining as public key a set of encryptions of 0 (typically, of the order of 2^{16} LWE samples). The security of the scheme is shown to hold under the standard RLWE assumption. Generalizations and extensions of the basic construction are presented and discussed. A packing technique to get a compressed representation of multiple ciphertexts and companion conversion to their standard LWE representation are described. The proposed public-key FHE scheme has recently been integrated in the private smart-contract protocol fhEVM.

References

1. Bellare, M., Boldyreva, A., Kurosawa, K., Staddon, J.: Multi-recipient encryption schemes: How to save on bandwidth and computation without sacrificing security. IEEE Transactions on Information Theory **53**(11), 3927–3943 (2007). https://doi.org/10.1109/TIT.2007.907471

2. Bellare, M., Boldyreva, A., Staddon, J.: Randomness re-use in multi-recipient encryption schemeas. In: Desmedt, Y. (ed.) PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography. Lecture Notes in Computer Science, vol. 2567, pp. 85–99. Springer, Heidelberg (Jan 2003). https://doi.org/10.1007/3-540-36288-6_7
3. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Ostrovsky, R. (ed.) 52nd Annual Symposium on Foundations of Computer Science. pp. 97–106. IEEE Computer Society Press (Oct 2011). https://doi.org/10.1109/FOCS.2011.12
4. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. Journal of Cryptology **33**(1), 34–91 (Jan 2020). https://doi.org/10.1007/s00145-019-09319-x
5. Chillotti, I., Joye, M., Paillier, P.: Programmable bootstrapping enables efficient homomorphic inference of deep neural networks. In: Dolev, S., et al. (eds.) Cyber Security Cryptography and Machine Learning (CSCML 2021). Lecture Notes in Computer Science, vol. 12716, pp. 1–19. Springer, Heidelberg (2021). https://doi.org/10.1007/978-3-030-78086-9_1
6. Ducas, L., Micciancio, D.: FHEW: Bootstrapping homomorphic encryption in less than a second. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015, Part I. Lecture Notes in Computer Science, vol. 9056, pp. 617–640. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_24
7. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012), https://eprint.iacr.org/2012/144
8. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th Annual ACM Symposium on Theory of Computing. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407
9. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences **28**(2), 270–299 (1984)
10. Joye, M.: SoK: Fully homomorphic encryption over the [discretized] torus. IACR Transactions on Cryptographic Hardware and Embedded Systems **2022**(4), 661–692 (2022). https://doi.org/10.46586/tches.v2022.i4.661-692
11. Joye, M., Walter, M.: Liberating TFHE: Programmable bootstrapping with general quotient polynomials. In: Brenner, M., Costache, A., Rohloff, K. (eds.) Proceedings of the 10th Workshop on Encrypted Computing & Applied Homomorphic Cryptography (WAHC 2022). pp. 1–11. ACM Press (2022). https://doi.org/10.1145/3560827.3563376
12. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010. Lecture Notes in Computer Science, vol. 6110, pp. 1–23. Springer, Heidelberg (May / Jun 2010). https://doi.org/10.1007/978-3-642-13190-5_1
13. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. Journal of the ACM **6**(43), 1–35 (2013). https://doi.org/10.1145/2535925
14. Peikert, C., Pepin, Z.: Algebraically structured LWE, revisited. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019: 17th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 11891, pp. 1–23. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-36030-6_1

15. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th Annual ACM Symposium on Theory of Computing. pp. 84–93. ACM Press (May 2005). https://doi.org/10.1145/1060590.1060603
16. Rosca, M., Stehlé, D., Wallet, A.: On the ring-LWE and polynomial-LWE problems. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part I. Lecture Notes in Computer Science, vol. 10820, pp. 146–173. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9_6
17. Rothblum, R.: Homomorphic encryption: From private-key to public-key. In: Ishai, Y. (ed.) TCC 2011: 8th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 6597, pp. 219–234. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_14

A Variance and Covariance

The variance captures how much a randomly drawn variable is spread out from the average value. Formally, the variance of a random variable X is defined as

$$\mathrm{Var}(X) = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right]$$

or, equivalently, as $\mathrm{Var}(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2$.

Composition formulas. For two *independent* variables X_1 and X_2, the expectation and variance of their sum and of their product satisfy

$$\begin{cases} \mathbb{E}[X_1 + X_2] = \mathbb{E}[X_1] + \mathbb{E}[X_2] \\ \mathrm{Var}(X_1 + X_2) = \mathrm{Var}(X_1) + \mathrm{Var}(X_2) \end{cases}$$

and

$$\begin{cases} \mathbb{E}[X_1\, X_2] = \mathbb{E}[X_1]\, \mathbb{E}[X_2] \\ \mathrm{Var}(X_1\, X_2) = \mathrm{Var}(X_1)\, \mathrm{Var}(X_2) + \mathrm{Var}(X_1)\, \mathbb{E}[X_2]^2 + \mathrm{Var}(X_2)\, \mathbb{E}[X_1]^2 \end{cases} .$$

The covariance indicates the joint variability of two random variables X_1 and X_2; it is written $\mathrm{Cov}(X_1, X_2)$. In particular, the covariance is zero when X_1 and X_2 are independent.

For correlated random variables X_1 and X_2, the composition formulas generalize to

$$\begin{cases} \mathbb{E}[X_1 + X_2] = \mathbb{E}[X_1] + \mathbb{E}[X_2] \\ \mathrm{Var}(X_1 + X_2) = \mathrm{Var}(X_1) + \mathrm{Var}(X_2) + 2\,\mathrm{Cov}(X_1, X_2) \end{cases}$$

and

$$\begin{cases} \mathbb{E}[X_1\, X_2] = \mathbb{E}[X_1]\, \mathbb{E}[X_2] + \mathrm{Cov}(X_1, X_2) \\ \mathrm{Var}(X_1\, X_2) = \mathrm{Cov}({X_1}^2, {X_2}^2) \\ \qquad\qquad + \left(\mathrm{Var}(X_1) + \mathbb{E}[X_1]^2\right) \left(\mathrm{Var}(X_2) + \mathbb{E}[X_2]^2\right) \\ \qquad\qquad - \left(\mathrm{Cov}(X_1, X_2) + \mathbb{E}[X_1]\, \mathbb{E}[X_2]\right)^2 \end{cases} .$$